Guaranteed Non-convex Machine Learning
Using Tensor Methods

Anima Anandkumar

U.C. Irvine
Regime of Modern Machine Learning
Massive datasets, growth in computation power, challenging tasks

Success of Supervised Learning

- Learn $p(y|x)$ from labeled samples $\{(x_i, y_i)\}$.
- Extract relevant features from large amounts of labeled data.

Image classification  Speech recognition  Text processing
Regime of Modern Machine Learning

Massive datasets, growth in computation power, challenging tasks

Missing Link in AI: Unsupervised Learning

- Learn $p(x)$ from unlabeled samples $\{x_i\}$.
- Discover latent variables related to observed variable $x$.

Image Credit: Whizz Education
Unsupervised Learning via Probabilistic Models

Data $\rightarrow$ Model $\rightarrow$ Learning Algorithm $\rightarrow$ Predictions

Challenges in High dimensional Learning

- Dimension of $x \gg$ dim. of latent variable $h$.
- Learning is like finding needle in a haystack.
- Computationally & statistically challenging.
Overview of Unsupervised Learning Methods

Goal: learn model parameters $\theta$ from observations $x$.

- Maximum likelihood: $\max_\theta p(x; \theta)$.
- Non-convex: stuck in local optima.
- Curse of dimensionality: Exponential no. of critical points.
- Heuristics: Expectation Maximization, Variational Inference . . .
- Other mechanisms such as Generative Adversarial Nets also non-convex.
Replace the objective function
Max Likelihood vs. Best Tensor decomp.

Preserves Global Optimum (infinite samples)

\[
\arg\max_{\theta} p(x; \theta) = \arg\min_{\theta} \|\hat{T}(x) - T(\theta)\|_F^2
\]

\(\hat{T}(x)\): empirical tensor, \(T(\theta)\): low rank tensor based on \(\theta\).
Guaranteed Learning through Tensor Methods

Replace the objective function
Max Likelihood vs. Best Tensor decomp.

Preserves Global Optimum (infinite samples)

$$\arg \max_{\theta} p(x; \theta) = \arg \min_{\theta} \| \hat{T}(x) - T(\theta) \|_F^2$$

$\hat{T}(x)$: empirical tensor, $T(\theta)$: low rank tensor based on $\theta$.

Finding globally opt tensor decomposition

Simple algorithms succeed under mild and natural conditions for many learning problems.
Guaranteed Learning through Tensor Methods

Replace the objective function
Max Likelihood vs. Best Tensor decomp.

Preserves Global Optimum (infinite samples)

$$\arg \max_{\theta} p(x; \theta) = \arg \min_{\theta} \| \hat{T}(x) - T(\theta) \|_{F}^{2}$$

$\hat{T}(x)$: empirical tensor, $T(\theta)$: low rank tensor based on $\theta$.

Finding globally opt tensor decomposition
Simple algorithms succeed under mild and natural conditions for many learning problems.
Matrix Decomposition: Discovering Latent Factors

- List of scores for students in different tests
- Learn hidden factors for Verbal and Mathematical Intelligence [C. Spearman 1904]

\[
\text{Score} \ (\text{student}, \text{test}) = \text{student}_{\text{verbal-intlg}} \times \text{test}_{\text{verbal}} + \text{student}_{\text{math-intlg}} \times \text{test}_{\text{math}}
\]
Matrix Decomposition: Discovering Latent Factors

- Identifying **hidden factors** influencing the observations
- Characterized as **matrix decomposition**
Matrix Decomposition: Discovering Latent Factors

- Decomposition is **not necessarily unique**.
- Decomposition cannot be **overcomplete**.
Tensor: Shared Matrix Decomposition

- **Shared** decomposition with different scaling factors
- Combine matrix slices as a tensor
Tensor Decomposition

- Outer product notation:

\[ T = u \otimes v \otimes w + \tilde{u} \otimes \tilde{v} \otimes \tilde{w} \]

\[ T_{i_1, i_2, i_3} = u_{i_1} \cdot v_{i_2} \cdot w_{i_3} + \tilde{u}_{i_1} \cdot \tilde{v}_{i_2} \cdot \tilde{w}_{i_3} \]
Tensor Decomposition

Uniqueness of Tensor Decomposition [J. Kruskal 1977]

- Above tensor decomposition: unique when rank one pairs are linearly independent
- Matrix case: when rank one pairs are orthogonal
Tensor Decomposition

Finding Best Tensor Decomposition? Overcome Non-convexity?
Notion of Tensor Contraction

Extends the notion of matrix product

Matrix product
\[ Mv = \sum_j v_j M_j \]

