We would like to thank the reviewers for insightful comments which help improving the clarity.

We will include a discussion comparing AVTA with other convex hull vertices detection algorithms.

We will also include a detailed description on the binary search type method in case \gamma is unknown. Also, more information and discussion about experiment will also be included in the supplement.

We would also like to stress that the application of vertices based method discussed in the paper requires robustness property of algorithm in convex hull problem. Our work provides the first robust guarantees for vertices detection for general convex hull. Below we provide answers to some specific questions:

Reviewer 1: (**Assigned\_Reviewer\_1:**)

- How does AVTA compare in runtime to other algorithms for finding the convex hull? In particular, a standard algorithm in higher dimensions is to my understanding QuickHull, which has average complexity O(n log n). This is comparable to the (roughly...) O(n K) of AVTA for the common case of K ~= log(n). The gift wrapping algorithm I believe has complexity O(n K).

- QuickHull/giftwrapping are exact, and don't have some of the complexity surrounding the approximations arising from use of the triangle algorithm. What are the benefits of AVTA here?

Chazelle et al [10] showed that for general d dimension, the optimal complexity to compute exact convex hull is O( nlogn + n^(d/2)). The average complexity O(nlogn) for QuickHull is for low dimensions (d=2,3). (Hossein Sartipizadeha, Tyrone L. Vincent 2016). In fact , we have made a comparison with the Quickhull algorithm and will include it in the supplement. Empirically,

Computing an approximation of vertices can be much cheaper as discussed in the paper.

In addition, exact algorithms can output much more vertices than desired. In practice, one may only need a small subset of vertices with ‘high quality’ among all vertices. The AVTA algorithm find vertices progressively and achieves both efficiency and robustness for practical applications.

- Similarly, for the robustness properties: is this AVTA specific? The pruning seems like it dovetails nicely with the tolerances that can be set for the triangle algorithm, but requires some prior knowledge as to \sigma. I do like the random projections approach, but that seems somewhat unrelated to AVTA.

Firstly, we are not aware of any other robust vertex generation algorithm other than AVTA. The Fast anchor word algorithm is robust but only works for a special case of convex hull problem.

In practice we do not need to know sigma. As mentioned in the article, by using binary search we can still compute all vertices. If only the number of vertices desired is given, the AVTA can computer a super set of vertices and prune in a non-parametric way. (i.e. Greedy algorithm). This part is not discussed in detail in the paper as our analysis is based on some prior knowledge of \sigma.

- Is there further explanation of the insight for using random projections? Specifically, "If the perturbed set is randomly projected onto a lower dimensional space, it is more likely for an original vertex to still be a vertex than for a spurious vertex." I couldn't find any proof or discussion of this claim in the supplementary material.

This is an empirical observation. Investigating the theoretical support of this phenomenon is an open problem for future work.

- The AVTA+CatchWord idea seems quite interesting, but it's not clear to me to what degree AVTA and the triangle algorithm are required. Would any convex hull algorithm (+ standard linear programming to find coefficients w.r.t. vertices) also work to embed the documents?

Computing vertices by standard linear programming is inefficient and there is no guarantee for the robustness. (Arora et,al. 2012 [1]) To embed the documents, one needs to find a small set of vertices of convex hull which can well represent each points. Here, AVTA is used for this purpose.

In short, there are several interesting claims/insights presented, but it's a bit difficult to disentangle them and determine how they relate to/depend on the central AVTA technique (vs. other ways of obtaining vertices of the convex hull). It seems like much of the paper doesn't depend on AVTA specifically, and it's not obvious to me that AVTA is a better way of finding convex hulls compared to existing approaches.

Summary:

The paper presents an algorithm for computing the vertices of the convex hull of a set of points, based on a prior "Triangle Algorithm" for solving the convex hull membership problem. Conceptually, the algorithm grows a subset of known vertices by finding some point "v" outside their current hull (using the triangle algorithm), and then identifying new vertices as point(s) with maximum (signed) distance from a hyperplane separating "v" from the current vertices. The underlying triangle algorithm is approximate, which results in requiring either knowledge of the "robustness" of the hull, the number of vertices, or specifying a tolerance. The algorithm is also analyzed under input perturbations, with a pruning stage introduced to recover . It is applied to several variants of topic modeling on a semi-synthetic and a real dataset, as well as non-negative matrix factorization on a synthetic dataset.  
  
