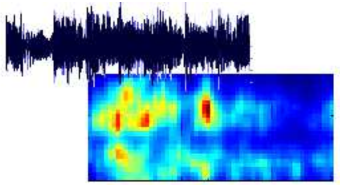
# Tensor Methods for Feature Learning

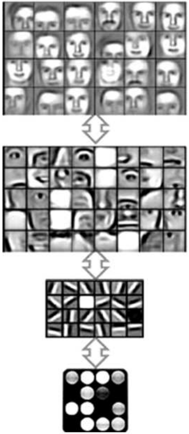
**Anima Anandkumar**

U.C. Irvine

# Feature Learning For Efficient Classification

Find good transformations of input for improved classification

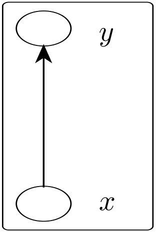




Figures used attributed to Fei-Fei Li, Rob Fergus, Antonio Torralba, et al.

# Principles Behind Feature Learning

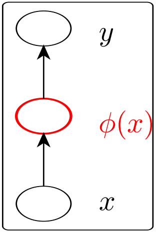
- Classification/regression tasks: Predict $y$ given $x$.
- Find feature transform $\phi(x)$ to better predict $y$.




Feature learning: Learn $\phi(\cdot)$ from data.

# Principles Behind Feature Learning

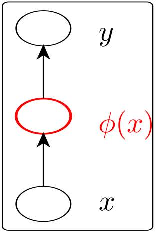
- Classification/regression tasks: Predict $y$ given $x$.
- Find feature transform $\phi(x)$ to better predict $y$.




Feature learning: Learn $\phi(\cdot)$ from data.

# Principles Behind Feature Learning

- Classification/regression tasks: Predict $y$ given $x$.
- Find feature transform $\phi(x)$ to better predict $y$.



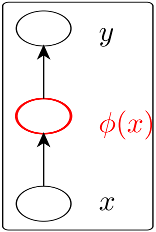

Feature learning: Learn $\phi(\cdot)$ from data.

## Learning $\phi(x)$ from Labeled vs. Unlabeled Samples

- Labeled samples $\{x_i, y_i\}$ and unlabeled samples $\{x_i\}$.
- Labeled samples should lead to better feature learning $\phi(\cdot)$ but are harder to obtain.

# Principles Behind Feature Learning

- Classification/regression tasks: Predict $y$ given $x$.
- Find feature transform $\phi(x)$ to better predict $y$.



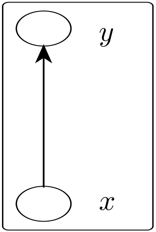

Feature learning: Learn $\phi(\cdot)$ from data.

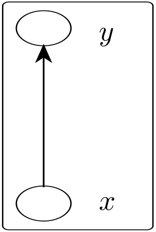
## Learning $\phi(x)$ from Labeled vs. Unlabeled Samples

- Labeled samples $\{x_i, y_i\}$ and unlabeled samples $\{x_i\}$.
- Labeled samples should lead to better feature learning $\phi(\cdot)$ but are harder to obtain.

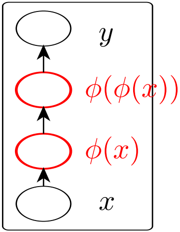
Learn features $\phi(x)$ through latent variables related to $x, y$.

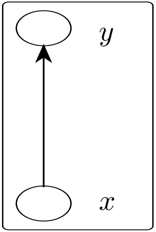
# Conditional Latent Variable Models: Two Cases



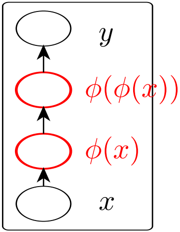
# Conditional Latent Variable Models: Two Cases

