Robust Vertex Enumeration for Convex Hulls in High Dimensions

Pranjal Awasthi
Computer Science Department,
Rutgers University,
Piscataway, NJ 08854, USA
pranjal.awasthi@cs.rutgers.edu

Bahman Kalantari
Computer Science Department,
Rutgers University,
Piscataway, NJ 08854, USA
kalantar@cs.rutgers.edu

Yikai Zhang
Computer Science Department,
Rutgers University
Piscataway, NJ 08854, USA
yz422@cs.rutgers.edu

Abstract

We design a fast and robust algorithm named *All Vertex Traingle Algorithm (AVTA)* for detecting the vertices of the convex hull of a set of points in high dimensions. Our proposed algorithm is very general and works for arbitrary convex hulls. In addition to being a fundamental problem in computational geometry and linear programming, vertex enumeration in high dimensions has numerous applications in machine learning. In particular, we apply AVTA to design new practical algorithms for topic models and non-negative matrix factorization. For topic models, our new algorithm leads to significantly better reconstruction of the topic-word matrix than state of the art approaches [2, 5]. Additionally, we provide a robust analysis of AVTA and empirically demonstrate that it can handle larger amounts of noise than existing methods. For non-negative matrix factorization we show that AVTA is competitive with existing methods that are specialized for this task [3].

1 Introduction

In this paper we study the classic problem of computing the vertices of the convex hull of a given set of points in the Euclidean space. More formally, we are given as input a set S of n points $\{v_1, v_2, \dots v_n\}$ in $\mathbb{R}^m$. We are interested in computing the subset $\overline{S}$ of all the vertices of $conv(S)$, the convex hull of S. Not only is this a fundamental problem in computational geometry, state of the art algorithms for many machine learning problems rely on being able to solve this problem efficiently. Consider for instance the problem of non-negative matrix factorization (NMF) [16]. Here, given access to a data matrix A, we want to compute non-negative, low rank matrices U and V such that $A = UV$. Although in general this problem is intractable, recent results show that under a natural *separability* assumption [13] such a factorization can be computed efficiently [3]. The key insight in these works is that under the separability assumption, the rows of the matrix V will appear among the rows of A. Furthermore, the rows of V will be the *vertices* of the convex hull of rows of A. Hence, a fast algorithm for detecting the vertices will lead to a fast factorization algorithm as well.

A problem related to NMF is known as *topic modeling* [6]. Here one is given access to a large corpus of documents, with each document represented as a long vector consisting of frequency in the document of every word in the vocabulary. This is known as the bag-of-words representation. Each document is assumed to represent a mixture of up to K hidden topics. A popular generative model for such documents is the following: For every document d, a K dimensional vector θ_d is drawn from a distribution over the simplex. Typically this distribution is the Dirichlet distribution. Then, for each word in the document, a topic is chosen according to θ_d. Finally, given a chosen topic i, a word is output according to the topic distribution vector β_i. This is known as the Latent Dirichlet Allocation (LDA) model [7]. The parameters of this model consist of the topic-word matrix β so that β_i defines the distribution over words for topic i. Additionally, there are hyper parameters associated with the Dirichlet distributions generating the topic distribution vector θ_d. The topic modeling problem concerns learning the topic-word matrix β and the parameters of the topic generating distribution. Similar to NMF, the problem is intractable in the worst case but can be efficiently solved under *separability* [4]. In this context, the separability assumption requires that for each topic i, there exists an *anchor word* that has a non-zero prob-

Proceedings of the 21^{st} International Conference on Artificial Intelligence and Statistics (AISTATS) 2018, Lanzarote, Spain. PMLR: Volume 84.

ability of occurring only under topic i. Separability is an assumption that is known to hold for real world documents [4]. The key component towards learning the model parameters is a fast algorithm for finding the anchor words. The algorithm of Arora et al. [4, 2] uses the word-word covariance matrix and shows that under separability, the vertices of the convex hull of the rows of the matrix will correspond to the anchor words. Similarly, the work of [12] shows that finding the vertices of the convex hull of the document-word matrix will also lead to detection of anchor words. Both approaches rely on the vertex detection subroutine. Furthermore, in the case of topic models, the documents are limited in size and this translates to the fact that one is given a perturbation of the set S. The goal is to use this perturbed set to approximate the original vertices $\overline{S}$. Hence in this application it is crucial that the approach to finding the vertices be robust to noise.

In both the approaches mentioned above, it is enough to solve the convex hull problem in the *simplex* case, i.e., when the underlying vertices are at the corners of a simplex. However, in many applications involving computational geometry and linear programming [20] it is important to have fast algorithms that work for arbitrary convex hulls. Furthermore, our improved algorithm for topic modeling will rely on the fact that we can solve the convex hull problem beyond the simplex case.

Our Results:

(a) We design an algorithm named *All Vertex Triangle Algorithm (AVTA)* for the general problem of computing the K vertices of the convex hull of a set S of points in $\mathbb{R}^m$. In the special case when $conv(S)$ is a K-simplex, our algorithm has a the same run time as the state of the art *fast anchor words* (FAW) algorithm of [2]In addition, our algorithm is *output-sensitive* namely its computational complexity scales with the number of vertices. This is a desirable property for applications in machine learning and computational geometry.

(b) We design a variant of AVTA that is robust to noise in the data. We theoretically demonstrate the noise robustness of our algorithm under a natural assumption on S called *weak robustness*, a concept that is complementary to the notion of robustness used in [2] (See Section 3). To the best of our knowledge, this work provides the first robustness based guarantees for the general convex hull problem.

