ECE 598: Representation Learning: Algorithms and Models Fall 2017

Lecture 1: Tensor Methods in Machine Learning

Lecturer: Pramod Viswanathan *Scribe: Bharath V. Raghavan, Oct. 3, 2017*

1.1 Introduction

Tensors are higher-order generalizations of matrices are often used to represent multi-linear relationships or data that involves higher order correlation. They are useful tools in parameter estimation in learning problems. A tensor can be visualized as shown in Fig. 1.1 Tensor decomposition have gained popularity in parameter estimation for a variety of problems. In this lecture, the focus is on how they may be used in estimating the parameters of Gaussian Mixture Models and Hidden Markov Models. In a Gaussian mixture model, there are k unknown n-dimensional multivariate Gaussian distributions. Samples are generated by first picking one of the k Gaussians, then drawing a sample from that Gaussian distribution. Given samples from the mixture distribution, our goal is to estimate the means and covariance matrices of these underlying Gaussian distributions [GHK15]. That is, given $\boldsymbol{x} = (x_1, x_2, ..., x_N)$

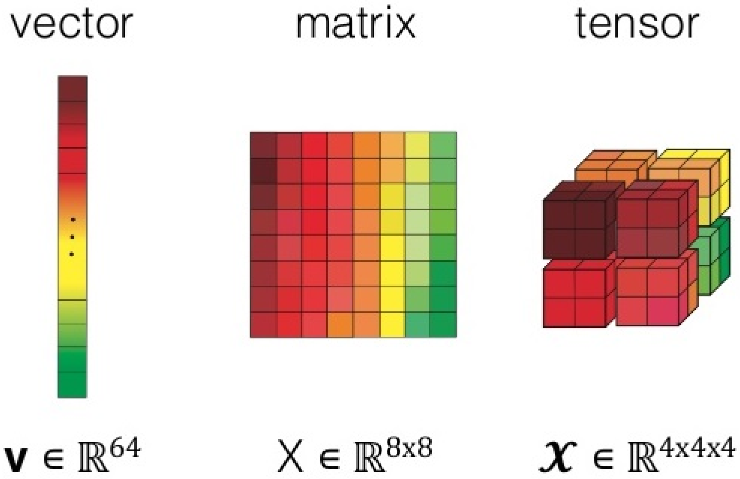
vector
matrix
tensor
$\mathbf{v} \in \mathbb{R}^{64}$
$X \in \mathbb{R}^{8x8}$
$\mathcal{X} \in \mathbb{R}^{4x4x4}$

Figure 1.1: Visualization of Tensors of different orders.

$$F_X(\boldsymbol{x}) = \sum_{l=1}^{r} w_l \mathcal{N}(\boldsymbol{x}_l; \boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l) \tag{1.1}$$

The goal is to learn the parameters of this mixture model. Alternatively, can a tensor $\boldsymbol{\mathcal{T}}$ which represents the data originating from a statistical model, be expressed as

$$\boldsymbol{\mathcal{T}} = \sum_{l=1}^{r} \boldsymbol{u}_l \otimes \boldsymbol{v}_l \otimes \boldsymbol{w}_l \tag{1.2}$$

where $\boldsymbol{u}_l$, $\boldsymbol{v}_l$, and $\boldsymbol{w}_l$ are the factors of the tensor representing the parameters of the model.

1.1.1 Comparison to Matrix Decomposition

For a matrix $\boldsymbol{M}$, Singular Value Decomposition allows it to be expressed as

$$\boldsymbol{M} = \sum_{l=1}^{r} \lambda_l \boldsymbol{u}_l \boldsymbol{v}_l^T \tag{1.3}$$

Tensor decomposition is a generalization of Matrix decomposition. However, if it exists it is unique. Finding a decomposition for a tensor is a computationally difficult task. Computing the rank, the best rank one approximation and the spectral norm are all NP-hard [HL13]. Also many of the familiar properties of matrices do not generalize to tensors. For example, subtracting the best rank one approximation to a tensor can actually increase its rank and there are rank three tensors that can be approximated arbitrarily well by a sequence of rank two tensors [BCMV14].