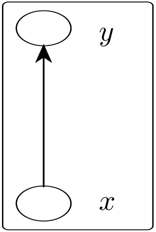


# Conditional Latent Variable Models: Two Cases

Multi-layer Neural Networks

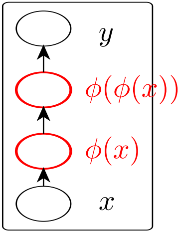


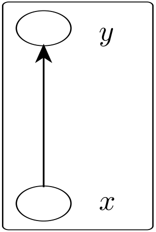



# Conditional Latent Variable Models: Two Cases

Multi-layer Neural Networks

$$\mathbb{E}[y|x] = \sigma(A_d\ \sigma(A_{d-1}\ \sigma(\cdots A_2\ \sigma(A_1 x))))$$

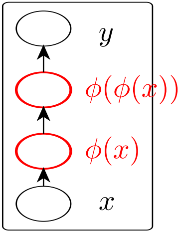


# Conditional Latent Variable Models: Two Cases

## Multi-layer Neural Networks

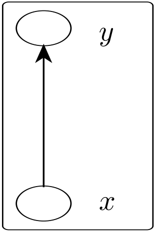
$$\mathbb{E}[y|x] = \sigma(A_d\ \sigma(A_{d-1}\ \sigma(\cdots A_2\ \sigma(A_1 x))))$$




---

## Mixture of Classifiers or GLMs

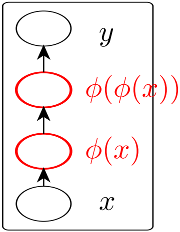
$$G(x) := \mathbb{E}[y|x, h] = \sigma(\langle Uh, x\rangle + \langle b, h\rangle)$$



# Conditional Latent Variable Models: Two Cases

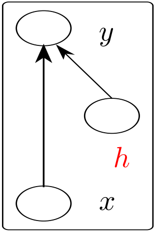
Multi-layer Neural Networks

$$\mathbb{E}[y|x] = \sigma(A_d\ \sigma(A_{d-1}\ \sigma(\cdots A_2\ \sigma(A_1 x))))$$




Mixture of Classifiers or GLMs

$$G(x) := \mathbb{E}[y|x,h] = \sigma(\langle Uh, x\rangle + \langle b, h\rangle)$$



# Challenges in Learning LVMs

## Challenge: Identifiability Conditions

- When can model be identified (given infinite computation and data)?
- Does identifiability also lead to tractable algorithms?

## Computational Challenges

- Maximum likelihood is NP-hard in most scenarios.
- Practice: Local search approaches such as Back-propagation, EM, Variational Bayes have no consistency guarantees.

## Sample Complexity

- Sample complexity needs to be low for high-dimensional regime.

Guaranteed and efficient learning through tensor methods

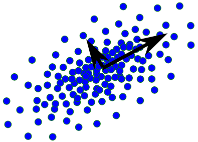
# Outline


1. Introduction
2. Spectral and Tensor Methods
3. Generative Models for Feature Learning
4. Proposed Framework
5. Conclusion

# Classical Spectral Methods: Matrix PCA and CCA

## Unsupervised Setting: PCA

For centered samples $\{x_i\}$, find projection $P$ with $\text{Rank}(P) = k$ s.t.



$$\boxed{\min_{P} \frac{1}{n} \sum_{i \in [n]} \|x_i - Px_i\|^2.}$$

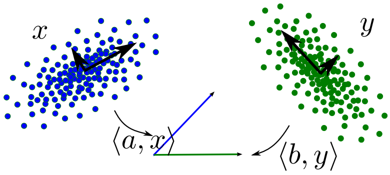
Result: Eigen-decomposition of $S = \text{Cov}(X)$.

---

## Supervised Setting: CCA

For centered samples $\{x_i, y_i\}$, find

$$\boxed{\max_{a,b} \frac{a^\top \hat{\mathbb{E}}[xy^\top] b}{\sqrt{a^\top \hat{\mathbb{E}}[xx^\top] a \; b^\top \hat{\mathbb{E}}[yy^\top] b}}.}$$




Result: Generalized eigen decomposition.

# Beyond SVD: Spectral Methods on Tensors

- How to learn the mixture models without separation constraints?
  - PCA uses covariance matrix of data. Are higher order moments helpful?
- Unified framework?
  - Moment-based estimation of probabilistic latent variable models?
- SVD gives spectral decomposition of matrices.
  - What are the analogues for tensors?