(c) We use AVTA to design a new algorithm for topic modeling that leads to significantly better reconstruction of the topic-word matrix as compared to the state of the art algorithms [2, 5]. We also empirically demonstrate that our new algorithm is more tolerant to noise in the data.

1.1 Related Work

The problem of enumerating the vertices of a convex hull has a long history and has been studied extensively in computational geometry [20]. There are various efficient algorithms that exist in low dimensions [21, 14, 8]. In high dimensions, a natural approach to solve the enumeration problem is via linear programming: *for each point solve a linear program to find out if it lies in the convex hull of the remaining points.* This approach does not scale well as the number of linear programs needed to solve the problem increase with the data size. Similarly, other existing algorithms for the enumeration problem have worst case complexity that depends exponentially or double exponentially in m, the dimensionality of the data [11, 17, 9, 10]. In this work we focus on the complexity of this problem on practical instances. We show that the runtime of our proposed algorithm, AVTA, is polynomial in *any* dimension and scales with the *robustness* of the convex hull of the points. Robustness is a property that has been studied for the enumeration problem in the context of topic modeling [2]. See Section 2 for a formal definition.

There has been a long line of work on designing algorithms for topic modeling with provable guarantees [18, 4, 1, 2, 5]. The work of [18] shows how to provably learn the topic word matrix provided each document is pure, i.e., contains only one topic. The work of Arora et al. [4] proposed a provable algorithm for learning general topic models under the *separability* assumption. Separability refers to the assumption that for each topic i, there exists an *anchor* word w_i that has a non-zero probability of appearing only in topic i. Later, the work of [2] proposed a fast and practical algorithm for learning separable topic models. The work of [5] relaxed the separability assumption and shows how to learn topic models under the presence of *catch words* and a few documents with pure topics. Catch words for topic i is a set of words S_i that appear in topic i with significantly higher frequency than in other topics. Their new algorithm called $TSVD$ leads to a much better recovery of topics in terms of the ℓ_1 error. For the problem of non-negative matrix factorization the work of [3] designed a provable algorithm under the separability assumption. The method of choice in practice is alternate minimization [16].

2 The Algorithm

Our algorithms uses, as a subroutine, the *Triangle Algorithm* described in [15]. The triangle algorithm is a simple iterative algorithm for solving the *convex hull membership problem*, a fundamental problem in linear programming and computational geometry. Given a

set of point $S = \{v_1, \ldots, v_n\} \subset \mathbb{R}^m$ with diameter R and a distinguished point $p \in \mathbb{R}^m$, typically with n much larger than m, the Triangle Algorithm tests if $p \in conv(S)$. It either computes a point in $conv(S)$ as close as desired to p, or a hyperplane that separates p from $conv(S)$. Given $u, v \in \mathbb{R}^m$ we interchangeably use $d(u, v) = \|u - v\|$. If $p \notin conv(S)$, the separating hyperplane is encoded in terms of a point known as the *witness*.

Definition 1. *A point $p' \in conv(S)$ is a p-witness if* $d(p', v_i) < d(p, v_i), \quad \forall i = 1, \ldots, n.$

Noticing that $d(\cdot, \cdot)$ is the Euclidean distance, we get that for every point $v_i \in S$, $2(p' - p) \cdot v_i > \|p'\|^2 - \|p\|^2$. It is also easy to see that $2(p' - p) \cdot p \leq \|p'\|^2 - \|p\|^2$. Hence, a witness provides a separating hyperplane.

The triangle algorithm either outputs a p-witness or a point $p' \in conv(S)$ that is as close to p as desired. Such a point is known as an ϵ-approximate solution.

Definition 2. *Given $\varepsilon \in (0, 1)$, $p' \in conv(S)$ is an ε-approximate solution if for some $v \in S$, $d(p', p) \leq \varepsilon R$.*

Theorem 1 ([15])**.** *Suppose $p \in conv(S)$. Given $\varepsilon \in (0, 1)$, the number of iterations to compute an ε-approximate solution $p_\varepsilon \in conv(S)$ is $O(1/\varepsilon^2)$. Suppose $p \notin conv(S)$. Let $\Delta = \min\{d(x, p) : x \in conv(S)\}$. The number of iterations to compute a p-witness $p_\Delta \in conv(S)$ is $O(R^2/\Delta^2)$.* □

Furthermore, each iteration of the triangle algorithm can be implemented in $O(n + m)$ time.

2.1 All Vertex Triangle Algorithm (AVTA)

Denote the set of vertices of $conv(S)$ by $\overline{S} = \{\overline{v}_1, \ldots, \overline{v}_K\}$ and let R be the diameter of S. A straightforward way to compute $\overline{S}$ is to test for each v_i if it lies in $conv(S \setminus \{v_i\})$, to within an ε precision. Thus overall this would take n times the complexity of Triangle Algorithm. Although in the worst case this cannot be improved upon, we next show that for realistic inputs one can do much better. In that vein, we study convex hulls that are robust. A robust convex hull has the property that each vertex is far from the convex hull of the remaining vertices.

Definition 3. *We say $conv(S)$ is Γ_*-robust if* $\Gamma_* = \min\{d(\overline{v}_i, conv(\overline{S} \setminus \{\overline{v}_i\})) : i = 1, \ldots, K\}$.