Overall, the paper is quite extensive, presenting a number of algorithms and approaches, and includes thorough analysis. The main AVTA technique itself is interesting and pleasingly geometric. However, I am somewhat uncertain as to the practical relevance of the technique, and would have appreciated a bit more discussion of how/whether AVTA and the triangle algorithm improve on existing approaches to convex hull problems. Furthermore, the experiments don't seem to indicate any benefit over existing approaches on real data. On the synthetic data only AVTA+CatchWord seems to provide a substantial improvement, and it's not clear to me whether this has anything to with AVTA or is mainly the insight of representing the documents via their coefficients w.r.t. to the hull vertices. If the latter (which seems likely to me), this is an interesting insight that would be worthy of significantly more exposition, and perhaps should have been made the focus of the paper.  
  
At the moment I am recommending weak reject. I have outlined a number of specific questions in the detailed comments section and am open to revising my score after the author's rebuttal.

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Reviewer 2 (**Assigned\_Reviewer\_4:**)

The paper proposes a new algorithm to compute the vertices of a set of points. Furthermore, the authors provide a robust version to handle small noise in the data. Experimental results show that the proposed algorithm significantly outperforms existing methods. I think this method will be useful for some machine learning applications (mainly dimensional reduction techniques such as NMF and LDA, when one is able to spend more time to compute an accurate decomposition).

Summary:

This is a good paper. See my comments above.

Reviewer 3 (**Assigned\_Reviewer\_5**)

Generally, the presentation of technical ideas in the paper is clear, although there are a few points where clarity could be improved, which are mentioned in more detail below.  
  
Technical comments:  
- Statement of Theorem 2. When I initially read this statement, I was confused by the difference between (i) and (ii). It may be worth mentioning that a binary search type method is used in Section 2, where \gamma is not known.

- How is the initial vertex of \bar{S} selected in the AVTA algorithm? The cost of doing this does not seem to be considered in the proof of Theorem 9.

The initial vertex is selected by the point with max l2 norm. This step has cost O(mn) operations which is drop in the notation.

- Experiment details:  
- Is the parameter \gamma is chosen using the binary search method + repeated calls to AVTA described in the appendix? This is unclear from the main text.

- How were the parameters K and \epsilon (appearing the algorithm AVTA+CatchWord) chosen in these experiments?  
- From the description of AVTA+CatchWord, it sounds as though the pruning step is not included. Is this correct?

Given number of desired vertices K, we use the binary search + repeated calls to AVTA so that output is a super set of vertices (more than K) then we run pruning step.

The parameter K is chosen by the number of topics in dataset. The choice of number of topics follows from Bansal, et al 2014 [5].

The description only states the high level idea of the algorithm. More detailed description will be included in the supplement.

- In general, more explanation and interpretation of the results would be welcome -- what are the suspected reasons that AVTA outperformed other methods in the experiments?

The generative model of Document word data set for semi synthetic data is similar to generating perturbed interior points by vertices of convex hull. The AVTA is a robust algorithm in solving perturbed convex hull problem which coincides the underlying generative model.

- It would have been interesting to see what the actual robustness and weak-robustness coefficients are on the datasets used in the experiments section.

Minor comments:  
- Non-exhaustive list of typos:  
- abstract: "non-negative matrix" -> add completion/factorisation?  
- After Theorem 3: "\Gamma^\* robustness" -> "\Gamma^\*-robustness"  
- Top of appendix page 2: "\gamma^\*" -> "\Gamma^\*"  
- Some short discussion around Theorem 1 would be welcome.  
- There appears to be some inconsistency in notation which is potentially confusing:  
- \hat{S} is used to denote the working set of extremal points in Section 2, whilst in Section 4, it appears to denote random projections of the input point set.  
- The \varepsilon that appears in the definition of AVTA+CatchWord is presumably different from that appearing in the discussion of the "Results on Real Data" section  
- p4, right column below Lemma 1. "From Theorem 1 we know...". Is this really from Theorem 1 and not Lemma 1?

Summary:

This paper introduces an algorithm for vertex enumeration of the convex hull of a finite set of points. Several theoretical guarantees for the algorithms are given, include a robustness theorem, subject to constraints on the geometry of the input set of points. The proposed methods are evaluated empirically on topic modeling tasks, and an image processing task.  
  
The main paper itself is well-presented. The appendix appears to be a self-contained document, with much (approximate) overlap with the main paper, although numberings of theorems etc. do not match. This reduces the clarity of the paper as a whole. The experiments demonstrate that the method performs well on the tasks evaluated from an accuracy point of view, but it seems a shame not to have more details of running time on these experiments, especially given that the majority of the theory of the paper is in relation to computational cost of the algorithm. As noted in the "Detailed Comments" section, I have several technical questions in relation to the statement and proof of Theorem 2 that require clarification.