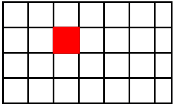
# Moment Matrices and Tensors

## Multivariate Moments

$$M_1 := \mathbb{E}[x], \quad M_2 := \mathbb{E}[x \otimes x], \quad M_3 := \mathbb{E}[x \otimes x \otimes x].$$

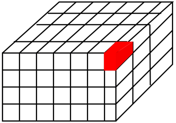
## Matrix

- $\mathbb{E}[x \otimes x] \in \mathbb{R}^{d \times d}$ is a second order tensor.
- $\mathbb{E}[x \otimes x]_{i_1,i_2} = \mathbb{E}[x_{i_1} x_{i_2}]$.
- For matrices: $\mathbb{E}[x \otimes x] = \mathbb{E}[xx^\top]$.



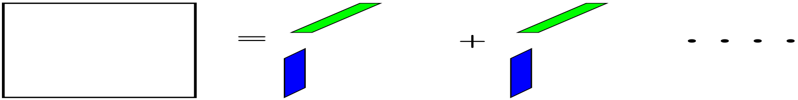
## Tensor

- $\mathbb{E}[x \otimes x \otimes x] \in \mathbb{R}^{d \times d \times d}$ is a third order tensor.
- $\mathbb{E}[x \otimes x \otimes x]_{i_1,i_2,i_3} = \mathbb{E}[x_{i_1} x_{i_2} x_{i_3}]$.



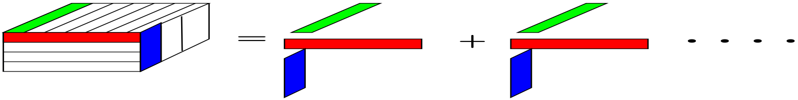
# Spectral Decomposition of Tensors

$$\boxed{M_2 = \sum_i \lambda_i u_i \otimes v_i}$$




---

$$\boxed{M_3 = \sum_i \lambda_i u_i \otimes v_i \otimes w_i}$$




- $u \otimes v \otimes w$ is a rank-1 tensor since its $(i_1, i_2, i_3)^{\text{th}}$ entry is $u_{i_1} v_{i_2} w_{i_3}$.

Guaranteed recovery. (Anandkumar et al 2012, Zhang & Golub 2001).

# Moment Tensors for Conditional Models

Multivariate Moments: Many possibilities...

$$\mathbb{E}[x \otimes y], \mathbb{E}[x \otimes x \otimes y], \mathbb{E}[\phi(x) \otimes y] \ldots.$$

Feature Transformations of the Input: $x \mapsto \phi(x)$

- How to exploit them?
- Are moments $\mathbb{E}[\phi(x) \otimes y]$ useful?
- If $\phi(x)$ is a matrix/tensor, we have matrix/tensor moments.
- Can carry out spectral decomposition of the moments.

# Moment Tensors for Conditional Models

Multivariate Moments: Many possibilities...

$$\mathbb{E}[x \otimes y], \mathbb{E}[x \otimes x \otimes y], \mathbb{E}[\phi(x) \otimes y] \ldots.$$

Feature Transformations of the Input: $x \mapsto \phi(x)$

- How to exploit them?
- Are moments $\mathbb{E}[\phi(x) \otimes y]$ useful?
- If $\phi(x)$ is a matrix/tensor, we have matrix/tensor moments.
- Can carry out spectral decomposition of the moments.

Construct $\phi(x)$ based on input distribution?