Γ_* robustness was studied in the context of topic modeling in the work of [2]. The following is immediate from Definition 3. We now describe AVTA. Assume $conv(S)$ is Γ_*-robust and we are given $\gamma \leq \Gamma_*/R$. The following proposition that easily follows from the definition of robustness will play an important role in the way AVTA works.

Proposition 1. *Let $\widehat{S} = \{\widehat{v}_1, \ldots, \widehat{v}_N\}$ be a subset of the vertices of $conv(S)$. Suppose $conv(S)$ is Γ_*-robust. Given $v \in S \setminus \widehat{S}$, if for some $\gamma \leq \Gamma_*/R$ we have $d(v, conv(\widehat{S})) < \gamma R$, then $v \notin \overline{S}$.* □

Given a working subset $\widehat{S}$ of $\overline{S}$, initially of cardinality $N = 1$, i.e., a single vertex of S, AVTA randomly selects $v \in S \setminus \widehat{S}$. It then tests via the Triangle Algorithm if $d(v, conv(\widehat{S})) \leq \gamma R/2$. If so, it discards v since by definition of γ it cannot belong to $\overline{S}$ (Proposition 1). Otherwise, it computes a v-witness $p' \in conv(\widehat{S})$. It then sets $c' = v - p'$ and maximizes $c'^T x$ where x ranges in $conv(S \setminus \widehat{S})$. The maximum value coincides with the maximum of $c'^T v_i$ where v_i ranges in $S \setminus \widehat{S}$. If S' is the set of optimal solution in $S \setminus \widehat{S}$, it can be shown that $conv(S')$ is a face of $conv(S)$. Thus a vertex v' of $conv(S')$ is a point in S' and is necessarily a vertex of $conv(S)$. Such a vertex can be computed efficiently. Having computed a new vertex v' of $conv(S)$, AVTA replaces $\widehat{S}$ with $\widehat{S} \cup \{v'\}$ and the process is repeated. However, if v coincides with v', AVTA selects a new point in $S \setminus \widehat{S}$. Otherwise, AVTA continues to test if the same v (for which a witness was found) is within a distance of $\gamma R/2$ of the convex hull of the augmented set $\widehat{S}$. Also, as an iterate AVTA uses the same witness p'. In doing so, each selected $v \in S$ is either determined to be a vertex itself, or it will continue to be tested if it lies to within a distance of $\gamma R/2$ of the growing set $\widehat{S}$. If within $\gamma R/2$ distance, it will be discarded before AVTA tests another point. The detailed algorithm is presented below.

AVTA (S, $\gamma \in (0, 1)$)

- **Step 0.** Set $\widehat{S} = \{Farthest(v, S)\}$ for some $v \in S$.
- **Step 1.** Randomly select $v \in S \setminus \widehat{S}$.
- **Step 2.** Call **Triangle Algorithm** $(\widehat{S}, v, \gamma/2)$.
- **Step 3.** If the output p' of Step 1 is a v-witness then Goto Step 4. Otherwise, p' is a $\gamma/2$-approximate solution to v. Set $S \leftarrow S \setminus \{v\}$.
  If $S = \emptyset$, stop. Otherwise, Goto Step 1.
- **Step 4.** Let $c' = v - p'$.
  Compute S', the set of optimal solutions of $\max\{c'^T x : x \in S \setminus \widehat{S}\}$. Randomly select $v' \in S'$. $v' \leftarrow Farthest(v', S')$, $\widehat{S} \leftarrow \widehat{S} \cup \{v'\}$.
- **Step 5.** If $v = v'$, Goto Step 1. Otherwise, Goto Step 2.

We now present our main theorem regarding AVTA. Due to space constraints, all the proof are deferred to the supplementary material.

Theorem 2. *Suppose that $conv(S)$ is Γ_*-robust.*

(i) Given any $0 < \gamma \leq \Gamma_*/R$, the number of arithmetic operations of AVTA to compute $\overline{S}$ is $O\big(nK(m+\gamma^{-2})\big)$.

(ii) If K is known but no lower bound on Γ_*/R is known, the complexity of computing $\overline{S}$ via AVTA is $O\big(\big(nK(m+(R^2/\Gamma_*^2)\big)\log(R/\Gamma_*)\big)$.

(iii) More generally, given any $t \in (0,1)$ in $O\big(nK^{(t)}(m+t^{-2})\big)$ operations AVTA computes a subset $\widehat{S}$ of $\overline{S}$ of size $K^{(t)}$ so that the distance from each point in $conv(S)$ to $conv(\widehat{S})$ is at most tR. □

See Theorem 9 in the supplementary material for the proof.

3 Robustness of AVTA

In machine learning applications such as topic modeling one is often not given the actual set of points S but rather a perturbed set S_ε. This is because the documents in the corpus are limited in size and hence are only stochastic samples from an underlying distribution. Given such a perturbed set, the goal still is to recover good approximations to the original set of vertices $\overline{S}$. In this section we will show how to modify the AVTA algorithm to get a provably robust variant. For a given $\varepsilon \in (0,1)$ let $S_\varepsilon = \{v_1^\varepsilon, \ldots, v_n^\varepsilon\}$, and $\overline{S}_\varepsilon = \{\overline{v}_1^\varepsilon, \ldots, \overline{v}_K^\varepsilon\}$ be ε-perturbations of S and $\overline{S}$, respectively, where for each $i = 1, \ldots, n$, $\|v_i - v_i^\varepsilon\| \leq \varepsilon R$. Recall that R is the diameter of S. We will assume that we are given as input S_ε instead of S. The first question that arises is: What is the relationship between the vertices of S and those of S_ε? Without any assumptions, the vertices of $conv(S_\varepsilon)$ could change drastically even under small perturbations. For instance, consider a triangle with three additional interior points, very close to the vertices. It is very likely that even under small perturbations all six points become vertices, or that the interior points become the new vertices while the vertices become the new interior points. Thus there is a need to make some assumptions before we can say anything about the nature of perturbed points. Let $\overline{V}_\varepsilon$ be the set of vertices of S_ε. We would hope that for small ε, $\overline{S}_\varepsilon$ would be a subset of $\overline{V}_\varepsilon$. In this vein, we introduce the following definition.