1.2 Motivating Examples

To motivate the need for Tensor Methods we first study *Spearman's Hypothesis* and representing the transition matrix *Hidden Markov Models.*

1.2.1 Spearman's Hypothesis

In 1904, psychologist Charles Spearman tried to understand whether human intelligence is a composite of different types of measureable intelligence. A highly simplified version of his method, hypothesizes that there are exactly two kinds of intelligence: *quantitative* and *verbal.* Spearmans method consists of making his subjects take several different kinds of tests. For instance, Classics, Math, Music, etc. The subjects scores can be represented by a matrix $\boldsymbol{M}$, which has one row per student, and one column per test. Table 1.1 illustrates the data,

	Classics	Math	Music	...
Alice	19	26	17	...
Bob	8	17	9	...
Charlie	7	12	7	...
⋮	⋮	⋮	⋮	

Table 1.1: Spearman's Hypothesis

One way to represent the data is with $\boldsymbol{x}_q$, $\boldsymbol{y}_q$, $\boldsymbol{x}_v$, and $\boldsymbol{y}_v$ as the quantitative and verbal intelligence of the students ($\boldsymbol{x}$) and the required intelligence for each subject ($\boldsymbol{y}$). Then, the data is

$$\boldsymbol{M} = \boldsymbol{x}_q \boldsymbol{y}_q^T + \boldsymbol{x}_v \boldsymbol{y}_v^T \tag{1.4}$$

It can be recognized that this matrix model is not unique. For instance, two plausible representations are

$$\begin{bmatrix} \boldsymbol{x}_q & \boldsymbol{x}_v \end{bmatrix} = \begin{bmatrix} 4 & 3 \\ 3 & 1 \\ 2 & 1 \end{bmatrix}$$

and

$$\begin{bmatrix} \boldsymbol{y}_q \\ \boldsymbol{y}_v \end{bmatrix} = \begin{bmatrix} 1 & 5 & 2 \\ 5 & 2 & 3 \end{bmatrix}$$

Or,

$$\begin{bmatrix} \boldsymbol{x}_q & \boldsymbol{x}_v \end{bmatrix} = \begin{bmatrix} 1 & 3 \\ 2 & 1 \\ 1 & 1 \end{bmatrix}$$

and

$$\begin{bmatrix} \boldsymbol{y}_q \\ \boldsymbol{y}_v \end{bmatrix} = \begin{bmatrix} 1 & 5 & 2 \\ 6 & 7 & 1 \end{bmatrix}$$

Hence, the low-rank structure is not unique. Whereas the first decomposition shows Alice is strongest in quantitative intelligence, the second indicates Bob as having the strongest quantitative intelligence, not Alice. To circumvent this ambiguity, one may add a third dimension to the data. This could be the time of day (i.e. Day or Night) represented by a binary variable $\boldsymbol{z}$. For example, the scores in Table 1.1 are those during the day, and the nightly scores are given in Table 1.2

	Classics	Math	Music	...
Alice	23	46	25	...
Bob	11	32	15	...
Charlie	9	22	11	...
⋮	⋮	⋮	⋮	

Table 1.2: Spearman's Hypothesis Night Scores

Now, the data in Table 1.1 and Table 1.2may be represented as tensor with the following form

$$\boldsymbol{\mathcal{T}} = \boldsymbol{x}_q \otimes \boldsymbol{y}_q \otimes \boldsymbol{z}_q + \boldsymbol{x}_v \otimes \boldsymbol{y}_v \otimes \boldsymbol{z}_v \tag{1.5}$$