# Outline


1 Introduction

2 Spectral and Tensor Methods

3 Generative Models for Feature Learning

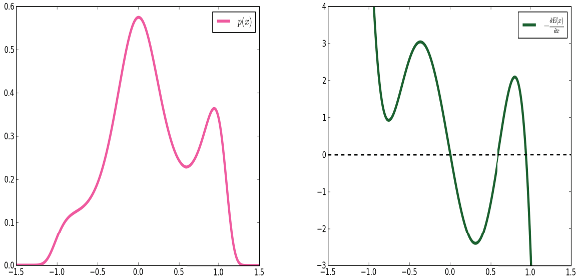
4 Proposed Framework

5 Conclusion

# Score Function of Input Distribution

Score function $\boxed{\mathcal{S}(x) := -\nabla \log p(x)}$

## 1-d PDF  1-d Score




(a) $p(x) = \frac{1}{Z}\exp(-E(x))$

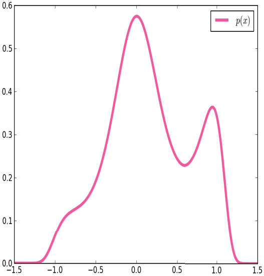
(b) $\frac{\partial}{\partial x}\log p(x) = -\frac{\partial}{\partial x}E(x)$

Figures from Alain and Bengio 2014.

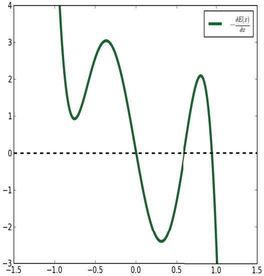
# Score Function of Input Distribution

Score function $\boxed{\mathcal{S}(x) := -\nabla \log p(x)}$

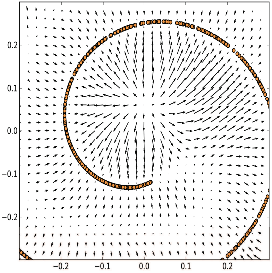
1-d PDF | 1-d Score | 2-d Score




(a) $p(x) = \frac{1}{Z}\exp(-E(x))$




(b) $\frac{\partial}{\partial x}\log p(x) = -\frac{\partial}{\partial x}E(x)$



Figures from Alain and Bengio 2014.

# Why Score Function Features?

$$\boxed{\mathcal{S}(x) := -\nabla \log p(x)}$$

- Utilizes generative models for input.
- Can be learnt from unlabeled data.
- Score matching methods work for non-normalized models.

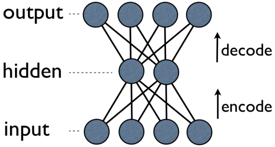
# Why Score Function Features?

$$\mathcal{S}(x) := -\nabla \log p(x)$$

- Utilizes generative models for input.
- Can be learnt from unlabeled data.
- Score matching methods work for non-normalized models.

## Approximation of score function using denoising auto-encoders

$$\nabla \log p(x) \approx \frac{r^*(x+n) - x}{\sigma^2}$$



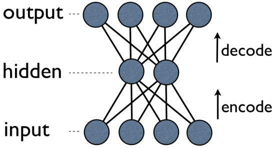
# Why Score Function Features?

$$\boxed{\mathcal{S}(x) := -\nabla \log p(x)}$$

- Utilizes generative models for input.
- Can be learnt from unlabeled data.
- Score matching methods work for non-normalized models.

Approximation of score function using denoising auto-encoders

$$\boxed{\nabla \log p(x) \approx \frac{r^*(x+n) - x}{\sigma^2}}$$




Recall our goal: construct moments $\mathbb{E}[y \otimes \phi(x)]$

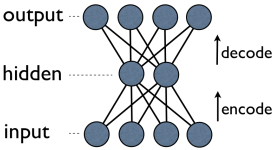
# Why Score Function Features?

$$\boxed{\mathcal{S}(x) := -\nabla \log p(x)}$$

- Utilizes generative models for input.
- Can be learnt from unlabeled data.
- Score matching methods work for non-normalized models.

Approximation of score function using denoising auto-encoders

$$\boxed{\nabla \log p(x) \approx \frac{r^*(x+n) - x}{\sigma^2}}$$




Recall our goal: construct moments $\mathbb{E}[y \otimes \phi(x)]$

Beyond vector features?