Definition 4. *We say $conv(S)$ is Σ_*-weakly robust if the Euclidean distance from each vertex to the convex hull of the remaining points in S is at least Σ_*. More precisely, $\Sigma_* = \min\{d(v, conv(S \setminus \{v\})) : v \in \overline{S}\}$.*

Although Σ_* and Γ_* corresponding to the set S may seem unrelated, in the following theorem we establish a relationship between the two that is useful in the analysis of our algorithm.

Theorem 3. *Let S and $\overline{S}$ be as before, with R being the diameter of S. Suppose $conv(S)$ is Γ_*-robust. Let $\rho_* = \min\{d(v_i, v_j) : v_i, v_j \in S, i \neq j\}$. Then $conv(S)$ is Σ_*-weakly robust where $\Sigma_* \geq \frac{1}{R}\rho_*\Gamma_*$.*

See Theorem 11 in the supplementary material for the proof.

The following theorem describes a simple condition under which the set of vertices of $conv(S)$ under perturbation remain to be vertices of the perturbed convex hull.

Lemma 1. *Suppose $conv(S)$ is Σ_*-weakly robust. Suppose S_ε is an ε-perturbation of S and ε is such that $\varepsilon < \frac{\Sigma_*}{2R}$. If $v \in S$ is a vertex $conv(S)$ and $v^\varepsilon \in S_\varepsilon$ its corresponding ε-perturbation, then v^ε is a vertex of $conv(S_\varepsilon)$.*

See Theorem 12 in the supplementary material for the proof.

Let $\hat{S}_\epsilon$ be the output of AVTA when run on S_ε with $\gamma = \frac{\Sigma_*}{2R}$. From Lemma 1 we know that this set will contain the perturbed vertices of $conv(S)$. We next show that running AVTA followed by a pruning stage will recover only the perturbed vertices of $conv(S)$.

Pruning ($\hat{S}_\varepsilon$, $\sigma \in \mathbb{R}$)

- **Step 0.** Set $V = \emptyset$.
- **Step 1.** For each $v_i \in \hat{S}_\varepsilon$, use Triangle Algorithm [15] to check if $d(v_i, conv(\hat{S}_\varepsilon \setminus v_i)) > \sigma$. If yes, then $V = V \cup \{v_i\}$.
- **Step 2.** Output V.

Our next lemma proves that the pruning stage will indeed recover the perturbed vertices of $conv(S)$ and nothing else.

Lemma 2. *Suppose $conv(S)$ is Σ_*-weakly robust. Suppose $\varepsilon < \Sigma_*/(4R)$. Let $\hat{S}_\varepsilon$ be the output of $AVTA(S_\varepsilon, \frac{\Sigma_*}{4R})$. Then the output of the pruning stage with inputs $\hat{S}_\epsilon$ and $\sigma = \Sigma_*/(8R)$ is exactly $\overline{S}_\varepsilon$.*

See Lemma 2 in the supplementary material for the proof. We now state the main result for this section.

Theorem 4. *Let $S = \{v_1, \ldots, v_n\} \subset \mathbb{R}^m$. Assume $conv(S)$ is Σ_*-weakly robust. Suppose $\varepsilon \leq \Sigma_*/(4R)$.*

(i) AVTA can be modified to compute a subset $\widehat{S}_\varepsilon$ of the set of vertices of S_ε containing $\overline{S}_\varepsilon$, then compute from this subset $\overline{S}_\varepsilon$ itself. If K_ε is the cardinality of $\widehat{S}_\varepsilon$ the total number of operations satisfies, $O\left(nK_\varepsilon(m+\frac{R^2}{\Sigma_*^2})\right)$.

(ii) Given only K, AVTA can be modified to compute $\overline{S}_\varepsilon$ with total number of operations bounded by $O(nK_\varepsilon(m+(\frac{R}{\Sigma_*})^2))\log(R/\Sigma_*)$.

(iii) More generally, given any $t \in (0,1)$, AVTA can be modified to compute a subset $\overline{S}_\varepsilon^t$ of the set of vertices of

$conv(\overline{S}^t_\varepsilon)$ of cardinality $K^{(t)}_\varepsilon$ so that the distance from each point in $conv(S_\varepsilon)$ to $conv(\overline{S}^t_\varepsilon)$ is at most t. In particular, the distance from each point in $conv(S)$ to $conv(S^t_\varepsilon)$ is at most $(t+\varepsilon)R$. The complexity of the computation is $O\left(nmK^{(t)}_\varepsilon + \frac{nK^{(t)}_\varepsilon}{t^2}\right)$.

See Theorem 13 in the supplementary material for the proof.