Tensor Contraction
\[ T(u, v, \cdot) = \sum_{i,j} u_i v_j T_{i,j,\cdot} \]
Symmetric Tensor Decomposition

\[ T = v_1 \otimes^3 + v_2 \otimes^3 + \cdots, \]

Symmetric Tensor Decomposition

Tensor Power Method

\[
\begin{align*}
    v \mapsto \frac{T(v, v, \cdot)}{\|T(v, v, \cdot)\|}.
\end{align*}
\]

\[
T(v, v, \cdot) = \langle v, v_1 \rangle^2 v_1 + \langle v, v_2 \rangle^2 v_2
\]

Symmetric Tensor Decomposition

Tensor Power Method

\[ v \mapsto \frac{T(v, v, \cdot)}{\|T(v, v, \cdot)\|}. \]

\[ T(v, v, \cdot) = \langle v, v_1 \rangle^2 v_1 + \langle v, v_2 \rangle^2 v_2 \]

Orthogonal Tensors

- \( \vec{v}_1 \perp \vec{v}_2 \).
- \( T(v_1, v_1, \cdot) = \lambda_1 v_1 \).

Symmetric Tensor Decomposition

Tensor Power Method

\[ v \mapsto \frac{T(v, v, \cdot)}{\|T(v, v, \cdot)\|}. \]

\[ T(v, v, \cdot) = \langle v, v_1 \rangle^2 v_1 + \langle v, v_2 \rangle^2 v_2 \]

Orthogonal Tensors

- \( \vec{v}_1 \perp \vec{v}_2 \).
- \( T(v_1, v_1, \cdot) = \lambda_1 v_1 \).

Symmetric Tensor Decomposition

Tensor Power Method

\[
\begin{align*}
\mathbf{v} &\rightarrow \frac{T(v, v, \cdot)}{\|T(v, v, \cdot)\|}.
\end{align*}
\]

\[
T(v, v, \cdot) = \langle v, v_1 \rangle^2 v_1 + \langle v, v_2 \rangle^2 v_2
\]

Exponential no. of stationary points for power method:
\[T(v, v, \cdot) = \lambda v\]

---

Symmetric Tensor Decomposition

Tensor Power Method

\[ v \mapsto \frac{T(v, v, \cdot)}{\|T(v, v, \cdot)\|}. \]

\[ T(v, v, \cdot) = \langle v, v_1 \rangle^2 v_1 + \langle v, v_2 \rangle^2 v_2 \]

Exponential no. of stationary points for power method:

\[ T(v, v, \cdot) = \lambda v \]

Stable

Unstable

Other stationary points

Symmetric Tensor Decomposition

Tensor Power Method

\[ v \mapsto \frac{T(v, v, \cdot)}{\|T(v, v, \cdot)\|}. \]

\[ T(v, v, \cdot) = \langle v, v_1 \rangle^2 v_1 + \langle v, v_2 \rangle^2 v_2 \]

Exponential no. of stationary points for power method:
\[ T(v, v, \cdot) = \lambda v \]

For power method on **orthogonal** tensor, no spurious stable points

Non-orthogonal Tensor Decomposition

\[ T = v_1 \otimes^3 + v_2 \otimes^3 + \cdots, \]

Non-orthogonal Tensor Decomposition

Orthogonalization

Input tensor $T$

Non-orthogonal Tensor Decomposition

Orthogonalization

\[ T(W, W, W) = \tilde{T} \]

Non-orthogonal Tensor Decomposition

Orthogonalization

\[ v_1 v_2 W = \tilde{v}_1 \tilde{v}_2 \]

\[ T(W, W, W) = \tilde{T} \]

Non-orthogonal Tensor Decomposition

Orthogonalization

\[ v_1 \quad \tilde{v}_1 \]
\[ v_2 \quad \tilde{v}_2 \]
\[ W \]
\[ \tilde{w}_1 \quad \tilde{w}_2 \]

\[ T(W, W, W) = \tilde{T} \]

\[ \tilde{T} = T(W, W, W) = \tilde{v}_1 \otimes^3 + \tilde{v}_2 \otimes^3 + \cdots \]

Non-orthogonal Tensor Decomposition

Orthogonalization

\[ \begin{align*}
\mathbf{v}_1 & \quad \mathbf{v}_2 & \quad \mathbf{W} \\
\tilde{\mathbf{v}}_1 & \quad \tilde{\mathbf{v}}_2 \\
\end{align*} \]

\[ T(\mathbf{W}, \mathbf{W}, \mathbf{W}) = \tilde{T} \]

Find \( \mathbf{W} \) using SVD of Matrix Slice

\[ M = T(\cdot, \cdot, \theta) = \]

Non-orthogonal Tensor Decomposition

Orthogonalization

Orthogonalization: invertible when $v_i$'s linearly independent.