# Matrix and Tensor-valued Features

Higher order score functions

$$\boxed{\mathcal{S}_m(x) := (-1)^m \frac{\nabla^{(m)} p(x)}{p(x)}}$$

- Can be a matrix or a tensor instead of a vector.
- Can be used to construct matrix and tensor moments $\mathbb{E}[y \otimes \phi(x)]$.

# Outline

1 Introduction

2 Spectral and Tensor Methods

3 Generative Models for Feature Learning

4 Proposed Framework

5 Conclusion

# Operations on Score Function Features

- Form the cross-moments: $\mathbb{E}\left[y \otimes \mathcal{S}_m(x)\right]$.

# Operations on Score Function Features

- Form the cross-moments: $\mathbb{E}\left[y \otimes \mathcal{S}_m(x)\right]$.

## Our result

$$\boxed{\mathbb{E}\left[y \otimes \mathcal{S}_m(x)\right] = \mathbb{E}\left[\nabla^{(m)} G(x)\right], \qquad G(x) := \mathbb{E}[y|x].}$$

# Operations on Score Function Features

- Form the cross-moments: $\mathbb{E}\left[y \otimes \mathcal{S}_m(x)\right]$.

## Our result

$$\boxed{\mathbb{E}\left[y \otimes \mathcal{S}_m(x)\right] = \mathbb{E}\left[\nabla^{(m)} G(x)\right], \qquad G(x) := \mathbb{E}[y|x].}$$

- Extension of Stein's lemma.

# Operations on Score Function Features

- Form the cross-moments: $\mathbb{E}\left[y \otimes \mathcal{S}_m(x)\right]$.

## Our result

$$\boxed{\mathbb{E}\left[y \otimes \mathcal{S}_m(x)\right] = \mathbb{E}\left[\nabla^{(m)} G(x)\right], \qquad G(x) := \mathbb{E}[y|x].}$$

- Extension of Stein's lemma.

## Extract discriminative directions through spectral decomposition

$$\boxed{\mathbb{E}\left[y \otimes \mathcal{S}_m(x)\right] = \mathbb{E}\left[\nabla^{(m)} G(x)\right] = \sum_{j \in [k]} \lambda_j \cdot \underbrace{u_j \otimes u_j \ldots \otimes u_j}_{m \text{ times}}}.$$

# Operations on Score Function Features

- Form the cross-moments: $\mathbb{E}\left[y \otimes \mathcal{S}_m(x)\right]$.

## Our result

$$\boxed{\mathbb{E}\left[y \otimes \mathcal{S}_m(x)\right] = \mathbb{E}\left[\nabla^{(m)} G(x)\right], \qquad G(x) := \mathbb{E}[y|x].}$$

- Extension of Stein's lemma.

## Extract discriminative directions through spectral decomposition

$$\boxed{\mathbb{E}\left[y \otimes \mathcal{S}_m(x)\right] = \mathbb{E}\left[\nabla^{(m)} G(x)\right] = \sum_{j \in [k]} \lambda_j \cdot \underbrace{u_j \otimes u_j \ldots \otimes u_j}_{m \text{ times}}}.$$

- Construct $\sigma(u_j^\top x)$ for some nonlinearity $\sigma$.

# Automated Extraction of Discriminative Features

Unlabeled data: $\{x_i\}$

General-purpose features: Score functions $\mathcal{S}_m(x)$

Form cross-moments: $\mathbb{E}\left[y \cdot \mathcal{S}_m(x)\right]$

Labeled data: $\{(x_i, y_i)\}$

Our result: obtaining derivatives of label function:

$$\mathbb{E}\left[y \cdot \mathcal{S}_m(x)\right] = \mathbb{E}\left[\nabla^{(m)} G(x)\right],$$

when $\mathbb{E}[y|x] := G(x)$

Spectral/tensor method:

find $u_j$'s s.t. $\mathbb{E}\left[\nabla^{(m)} G(x)\right] = \sum_{j \in [k]} u_j^{\otimes m}$

Extract discriminative features using $u_j$'s/

do model-based prediction with $u_j$'s as parameters

# Learning Mixtures of Classifiers/GLMs

- A mixture of $r$ classifiers, hidden choice variable $h \in \{e_1, \ldots, e_r\}$.