A practical implementation: While the modified AVTA algorithm comes with theoretical guarantees, in certain cases the algorithm might output many more vertices, K_ε, than desired. Here we present a practical implementation that always outputs exactly K vertices, provided K is known. When K is unknown, our experiments in the next section reveal that the algorithm can automatically detect a slightly larger set that contains a good approximation to the K vertices of interest. Notice that we want a fast way to detect good approximations to the original vertices of the set S and prune out spurious points, i.e., additional vertices of the set S_ε. The key insight on top of the AVTA algorithm is the following: *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.* Using this insight the algorithm outlined below runs the modified AVTA algorithm over several random projections and outputs the set of points that appear as vertices in many random projections.

AVTA with multiple random projections ($S = \{v_1, \ldots, v_n\}$, K, γ, M)

- **Step 0.** Set $Freq \leftarrow 0^{|S|}$.
- **Step 1.** For $i = 1$ to M:
  - $S' \leftarrow S$: Project data on to randomly chosen $\frac{4log(n)}{\epsilon^2}$ dimensions.
  - $\hat{S} \leftarrow$ AVTA(S', γ)
  - For each $d_j \in \hat{S}$, $Freq[j] = Freq[j] + 1$.
- **Step 2.** Output top K frequent vertices.

4 Applications

We now show how AVTA can be used to solve the topic modeling in a variety of ways. We first look at how to solve the problem under the separability assumption.

AVTA in the presence of anchor words: Arora et al. [2] provide a practical algorithm for topic modeling with provable guarantees. Their algorithm works under the assumptions that the topic-word matrix is *separable*. In particular, they assume that corresponding to each topic i, there exists an *anchor word* w_i that has a non zero probability of appearing only under topic i. Under this assumption, the algorithm of [2] consists of two stages: a) find the anchor words, and b) use the anchor words to learn the topic word matrix. The problem of finding anchor words corresponds to finding the vertices of the convex hull of the word-word covariance matrix. They propose an algorithm named *fast anchor words* in order to find the vertices. Since AVTA works in general setting, we can instead use AVTA to find the anchor words. Additionally, the fast anchor words algorithm needs to know the value of the number of anchor words, as an input. On the other hand, from the statements of Theorems 2 and4 it is easy to see that AVTA can work in a variety of settings when other properties of the data are known such as the robustness. We argue that robustness is a parameter that can be tuned in a better manner than trying different values of the number of anchor words. In fact, one can artificially add random noise to the data and make it robust up to certain value. One can then run AVTA with the lower bound on robustness as input and let the algorithm automatically discover the number of anchor words. This is much more desirable in practical settings. Our first implementation of AVTA is named AVTA+RecoverL2 that uses AVTA to detect anchor words and then uses the anchor words to learn the topic word matrix using the approach from [2]. AVTA is also theoretically superior than fast anchor words and achieves slightly better run times in the regime when the number of topics is $o(\log n)$, where n is the number of words in the vocabulary. This is usually the case in most practical scenarios.

AVTA in the absence of anchor words: The presence of anchor words is a strong assumption that often does not hold in practice. Recently, the work of Bansal et al. [5] designed a new practical algorithm for topic models under the presence of catch words. Catch words for topic i correspond to set S_i such that it's total probability of appearing uunder topic i is significantly higher than in any other topic. Their algorithm called TSVD recovers much better reconstruction of the topic-word matrix in terms of the ℓ_1 error. They also assume that for each topic i, there are a few dominant documents that mostly contain words from topic i. The TSVD algorithm works in two stages. In stage 1, the (thresholded) word-document data matrix is projected onto a K-SVD space to compute a different embedding of the documents. Then, the documents are clustered into K clusters. Under the assumptions mentioned above, one can show that the dominant documents for each topic will be clustered correctly. In stage 2, a simple post processing algorithm can approximate the topic-word matrix from the clustering.

We improve on TSVD by asking the following question: *is K-SVD the right representation of the data?*. Our

key insight is that if dominant documents are present in the topic, it is easy to show that most other documents will be approximated by a convex combination of the dominant topics. Furthermore, the coefficients in the convex combinations will provide a much more faithful low dimensional embedding of the data. Using this insight, we propose a new algorithm that runs AVTA on the data matrix to detect vertices and to approximate each point using a convex combination of the vertices. We then use the coefficient matrix as the new representation of the data that needs to be clustered. Once the clustering is obtained, the same post processing step from [5] can be used to recover the topic-word matrix. Our results show that the embedding produced by AVTA leads to much better reconstruction error than of that produced by TSVD. Furthermore, K-SVD is an expensive procedure and very sensitive to the presence of outliers in the data. In contrast, our new algorithm called AVTA+CatchWord is much more stable to noise in the data.

AVTA+CatchWord ($S = \{v_1, \ldots, v_n\}$, γ, K, ε)

- **Step 0.** Randomly project S onto $2K$ dimensions to get $\tilde{S}$.
- **Step 1.** Compute a a super set of vertices $\tilde{V}$ by $AVTA(\tilde{S}, \gamma)$.
- **Step 2.** Prune $\tilde{V}$ into $\hat{V}$ (of size K) by iteratively picking $\tilde{v} \in \{\tilde{V}\}/\{\hat{V}\}$ which has the maximum distance to $conv(\hat{V})$.
- **Step 3.** For each projected point $\hat{v}_i \in \tilde{S} \setminus \hat{V}$, compute a vector α_i such that $\|\hat{V}\alpha_i - \hat{v}_i\| \leq \varepsilon$.
- **Step 4.** Initialize cluster assignment for each point by majority weight: $\underset{j \in [K]}{\operatorname{argmax}}\, \alpha_j$.
- **Step 5.** Clustering using Lloyds algorithm on the embedding provided by the α vectors.
- **Step 6.** Use the post processing as described in [5] to recover the topic-word matrix from the clustering.