The main motivation for the use of tensor methods is that under certain conditions, tensors have a unique low-rank decomposition, even though the corresponding matrix model does not. Now, the decomposition is

$$\begin{bmatrix} \boldsymbol{x}_q & \boldsymbol{x}_v \end{bmatrix} = \begin{bmatrix} 4 & 3 \\ 3 & 1 \\ 2 & 1 \\ \vdots & \vdots \end{bmatrix}$$

and

$$\begin{bmatrix} \boldsymbol{y}_q & \boldsymbol{y}_v \end{bmatrix} = \begin{bmatrix} 1 & 5 \\ 5 & 2 \\ 2 & 3 \\ \vdots & \vdots \end{bmatrix}$$

and

$$\begin{bmatrix} \boldsymbol{z}_q & \boldsymbol{z}_v \end{bmatrix} = \begin{bmatrix} 1 & 3 \\ 2 & 1 \\ \vdots & \vdots \end{bmatrix}$$

It can be observed that the second matrix decomposition that indicated Bob with the highest quantitative intelligence is no longer valid. There are no values of z_q and z_v at night that could generate the matrix Table 1.2.

1.2.2 Transition Matrix of a Hidden Markov Model

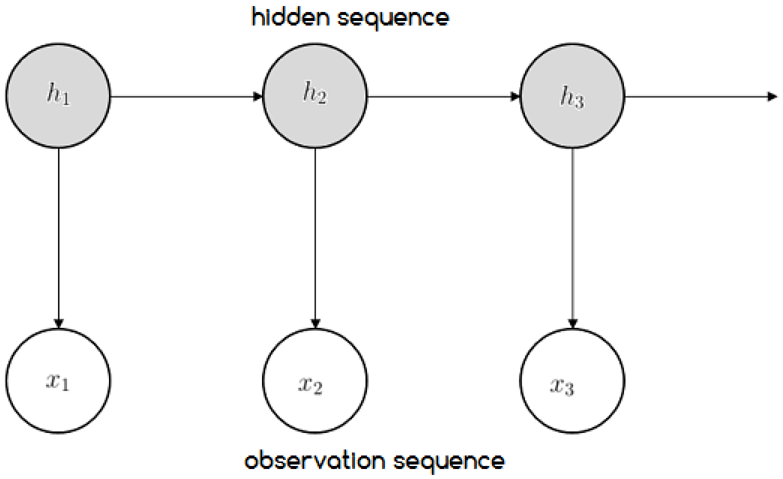

Figure 1.2: Hidden Markov Model with observation sequence $\boldsymbol{x}$ and hidden sequence $\boldsymbol{h}$.

Consider a time series model in which the observations (such as a sequence of words) is generated by an underlying Markov process (such as the subject topic). The natural idea is to compute correlations to detect patterns. For instance, if (i, j, k) represent a sequence of words, we count the number of times that these are the first three words of a sentence. Enumerating over i, j, and k gives us a three dimensional array (a tensor) $\boldsymbol{\mathcal{T}} = \mathcal{T}_{ijk}$. We can further normalize it by the total number of sentences. After normalization the (i, j, k)-th entry of the tensor will be an estimation of the probability that the first three words are (i, j, k). For simplicity assume we have enough samples and the estimation is accurate. Then

$$\mathcal{T}_{ijk} = P(x_1 = i, x_2 = j, x_3 = k) \tag{1.6}$$

Now, if we condition on h_2 we obtain

$$T_{ijk} = \sum_{l=1}^{n} P(h_2 = l)P(x_1 = i|h_2 = l)P(h_2 = l)P(x_2 = j|h_2 = l)P(h_2 = l)P(x_3 = j|h_2 = l) \tag{1.7}$$

By letting $\boldsymbol{x}_l$ be a vector whose i-th entry is the probability of the first word is i, given the topic of the second word is l, and similarly $\boldsymbol{y}_l$ and $\boldsymbol{z}_l$, we have

$$\mathcal{T}_{ijk} = \sum_{l=1}^{n} P(h_2 = l)\boldsymbol{x}_l \otimes \boldsymbol{y}_l \otimes \boldsymbol{z}_l \tag{1.8}$$

which it the tensor form that we require to learn the parameters from.