$$\boxed{\mathbb{E}[y|x,h] = g(\langle Uh, x\rangle + \langle b, h\rangle)}$$

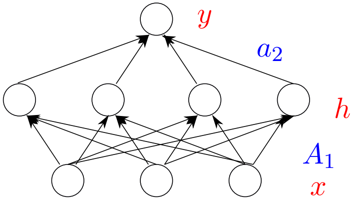
  - $*$ $U = [u_1|u_2 \ldots |u_r]$ are the weight vectors of GLMs.
  - $*$ $b$ is the vector of biases.

$$\boxed{M_3 = \mathbb{E}[y \cdot \mathcal{S}_3(x)] = \sum_{i\in[r]} \lambda_i \cdot u_i \otimes u_i \otimes u_i.}$$

First results for learning non-linear mixtures using spectral methods

# Learning Multi-layer Neural Networks

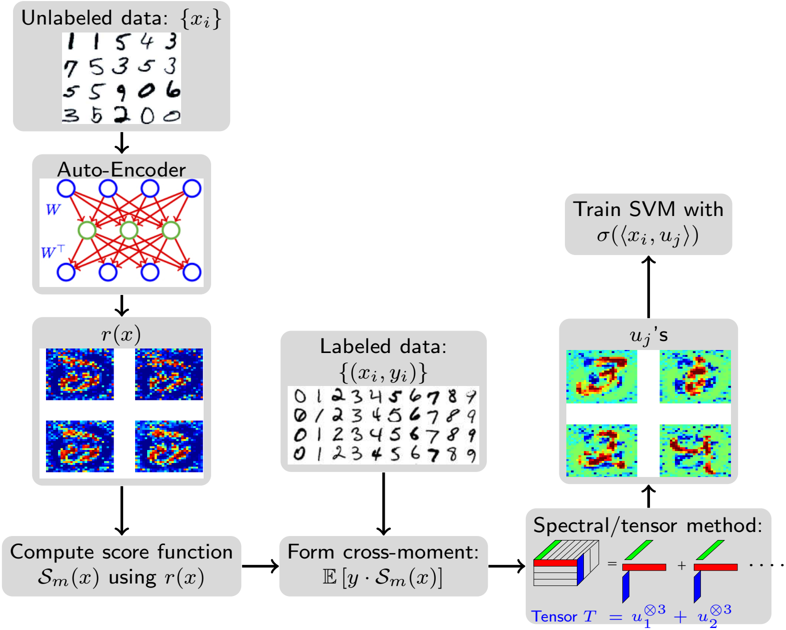
$$\boxed{\mathbb{E}[y|x] = \langle a_2, \sigma(A_1^\top x)\rangle}.$$




Our result

$$\boxed{M_3 = \mathbb{E}[y \cdot \mathcal{S}_3(x)] = \sum_{i \in [r]} \lambda_i \cdot A_{1,i} \otimes A_{1,i} \otimes A_{1,i}.}$$

# Framework Applied to MNIST



# Outline


1. Introduction
2. Spectral and Tensor Methods
3. Generative Models for Feature Learning
4. Proposed Framework
5. Conclusion

# Conclusion: Learning Conditional Models using Tensor Methods

## Tensor Decomposition

- Efficient sample and computational complexities
- Better performance compared to EM, Variational Bayes etc.
- Scalable and embarrassingly parallel: handle large datasets.

## Score function features

- Score function features crucial for learning conditional models.

## Related: Guaranteed Non-convex Methods

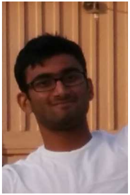
- Overcomplete Dictionary Learning/Sparse Coding: Decompose data into a sparse combination of unknown dictionary elements.
- Non-convex robust PCA: Same guarantees as convex relaxation methods, lower computational complexity. Extensions to tensor setting.

# Co-authors and Resources

Majid Janzamin

Hanie Sedghi

Niranjan UN



- Papers available at `http://newport.eecs.uci.edu/anandkumar/`