AVTA for NMF: The work of [3] showed that convex hull detection can be used to solve the non-negative matrix factorization problem under the separability assumption. We show that by using the more general AVTA algorithm for solving the convex hull problem results in comparable performance guarantee.

5 Experiments [1]

We compare our algorithms with the Fast Anchor + Recoverl2 algorithm of [2] and the TSVD algorithm of [5] on two types of data sets: semi-synthetic data and real world data. We next describe our methodology and empirical results in detail.

Semi-Synthetic Data: For Semi-Synthetic data set, we use similar methodology as in [2]. We first train the model on real data set using Gibbs sampling with 1000 iterations. We choose 50 as the number of topics. Given the parameters learned from dataset, we generate documents with α set to be 0.01. The average document length is 1000. Then the reconstruction error is measured by the l_1 distance of bipartite matched pairs between the true word-topic distribution and the word-topic distribution [2]. We then average the errors to compute the final mean error.

Real Data: We use the **NIPS** data set with 1500 documents , and a pruned vocabulary of 2K words, and the NYTimes Corpus with sub sampled 30000 documents, and a pruned vocabulary of 5k words. [2]. For our experiments on NMF we use the Swimmer data set [13] that consists of 256 swimmer figures with each a 32×32 binary pixel image. One can interpret each image as a document and pixels as a word in the document [12]. All swimmers consist of 4 limbs with each limb having 4 different possible poses. One can then consider the different poses of limbs as the true underlying topics [13]. We compare the algorithm proposed in [3] with AVTA+Recoverl2 on the swimmer data set. Since the true underlying topics are known, we will plot the output of the algorithms and compare it with the underlying truth.

Implementation Details: We compare 4 algorithms, AVTA+CatchWord, TSVD, the Fast Anchor + Recoverl2 and the AVTA+Recoverl2. We implement our own version of Fast Anchor + Recoverl2 as described in [2]. TSVD is implemented using the code provided by the authors in [5]. AVTA+Recoverl2 corresponds to using AVTA to detect anchor words from the word-word covariance matrix and then using the Recoverl2 procedure from [2] to get the topic-word matrix. AVTA + CatchWord corresponds to finding the low dimensional embedding of each document in terms of the coefficient vector of its representation in the convex hull of the vertices. The next step is to cluster these points. In practice, one could use the Lloyd's algorithm for this step which could be sensitive to initialization. To remedy this, we use similar heuristic as [5] of the initialization step. We repeat AVTA for 3 times and pick the set of vertices with maximum sum of distance between each vertex and convex hull of remaining vertices. We set the number of output vertices $K = 50$ which is the same as the number of topics, i.e. each vertex corresponds to a topic. The precision parameter ε is set to 0.01. We found that initializing by simply as-

[1] Resources: `https://github.com/yikaizhang/AVTA`

[2] `https://archive.ics.uci.edu/ml/datasets/bag+of+words`

signing clusters using neighborhoods of highest degree vertices works effectively. As a final step, we use the post processing step from [5] to recover the topic-word matrix from the clustering.

Robustness: We also generate perturbed version of the semi-synthetic data. We generate a random matrix with i.i.d. entries uniformly distributed with different scales varying from $0.005 - 0.05$. We test all the algorithms with the document-word matrix added with the noise matrix.

Results on Semi-Synthetic Data: Figures 1 and 2 show the ℓ_1 reconstruction of all the four algorithms under both clean and noisy versions of the semi-synthetic data set. For topic i, let A_i be the ground truth topic vector and $\hat{A}_i$ be the topic vector recovered by the algorithm. Then the ℓ_1 error is defined as $\frac{1}{K}\sum_{i=1}^{K}\|A_i - \hat{A}_i\|_1$. The plots show that AVTA+CatchWord is consistently better than both TSVD and Fast Anchor + Recoverl2 and produces significantly more accurate topic vectors. In order to further test the robustness of out approach, we plot in Figure 3 the range of the ℓ_1 error obtained across multiple runs of the algorithms on the same data set. The range is defined to be the difference between the maximum and the minimum error recovered by the algorithm across different runs. We see that AVTA+CatchWord produces solutions that are much more stable to the effect of the noise as compared to other algorithms.

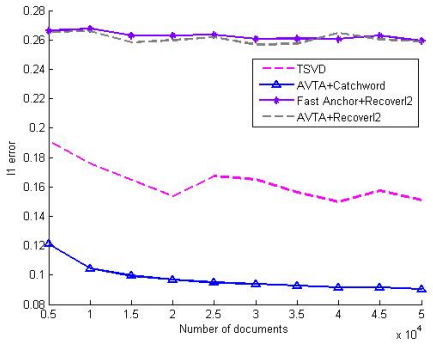

Figure 1: ℓ_1 error in the semi-synthetic dataset ($K = 50$).