1.3 Parameter learning in Gaussian Mixtures

1.3.1 Problem Setup

Formally, the learning problem is given the data $\boldsymbol{x} \subset \mathbb{R}^d$ is drawn from a mixture of k Gaussians with $d \geq k$. So

$$\boldsymbol{x} \sim \sum_{i=1}^{k} w_i \mathcal{N}(\boldsymbol{\mu}_i, \sigma_i^2 \boldsymbol{I}) \tag{1.9}$$

The goal is learn the parameters w_i, $\boldsymbol{\mu}_i$, and σ_i^2. The setting above corresponds to the mixture of spherical Gaussians studied in Ref. [AGH+14] with differing covariances. The non-degeneracy condition is that the vectors $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, ..., \boldsymbol{\mu}_k \in \mathbb{R}^d$ are linearly independent, and the scalars $w_1, w_2, ..., w_k > 0$ are strictly positive [AGH+14]. The moments are given as

$$E[\boldsymbol{x}] = \sum_{i=1}^{k} w_i \boldsymbol{\mu}_i \tag{1.10}$$

$$E[\boldsymbol{x} \otimes \boldsymbol{x}] = \sum_{i=1}^{k} w_i \boldsymbol{\mu}_i \otimes \boldsymbol{\mu}_i + \bar{\sigma}^2 \boldsymbol{I} \tag{1.11}$$

$$E[\boldsymbol{x} \otimes \boldsymbol{x} \otimes \boldsymbol{x}] = \sum_{i=1}^{k} w_i \boldsymbol{\mu}_i \otimes \boldsymbol{\mu}_i \otimes \boldsymbol{\mu}_i + \sum_{i=1}^{k} \boldsymbol{m}_1 \otimes \boldsymbol{e}_i \otimes \boldsymbol{e}_i + \boldsymbol{e}_i \otimes \boldsymbol{m}_1 \otimes \boldsymbol{e}_i + \boldsymbol{e}_i \otimes \boldsymbol{e}_i \otimes \boldsymbol{m}_1 \tag{1.12}$$

where

$$\boldsymbol{m}_1 = E[\boldsymbol{x}(\boldsymbol{v}^T(\boldsymbol{x} - E[\boldsymbol{x}])^2] \tag{1.13}$$

and let $\boldsymbol{v}$ be any unit-norm eigenvector corresponding to $\bar{\sigma}^2$. The mean covariance is

$$\bar{\sigma}^2 = \sum_{i=1}^{k} w_i \sigma_i^2 \tag{1.14}$$

and it is the smallest eigenvalue of the covariance matrix $E[(\boldsymbol{x} - E[\boldsymbol{x}])(\boldsymbol{x} - E[\boldsymbol{x}])^T]$. Now, we have

$$\boldsymbol{m}_1 = \sum_{i=1}^{k} w_i \sigma_i^2 \tag{1.15}$$

$$\boldsymbol{M}_2 = \sum_{i=1}^{k} w_i \boldsymbol{\mu}_i \otimes \boldsymbol{\mu}_i \tag{1.16}$$

$$\boldsymbol{\mathcal{M}}_3 = \sum_{i=1}^{k} w_i \boldsymbol{\mu}_i \otimes \boldsymbol{\mu}_i \otimes \boldsymbol{\mu}_i \tag{1.17}$$

1.3.2 Parameter Estimation via Jenrich's Algorithm

The idea is to now use $\boldsymbol{M}_2$ and $\boldsymbol{\mathcal{M}}_3$, which are estimated from data, to estimate w_i and $\boldsymbol{\mu}_i$. The process requires using $\boldsymbol{M}_2$ and $\boldsymbol{\mathcal{M}}_3$. In particular, how may we recover w_i and $\boldsymbol{\mu}_i$ from $\boldsymbol{\mathcal{M}}_3$ and under what conditions. Revisiting Eq. 1.17, for any vector $\boldsymbol{v} \in \mathcal{R}^d$, we have

$$\boldsymbol{\mathcal{M}}_3(\boldsymbol{v}, \boldsymbol{I}, \boldsymbol{I}) = \sum_{i=1}^{k} w_i(\boldsymbol{v}^T \boldsymbol{\mu}_i) \otimes \boldsymbol{\mu}_i \otimes \boldsymbol{\mu}_i \tag{1.18}$$