Guaranteed tensor decomposition: when $v_i$'s linearly independent.

Outline

1 Introduction

2 Tensor Decomposition Algorithms

3 Tensors for Probabilistic Models

4 Tensors in Deep Learning

5 Steps Forward
Tensor Methods for Topic Modeling

- Topic-word matrix: \( P[\text{word} = i | \text{topic} = j] \)
- Linearly independent columns

Moment Tensor: Co-occurrence of Word Triplets
Extracting Communities in Social Networks

Moment Tensor: Common Friends among Node Triplets

Tensors vs. Variational Inference
Criterion: Perplexity = \( \exp[-\text{likelihood}] \).

Learning Topics from PubMed on Spark, 8mil articles

Tensors vs. Variational Inference
Criterion: Perplexity = exp[−likelihood].

Learning Topics from PubMed on Spark, 8mil articles

Learning network communities on single workstation
Facebook $n \sim 20k$, Yelp $n \sim 40k$, DBLP-sub $n \sim 1e5$, DBLP $n \sim 1e6$.

Tensors vs. Variational Inference
Criterion: Perplexity = exp[−likelihood].

Learning Topics from PubMed on Spark, 8mil articles

Learning network communities on single workstation
Facebook $n \sim 20k$, Yelp $n \sim 40k$, DBLP-sub $n \sim 1e5$, DBLP $n \sim 1e6$.

Orders of Magnitude Faster & More Accurate

Learning Representations

Sparse coding prevalent in neural signaling.

Neural sparse coding
[Panadopoulou11]

Learning Representations

Sparse coding prevalent in neural signaling.

Neural sparse coding [Papadopoulou11]

Linear Model with Overcomplete Dictionary


Learning Representations

Contribution: learn overcomplete incoherent dictionaries

Neural sparse coding [Papadopoulou11]  

Linear Model with Overcomplete Dictionary


Learning Representations

Shift-invariant Dictionary

Image

Dictionary

Convolutional Model

\[ \text{Image} \begin{bmatrix} \star & \Box & \bigcirc \end{bmatrix} = \begin{bmatrix} \star & \Box & \bigcirc \end{bmatrix} \ast \begin{bmatrix} \text{Color} \end{bmatrix} + \begin{bmatrix} \text{Color} \end{bmatrix} \ast \begin{bmatrix} \text{Color} \end{bmatrix} \]


Learning Representations
Efficient Tensor Decomposition with Shifted Components

\[ \text{Image} = \text{Dictionary} + \ldots + \text{Dictionary} + \ldots + \text{Dictionary} \]

Shift-invariant Dictionary

Convolutional Model


Fast Text Embeddings through Tensor Methods

Word Embeddings

Car

Accident
Fast Text Embeddings through Tensor Methods

Word Embeddings

Car
Accident

Sentence Embeddings

Sentence
It’s paraphrase
Fast Text Embeddings through Tensor Methods

Paraphrase Detection on MSR corpus with ~ 5000 Sentences
Fast Text Embeddings through Tensor Methods

Paraphrase Detection on MSR corpus with \(~ 5000\) Sentences

<table>
<thead>
<tr>
<th>Method</th>
<th>F score</th>
<th>No. of samples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vector Similarity (Baseline)</td>
<td>75%</td>
<td>(~ 4k)</td>
</tr>
<tr>
<td>Tensor (Proposed)</td>
<td>81%</td>
<td>(~ 4k)</td>
</tr>
<tr>
<td>Skipthought (RNN)</td>
<td>82%</td>
<td>(~ 74)mil</td>
</tr>
</tbody>
</table>

- **Unsupervised** learning of embeddings.
- No outside info for tensor vs. large book corpus (74 million) for skipthought
- Similar story with **holographic embeddings for knowledge bases** by M. Nickel et al.
Reinforcement Learning of Partially Observable Markov Decision Process

Learning in Adaptive Environments

- Learner changes environment
- Hidden state estimation.
Reinforcement Learning of Partially Observable Markov Decision Process

Learning in Adaptive Environments

- Learner changes environment
- Hidden state estimation.

Partially Observable Markov Decision Process

- Design of tensor algorithms under memoryless policies
- Guaranteed regret bounds: comparable to fully observed environment.
Faster convergence to better solution via tensor methods.