Results on Real Data: For the real world data set, as in prior works [2, 5], we evaluate the coherence to measure topic quality [22]. Given a set of words $\mathcal{W}$ associated with a learned topic, the coherence is computed as: $Coherence(\mathcal{W}) = \sum_{w_1,w_2\in\mathcal{W}} \log \frac{D(w_1,w_2)+const}{D(w_2)}$, where $D(w_1)$ and $D(w_1, w_2)$ are the number of documents where w_1 appears and (w_1, w_2) appear together respectively [2], and small constant *const* is set to 0.01 to avoid w_1, w_2 that never co-occur [19]. The total coherence is the sum of the coherence of each topic. In the NIPS dataset, 1000 out of the 1500 documents were selected as the training set to learn the word-topic distributions. The rest of the documents were used as the testing set. Tables 1 shows the topic coherence obtained by the algorithms. One can see that in both the approaches, either via anchor words or the clustering approach, AVTA based algorithms perform comparably to state of the art methods [3].

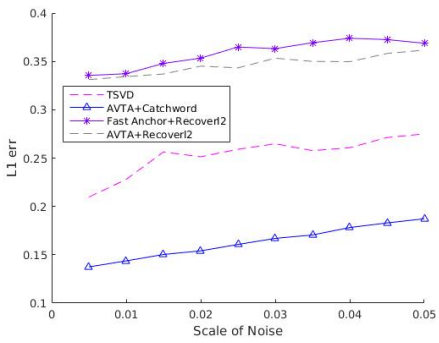

Figure 2: ℓ_1 error in the perturbed semi-synthetic dataset ($K = 50$).

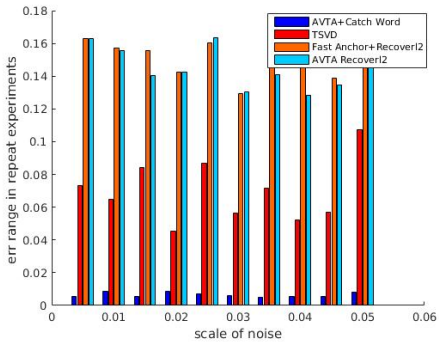

Figure 3: Range of the ℓ_1 error over 10 runs on the same data set.

Table 1: Topic coherence on real data

	FastAnchor+ RecoverL2	*AVTA + RecoverL2*	TSVD	*AVTA + CatchWord*
NIPS	-15.8 ±2.24	-16.04 ±2.09	-16.86 ±1.66	-18.65 ±1.78
NYTimes	-32.15 ±2.7	-32.13 ±2.43	-29.39 ±1.43	-30.13 ±1.98

Running time of experiments:

Table 2 and Table 3 shows the running time for algorithms to learn the model from semi-synthetic data

[3]The topic coherence results for TSVD do not match the ones presented in [5] since in their experiments, the authors look at top 10 most frequent words in each topic. In our experiments we compute coherence for the top 5 most frequent words in each topic.

and real data. As can be seen, when using AVTA to learn topic models via the anchor words approach, our algorithm has comparable run time to Fast Anchor + Recoverl2. Compared to clustering approach, the AVTA+CathchWord has less running time in the real data experiments. Per our observation, the convex hull of word-document vectors in real data set has more vertices than K, the number of topics. The AVTA catches K vertices efficiently due to its small number of iterations on line search for γ. In semi-synthetic data set, the number of 'robust' vertices is approximately the same as number of topics K thus AVTA needs to find almost all vertices. To catch enough vertices, AVTA needs several iterations decreasing γ which is computationally expensive.

Table 2: Running time on semi-synthetic data (secs)

# of documents	*FastAnchor+ RecoverL2*	*AVTA + RecoverL2*	TSVD	*AVTA + CatchWord*
5000	5.49	7.82	17.02	29.89
15000	6.00	7.68	43.27	120.04
30000	10.30	12.84	81.24	372.17
50000	13.60	16.40	112.80	864.30

Table 3: Running time on real data experiments (secs)

	FastAnchor+ RecoverL2	*AVTA + RecoverL2*	TSVD	*AVTA + CatchWord*
NIPS	3.22	4.41	56.58	22.78
NYTimes	26.05	27.79	237.6	101.07

Results on NMF: The swimmer data set [13] consists of 256, 32×32 images and the task consists of inferring the underlying "poses" in a given image. In our experiments we use the original swimmer data set and also construct a noisy version by adding spurious poses. Let $\Omega(A)$ be a function that outputs a randomly chosen 32×8 block of an image. We generate a 'spurious pose' of size 32×8 by $\Omega(M_i)$ where M_i is a randomly chosen swimmer image. Then we take another randomly chosen image M_j and compute the corrupted image as $M'_j = M_j + c \cdot \Omega(M_i)$ where we simply set $c = 0.1$. An illustration of the noise data set is shown in Figure 4. We compare the performance of AVTA on these data sets with the performance of the Separable NMF algorithm proposed in [3]. Figures 5 and 6 show the output of the Separable NMF algorithm and that of our algorithm respectively on the noisy data set. Our approach produces competitive results as compared to the Separable NMF algorithm.

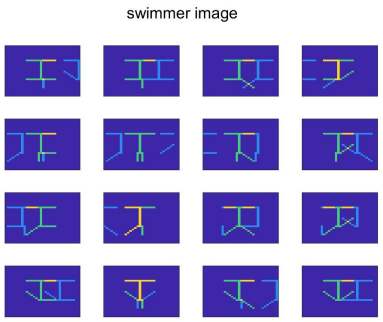

Figure 4: An example of spurious actions in swimmer images.