Let Eq. 1.18 have a diagonal representation $\boldsymbol{U\Lambda U}^T$. If the $\boldsymbol{\mu}_i$'s are linearly independent (sufficient for Jenrich's inequality), and are orthonormal, then we can recover them as eigenvectors, and w_i's as solutions to the linear equation $\lambda_i = w_i(\boldsymbol{v}^T\boldsymbol{\mu}_i)$. Jenrich's algorithm proceeds as follows:

1. Let $\boldsymbol{\mathcal{T}} = \sum_{i=1}^{k} \boldsymbol{v}_i \otimes \boldsymbol{v}_i \otimes \boldsymbol{v}_i$ with $\boldsymbol{v}_i$ being linearly independent. This implies $k \leq d$.
2. For $\boldsymbol{x} \in \mathbb{R}^d$, we have $\boldsymbol{T}_x = \boldsymbol{\mathcal{T}}.\boldsymbol{x} = \boldsymbol{U}\boldsymbol{D}_x\boldsymbol{U}^T$, $\boldsymbol{U} = [\boldsymbol{v}_1, ..., \boldsymbol{v}_k] \in \mathbb{R}^{d\times k}$. $\boldsymbol{D}_x$ is the diagonal matrix with entries $\boldsymbol{v}_i^T\boldsymbol{x}$
3. Draw two vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ at random in $\mathbb{R}^d$. Then $\boldsymbol{T}_x(\boldsymbol{T}_y)^+ = \boldsymbol{U}\boldsymbol{D}_x(\boldsymbol{D}_y)^{-1}\boldsymbol{U}^+$. Drawing $\boldsymbol{x}$ and $\boldsymbol{y}$ at random ensures that $\boldsymbol{D}_y$ is invertible and diagonal entries of $\boldsymbol{D}_x(\boldsymbol{D}_y)^{-1}$ are distinct.
4. Since $\boldsymbol{U}$ has rank k, we have $\boldsymbol{U}^+\boldsymbol{U} = \boldsymbol{I}$. So, the $\boldsymbol{v}_i$'s can be recovered as eigenvectors of $\boldsymbol{T}_x(\boldsymbol{T}_y)^+$

In the algorithm, + denotes pseudo-inverse of a matrix. The algorithm looks at weighted slices of the tensor $\boldsymbol{\mathcal{T}}$: a weighted slice is a matrix that is the projection of the tensor along the $\boldsymbol{x}$ or $\boldsymbol{y}$ directions (similarly if we take a slice of a matrix $\boldsymbol{M}$, it will be a vector that is equal to $\boldsymbol{M}.\boldsymbol{x}$). Because of the low rank structure, all the slices must share matrix decompositions with the same components. The main observation of the algorithm is that although a single matrix can have infinitely many low rank decompositions, two matrices can only have a unique decomposition if we require them to have the same components. In fact, it is highly unlikely for two arbitrary matrices to share decompositions with the same components. In the tensor case, because of the low rank structure we have

$$\boldsymbol{T}_x = \boldsymbol{U}\boldsymbol{D}_x\boldsymbol{U}^T, \boldsymbol{T}_y = \boldsymbol{U}\boldsymbol{D}_y\boldsymbol{U}^T \tag{1.19}$$