Outline

1. Introduction
2. Tensor Decomposition Algorithms
3. Tensors for Probabilistic Models
4. Tensors in Deep Learning
5. Steps Forward
Local Optima in Backpropagation

“..few researchers dare to train their models from scratch.. small miscalibration of initial weights leads to vanishing or exploding gradients.. poor convergence..”

Exponential (in dimensions) no. of local optima for backpropagation

Moments of a Neural Network

\[ \mathbb{E}[y|x] := f(x) = \langle a_2, \sigma(A_1^\top x) \rangle \]

Moments of a Neural Network

\[ E[y|x] := f(x) = \langle a_2, \sigma(A_1^\top x) \rangle \]

Moments using score functions \( S(\cdot) \)

Moments of a Neural Network

Moments using score functions $S(\cdot)$

$$E[y | x] := f(x) = \langle a_2, \sigma(A_1^T x) \rangle$$

Moments of a Neural Network

\[ E[y|x] := f(x) = \langle a_2, \sigma(A_1^\top x) \rangle \]

Moments using score functions \( S(\cdot) \)

\[ E[y \cdot S_2(x)] = + \]

Moments of a Neural Network

\[ E[y|x] := f(x) = \langle a_2, \sigma(A_1^\top x) \rangle \]

Moments using score functions \( S(\cdot) \)

\[ E[y \cdot S_3(x)] = \begin{array}{c}
\text{score function features}
\end{array} + \begin{array}{c}
\text{score function features}
\end{array} \]

Moments of a Neural Network

\[ \mathbb{E}[y|x] := f(x) = \langle a_2, \sigma(A_1^T x) \rangle \]

Moments using score functions \( S(\cdot) \)

\[ \mathbb{E}[y \cdot S_3(x)] = \begin{pmatrix} \text{Blue} \end{pmatrix} + \begin{pmatrix} \text{Red} \end{pmatrix} \]

Given input pdf \( p(\cdot) \), \( S_m(x) := (-1)^m \frac{\nabla^{(m)} p(x)}{p(x)} \).

Gaussian \( x \Rightarrow \) Hermite polynomials.

Tensorizing Neural Networks

- Multi-linear representation of dense layers of CNNs.
  - **Tensor train** format for low rank approximation of weight matrix.
- Compact representation: solves memory problem.

$$Y(i_1, i_2 \ldots) = \sum_{j_1, j_2 \ldots} G(i_1, j_1)G(i_2, j_2) \ldots X(j_1, j_2 \ldots)$$

---

Results on ImageNet

- Compression rate 200,000!
- Negligible performance loss.

---

Hierarchical Tucker tensors for representing arithmetic conv nets.

Employs locality, sharing and pooling.

Exponentially more parameters in shallow net vs. deep net.

Tensors in Memory Embeddings
Human Memory Model. Semantic decoding through Tensor Tucker.

Scaling up and Deploying Tensor Methods

Scaling up

- Dimensionality reduction through sketching.
- Communication efficient methods.

Deployment

- Multi-platform support: CPU, GPU, Cloud, FPGA ...
- Extended BLAS kernels: Beyond linear algebra.
- Many deep learning operations involve tensor contractions.

__________________________

Innovations in Non-Convex Methods

Smoothing and Continuation Methods

- Global approach vs. local search.
- Unified guarantees for non-convex problems?

H. Mobahi, “Training RNNs by Diffusion”.
Innovations in Non-Convex Methods

Learning to add using RNN

H. Mobahi, “Training RNNs by Diffusion”.
Innovations in Non-Convex Methods

- Escaping saddle points in high dimensions?
- Can SGD escape in bounded time?
- Degeneracy of saddle points in various non-convex problems?

Efficient approaches for escaping higher order saddle points in non-convex optimization by A. R. Ge, COLT 2016.
Innovations in Non-Convex Methods

Contribution: First method to escape third order saddle

- Escaping saddle points in high dimensions?
- Can SGD escape in bounded time?
- Degeneracy of saddle points in various non-convex problems?

Efficient approaches for escaping higher order saddle points in non-convex optimization by A., R. Ge, COLT 2016.
Research Connections and Resources

Collaborators
Jennifer Chayes, Christian Borgs, Prateek Jain, Alekh Agarwal & Praneeth Netrapalli (MSR), Srinivas Turaga (Janelia), Michael Hawrylycz & Ed Lein (Allen Brain), Allesandro Lazaric (Inria), Alex Smola (CMU), Rong Ge (Duke), Daniel Hsu (Columbia), Sham Kakade (UW), Hossein Mobahi (MIT).

- Podcast/lectures/papers/software available at http://newport.eecs.uci.edu/anandkumar/