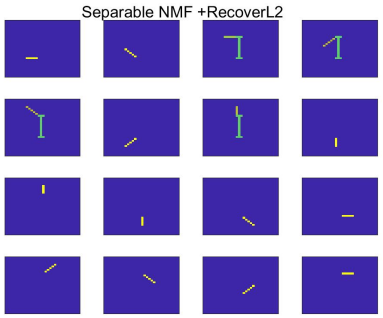

Figure 5: Output of NMF +RecoverL2

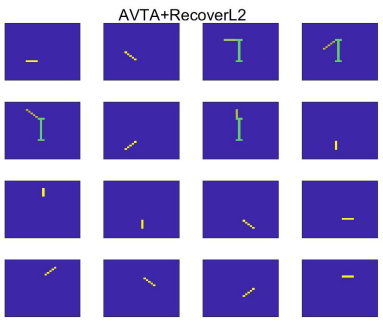

Figure 6: Output of AVTA +RecoverL2

6 Conclusion

In this work we have presented a fast and robust algorithm for computing the vertices of the convex hull of a set of points. Our algorithm efficiently computes the vertices of convex hulls in high dimensions and even in the special case of the simplex is competitive with the state of the art approaches in terms of runing time [2]. Furthermore, our algorithm leads to an improved algorithm for topic modeling that is more robust and produces better approximations to the topic-word matrix. It will be interesting to provide theoretical claims supporting this observation in the context of specific applications. Furthermore, we believe that our algorithm will have more applications in machine learning problems beyond the ones investigated here as well as applications in computational geometry and in linear programming.

Acknowledgement

We thank anonymous reviewers for helpful comments.

References

[1] Anima Anandkumar, Dean P Foster, Daniel J Hsu, Sham M Kakade, and Yi-Kai Liu. A spectral algorithm for latent dirichlet allocation. In *Advances in Neural Information Processing Systems*, pages 917–925, 2012.

[2] Sanjeev Arora, Rong Ge, Yonatan Halpern, David Mimno, Ankur Moitra, David Sontag, Yichen Wu, and Michael Zhu. A practical algorithm for topic modeling with provable guarantees. In *International Conference on Machine Learning*, pages 280–288, 2013.

[3] Sanjeev Arora, Rong Ge, Ravindran Kannan, and Ankur Moitra. Computing a nonnegative matrix factorization–provably. In *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*, pages 145–162. ACM, 2012.

[4] Sanjeev Arora, Rong Ge, and Ankur Moitra. Learning topic models–going beyond svd. In *Foundations of Computer Science (FOCS), 2012 IEEE 53rd Annual Symposium on*, pages 1–10. IEEE, 2012.

[5] Trapit Bansal, Chiranjib Bhattacharyya, and Ravindran Kannan. A provable svd-based algorithm for learning topics in dominant admixture corpus. In *Advances in Neural Information Processing Systems*, pages 1997–2005, 2014.

[6] David M Blei. Probabilistic topic models. *Communications of the ACM*, 55(4):77–84, 2012.

[7] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.

[8] Timothy M Chan. Optimal output-sensitive convex hull algorithms in two and three dimensions. *Discrete & Computational Geometry*, 16(4):361–368, 1996.

[9] Timothy M Chan. Output-sensitive results on convex hulls, extreme points, and related problems. *Discrete & Computational Geometry*, 16(4):369–387, 1996.

[10] Bernard Chazelle. An optimal convex hull algorithm in any fixed dimension. *Discrete & Computational Geometry*, 10(1):377–409, 1993.

[11] Kenneth L Clarkson. More output-sensitive geometric algorithms. In *Foundations of Computer Science, 1994 Proceedings., 35th Annual Symposium on*, pages 695–702. IEEE, 1994.

[12] Weicong Ding, Mohammad Hossein Rohban, Prakash Ishwar, and Venkatesh Saligrama. Topic discovery through data dependent and random projections. In *ICML (3)*, pages 1202–1210, 2013.

[13] David Donoho and Victoria Stodden. When does non-negative matrix factorization give a correct decomposition into parts? In *Advances in Neural Information Processing Systems*, 2003.

[14] Ray A Jarvis. On the identification of the convex hull of a finite set of points in the plane. *Information Processing Letters*, 2(1):18–21, 1973.

[15] Bahman Kalantari. A characterization theorem and an algorithm for a convex hull problem. *Annals of Operations Research*, 226(1):301–349, 2015.

[16] Daniel D Lee and H Sebastian Seung. Algorithms for non-negative matrix factorization. In *Advances in neural information processing systems*, pages 556–562, 2001.

[17] Jiří Matoušek and Otfried Schwarzkopf. Linear optimization queries. In *Proceedings of the eighth annual symposium on Computational geometry*, pages 16–25. ACM, 1992.

[18] Christos H Papadimitriou, Hisao Tamaki, Prabhakar Raghavan, and Santosh Vempala. Latent semantic indexing: A probabilistic analysis. In *Proceedings of the seventeenth ACM SIGACT-SIGMOD-SIGART symposium on Principles of database systems*, pages 159–168. ACM, 1998.

[19] Keith Stevens, Philip Kegelmeyer, David Andrzejewski, and David Buttler. Exploring topic coherence over many models and many topics. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 952–961. Association for Computational Linguistics, 2012.

[20] Csaba D Toth, Joseph O'Rourke, and Jacob E Goodman. *Handbook of discrete and computational geometry*. CRC press, 2004.

[21] Godfried T Toussaint. Solving geometric problems with the rotating calipers. In *Proc. IEEE Melecon*, volume 83, page A10, 1983.

[22] Limin Yao, David Mimno, and Andrew McCallum. Efficient methods for topic model inference on

streaming document collections. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 937–946. ACM, 2009.