Therefore,

$$\boldsymbol{T}_x(\boldsymbol{T}_y)^+ = \boldsymbol{U}\boldsymbol{D}_x(\boldsymbol{D}_y)^{-1}\boldsymbol{U}^+ \tag{1.20}$$

where $\boldsymbol{D}_x$,$\boldsymbol{D}_y$ are diagonal matrices. This is called a simultaneous diagonalization for $\boldsymbol{T}_x$ and $\boldsymbol{T}_y$. With this structure it is easy to show that $\boldsymbol{v}_i$'s are eigenvectors of $\boldsymbol{T}_x(\boldsymbol{T}_y)^+ = \boldsymbol{U}\boldsymbol{D}_x(\boldsymbol{D}_y)^{-1}\boldsymbol{U}^+$. So we can actually compute tensor decompositions using spectral decompositions for matrices.

1.3.3 Parameter Estimation via Simultaneous Orthogonal Tensor Diagonlization

Alternatively, a more robust method first uses $\boldsymbol{M}_2$ in Eq. 1.16 to whiten $\boldsymbol{\mathcal{M}}_3$ in Eq. 1.17. This method essentially, reduces the problem of solving

$$\begin{bmatrix} \boldsymbol{M}_2 & = & \sum_{i=1}^{k} w_i\boldsymbol{\mu}_i \otimes \boldsymbol{\mu}_i \\ \boldsymbol{\mathcal{M}}_3 & = & \sum_{i=1}^{k} w_i\boldsymbol{\mu}_i \otimes \boldsymbol{\mu}_i \otimes \boldsymbol{\mu}_i \end{bmatrix}$$

into finding an orthogonal decomposition for the tensor $\boldsymbol{\mathcal{T}}$, i.e. we find $\boldsymbol{W} \in \mathbb{R}^{k \times d}$ which is a linear transformation such that

$$\boldsymbol{M}_2(\boldsymbol{W}, \boldsymbol{W}) = \boldsymbol{W}^T \boldsymbol{M}_2 \boldsymbol{W} = \boldsymbol{I} \tag{1.21}$$

where $\boldsymbol{I}$ is the $k \times k$ identity matrix. Hence, $\boldsymbol{W}$ whitens $\boldsymbol{M}_2$. Since, $\boldsymbol{M}_2 = \boldsymbol{U}\boldsymbol{D}\boldsymbol{U}^T$, we can have $\boldsymbol{W} := \boldsymbol{U}\boldsymbol{D}^{-1/2}$, where $\boldsymbol{U} \in \mathbb{R}^{d \times k}$ is the matrix of orthonormal eigenvectors of $\boldsymbol{M}_2$, and $\boldsymbol{D} \in \mathbb{R}^{k \times k}$ is the diagonal matrix of positive eigenvalues of $\boldsymbol{M}_2$. Additionally, we define

$$\tilde{\boldsymbol{\mu}}_i := \sqrt{w_i} \boldsymbol{W}^T \boldsymbol{\mu}_i \tag{1.22}$$

Such that,

$$\boldsymbol{M}_2(\boldsymbol{W}, \boldsymbol{W}) = \sum_{i=1}^{k} \boldsymbol{W}^T (\sqrt{w_i} \boldsymbol{W}^T \boldsymbol{\mu}_i)(\sqrt{w_i} \boldsymbol{W}^T \boldsymbol{\mu}_i)^T \boldsymbol{W} = \sum_{i=1}^{k} \tilde{\boldsymbol{\mu}}_i \tilde{\boldsymbol{\mu}}_i = I \tag{1.23}$$

so $\tilde{\boldsymbol{\mu}}_i \in \mathbb{R}^k$ are orthonormal vectors.

