Recovery of Joint Probability Distributions from One-Way Marginals Using Low Rank Tensors and Random Projections

Jian Vora

Department of Electrical Engineering, IIT Bombay

jianvora@iitb.ac.in

May 14, 2021

Overview

Problem Statement

The ever classical problem of given N random variables $X_1, X_2, ..., X_N$, and their realisations as samples, we need to estimate $p(X_1, X_2, ..., X_N)$

Lots of work in this domain like Gaussian mixture models, Bayesian Nets, Markov Random Fields, Variational Autoencoders ...

Can we estimate the joint density from the marginals? Does transforming the data help in the above goal? Do we have any prior knowledge of naturally occurring probability densities?

Low Rank of the Joint Probability

Q. Do we have any prior knowledge of joint probability densities? Yes!

- For real world data, random variables are reasonably (in)dependent
- For example, in the case of images, we know neighbouring pixels values are highly correlated while value of far off pixels are almost independent.

This implies that the joint probability density $p(X_1, X_2, ..., X_N)$ has a *low-rank*. But what does it mean for a tensor to have a low-rank?

We shall now explore a model which helps in capturing this low-rankness of a tensor which will be used to model the joint density throughout this talk.

Canonical Polyadic Decomposition of a Tensor

$N-$way tensor $\boldsymbol{Z} \in \mathbb{R}^{I_1 \times I_2 \times I_3 \times \ldots \times I_N}$ admits a Canonical Polyadic Decomposition(CPD) if it can be decomposed as a sum of F rank-1 tensors. Let $a \otimes b$ denote the outerproduct of two vectors, then formally the CPD model can be stated as follows:

$$\boldsymbol{Z} = \sum_{f=1}^{f=F} \boldsymbol{\lambda}(f) \boldsymbol{A_1}(:, f) \otimes \boldsymbol{A_2}(:, f) \otimes \boldsymbol{A_3}(:, f) \otimes \ldots \otimes \boldsymbol{A_N}(:, f)$$

where $\boldsymbol{\lambda} \in \mathbb{R}^F$ and $\boldsymbol{A_i} \in \mathbb{R}^{I_i \times F}$.

The joint probability density $p(X_1, X_2, ..X_N)$ is an $N-$way tensor(referred to as $\boldsymbol{Z}$ henceforth), more specifically

$$\boldsymbol{Z}(x_1, x_2, \ldots x_N) = p(X_1 = x_1, X_2 = x_2, ..X_N = x_N)$$

Canonical Polyadic Decomposition of a Tensor(Notation)

- $\{A_i\}_{i=1}^{N}$ are referred to as mode latent factors
- Elements of λ are referred to as component mixing weights
- Z is the core pmf tensor which we aim to recover
- F is the rank of the tensor

For a tensor Z permitting a Canonical Polyadic Decomposition, we refer $Z = [\lambda, A_1, A_2, ..., A_N]$. Thus there exists a bijection between Z and the set $[\lambda, A_1, A_2, ..., A_N]$ and recovering the core tensor is equivalent to recovering the mode latent factors and the mixing weights.

This drastically reduces the number of parameters to be estimated which were previously exponential in N!

Canonical Polyadic Decomposition of a Tensor

The CPD model can be considered as a naive bayes model with the hidden state H taking a bounded number of states(= the rank F)

$$Pr(X_1 = i_1, X_2 = i_2, .., X_N = i_n) = \sum_{f=1}^{f=F} Pr(H = f) \prod_{n=1}^{n=N} Pr(X_n = i_n | H = f)$$

with $\boldsymbol{\lambda}(f) = Pr(H = f)$ and $\boldsymbol{A}_n(i_n, f) = Pr(X_n = i_n | H = f)$

Uniqueness of the CPD Model

For a tensor $\boldsymbol{Z}$ of rank F, we say that a decomposition $\boldsymbol{Z} = [\boldsymbol{\lambda}, \boldsymbol{A_1}, \boldsymbol{A_2}, .., \boldsymbol{A_N}]$ is essentially unique if the factors are unique up to a common permutation and scaling / counter-scaling of columns of the mode latent factors.

Random Linear Projections

For a given vector $X \in \mathbb{R}^N$, we define the following operation as random linear projections:

$$Y = \Phi X$$

where $\Phi \in \mathbb{R}^{M \times N}$ is composed of entries drawn i.i.d from a standard normal Gaussian or a bernoulli $\{-1, 1\}$ distribution.

Used a lot in compressed sensing, dimensionality reduction, GMM learning

Radon and Inverse Radon Transform

The Radon transform of a 2-D function $f(x, y)$ is defined as:

$$R(r, \theta)[f] = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} f(x, y)\delta(r - xcos\theta - ysin\theta)dxdy$$

The above operation is invertible and is referred to as the inverse radon transform. Filtered backprojection(FBP) is one of the common algorithms used for inverse radon transform defined as:

$$\hat{f}(x, y) = \int_{0}^{\pi} \int_{-\infty}^{+\infty} R(v, \theta)H(v)dvd\theta$$

$H(v)$ is the filter transfer function with some common choices being Ram-Lak, Cosine, Shepp Logan. θ is the key parameter which determines the direction of projection.

Density Estimation from Random Projections

Consider a random vector $X \in \mathbb{R}^N$ and another random vector $\theta \in \mathbb{R}^N$

$\theta^T X$ is a scalar variable and hence we can estimate $p(\theta^T X)$ which is a 1D density estimate and is fairly easy to do using either histogramming or kernel density estimate.

