We would like to thank the reviewers for insightful comments that will help in improving the clarity of the paper.

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 like to stress that our work provides the first robust guarantees for detection of vertices for general convex hulls. Below we provide answers to some specific questions in particular related to how AVTA compares to other convex hull algorithms.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Questions by Assigned\_Reviewer\_1:

Q1, Q2:

The stated runtime of O(n log n) or O(n K) for QuickHull and gift wrapping algorithms is only true for small dimensions, say d=2 or 3. In fact, Chazelle[10] showed that in higher dimensions, the optimal complexity to compute exact convex hull is O( n logn + n^(d/2)). If the number of vertices is small, say K, then in general the run time required is O(n^2 K). Hence, one either has to pay exponential in dimensionality of quadratic in the number of data points. AVTA gets around this barrier by providing run time guarantee that is linear in n and linear in d as well and depends polynomially on the robustness of the convex hull. This makes it a much more practical algorithm for ML type of applications. In fact, we have made a comparison with the QuickHull algorithm and we could not run Quick Hull beyond d=10 in any reasonable amount of time. We will include this result in the full version of the paper. Previously such robustness based guarantees were known for the special case when the convex hull is a simplex. Our work provides the first robustness based guarantees for the general convex hull problem.

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.

Q3:

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 paper, by using binary search we can still compute all vertices. If only the number of vertices desired is given, then AVTA can compute a super set of vertices and prune in a greedy way.

Q4:

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

Q5:

Note that standard algorithms for computing vertices by standard linear programming or other algorithms such as quick hull are inefficient cannot make use of the robustness of the convex hull. In particular, they will output \*all\* the vertices. However, to embed the documents, it is enough to find a small set of high quality or robust vertices of the convex hull that can represent each data point well. To the best of our knowledge, AVTA is the only algorithm capable of achieving this.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Questions by Assigned\_Reviewer\_5:

Technical comments:

Q2:

The initial vertex is selected by the point with the maximum l\_2 norm. This step has cost O(mn) operations and hence is accounted for in Theorem 9.

Experiment details: Q1,Q2,Q3:

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 the 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].

A pruning step is used in AVTA+CatchWord. In practice where only K vertice are needed, we first run AVTA to compute a super set of vertices then prune it in a greedy way. We iteratively pick the point which has largest distance to the convex hull, i.e. pick the K most ‘robust’ points.

Q4:

A key reason AVTA outperforms other methods is its ability to discover high quality or robust vertices efficiently. In the AVTA+Catchword implementation we show that such high quality vertices provide a good embedding of the data. Previous works on topic modeling have not considered such embeddings. Furthermore, computing such an embedding via a standard algorithm such as linear programming or Quickhull would be computationally time consuming as these methods do not distinguish between robust and non-robust vertices.