Now, we define $\boldsymbol{\mathcal{T}} := \boldsymbol{\mathcal{M}}_3(\boldsymbol{W}, \boldsymbol{W}, \boldsymbol{W}) \in \mathbb{R}^{k \times k \times k}$, so that

$$\boldsymbol{\mathcal{T}} = \sum_{i=1}^{k} \frac{1}{\sqrt{w_i}} \tilde{\boldsymbol{\mu}}_i \otimes \tilde{\boldsymbol{\mu}}_i \otimes \tilde{\boldsymbol{\mu}}_i = \sum_{i=1}^{k} \tilde{w}_i \tilde{\boldsymbol{\mu}}_i \otimes \tilde{\boldsymbol{\mu}}_i \otimes \tilde{\boldsymbol{\mu}}_i \tag{1.24}$$

Since, $\boldsymbol{U}\boldsymbol{U}^T \boldsymbol{\mu}_i = \boldsymbol{\mu}_i \; \forall i$, $\boldsymbol{W}^{+} \tilde{\boldsymbol{\mu}}_i = \sqrt{w_i} \boldsymbol{\mu}_i$. Hence, w_i's and $\boldsymbol{\mu}_i$'s can be recovered as eigenvalue/eigenvectors for any vector $\boldsymbol{v}$. Proceeding with the decomposition, we want to find the orthogonal decomposition to Eq. 1.24. For any vector $\boldsymbol{v}$, we have

$$\boldsymbol{\mathcal{T}}.\boldsymbol{v} = \sum_{i=1}^{k} \tilde{w}_i (\boldsymbol{v}^T \tilde{\boldsymbol{\mu}}_i) \tilde{\boldsymbol{\mu}}_i \otimes \tilde{\boldsymbol{\mu}}_i = \boldsymbol{U} \boldsymbol{\Lambda} \boldsymbol{U}^T \tag{1.25}$$

with $\boldsymbol{U} = [\tilde{\boldsymbol{\mu}}_1, ..., \tilde{\boldsymbol{\mu}}_k]$ and the diagonals of the matrix $\boldsymbol{\Lambda}_{jj} = \tilde{w}_i(\boldsymbol{v}^T \tilde{\boldsymbol{\mu}}_i)$. The improve the robustness of the algorithm by reducing its sensitivity to noise, performing simultaneous diagonalization of several random projections is proposed [KCL15]. The final parameters are obtained by reverting the transformations for $\tilde{\boldsymbol{\mu}}_i$ and $\tilde{w}_i$.

1.4 Conclusion

This lecture focuses on tensor methods to estimate the parameters of mixture models. We demonstrated how this estimation is obtained via an orthogonal decomposition which is analogous to matrix decomposition. These algorithms usually only work when the dimension of each $\boldsymbol{\mu}_i$ is greater than the number of Gaussians - commonly referred to as the 'blessing' of dimensionality. The procedure is illustrated for the case of a mixture of Gaussians but can be used for other single-topic models and multiview models.

Additionally, there exist more robust algorithms that can compute these decompositions in polynomial time such as in Refs. [BCMV14], [GHK15], [GM15].

Bibliography

[AGH^{+}14] Animashree Anandkumar, Rong Ge, Daniel J Hsu, Sham M Kakade, and Matus Telgarsky. Tensor decompositions for learning latent variable models. *Journal of Machine Learning Research*, 15(1):2773–2832, 2014.

[BCMV14] Aditya Bhaskara, Moses Charikar, Ankur Moitra, and Aravindan Vijayaraghavan. Smoothed analysis of tensor decompositions. In *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*, pages 594–603. ACM, 2014.

[GHK15] Rong Ge, Qingqing Huang, and Sham M Kakade. Learning mixtures of gaussians in high dimensions. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, pages 761–770. ACM, 2015.

[GM15] Rong Ge and Tengyu Ma. Decomposing overcomplete 3rd order tensors using sum-of-squares algorithms. *arXiv preprint arXiv:1504.05287*, 2015.

[HL13] Christopher J Hillar and Lek-Heng Lim. Most tensor problems are np-hard. *Journal of the ACM (JACM)*, 60(6):45, 2013.

[KCL15] Volodymyr Kuleshov, Arun Chaganty, and Percy Liang. Tensor factorization via matrix factorization. In *Artificial Intelligence and Statistics*, pages 507–516, 2015.