We can we this for various angles θ and hence we can accumulate a set of 1D densities of the form $p(\theta_i^T X)$. But do these mean anything in our original goal of estimating the joint density $p(X_1, X_2, ..., X_N)$?

We can infact relate this 1D densities of random projections to the core pmf tensor $\boldsymbol{Z}$. It is tomography which comes to our rescue!

Density Estimation from Random Projections

Let $\theta = [\theta_1, \theta_2, ..., \theta_N]$ and $X = [X_1, X_2, ..., X_N]$, then

$$p(\theta^T X = t) = p(\sum_i \theta_i X_i = t)$$

$$p(\theta^T X = t) = \sum_{a_1} \sum_{a_2} .. \sum_{a_{N-1}} p(\theta_1 X_1 = a_1, \theta_2 X_2 = a_2, ... \theta_N X_N = t - \sum_{i=1}^{i=N-1} a_i)$$

Thus the 1D density estimate of a projection of a data vector is infact the radon transform of the joint probability tensor taken at an angle θ.

Thus performing an inverse radon transform approach on these densities projected along various angles gives us the original core tensor!

[Sullivan et.al, Journal of the Royal Statistical Society, 1993]

Empirical Results

We perform preliminary experiments on recovering just 2 dimensional densities from these 1D projections. Inverting the radon transform becomes costly as the dimensionality increases.

Evaluation Metric: Jensen-Shannon Divergence

$$JS(P,Q) = \frac{KL(P||(P+Q)/2) + KL(Q||(P+Q)/2)}{2}$$

Empirical Results on GMMs

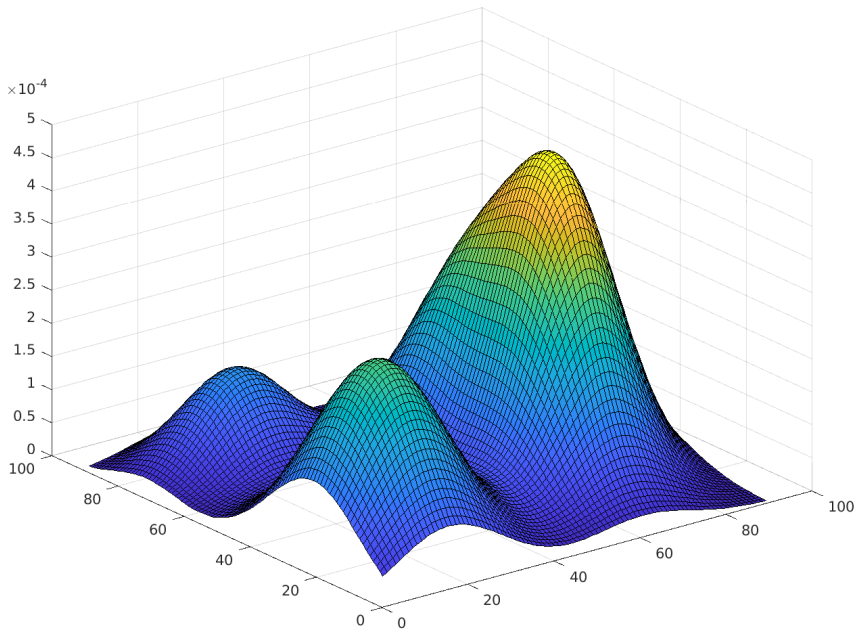

Figure: GMM 1 (4 components)

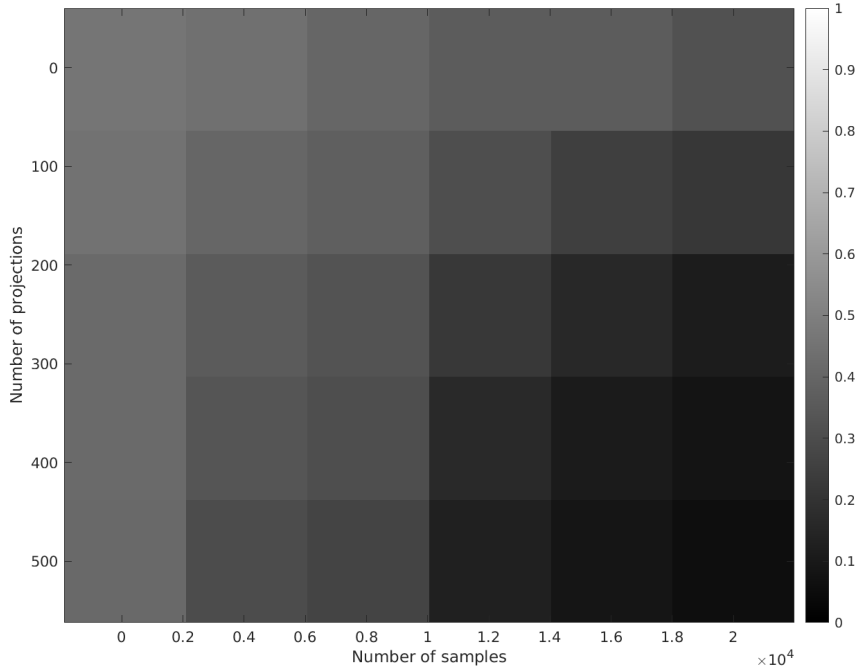

Figure: GMM 1 Errors

Empirical Results on GMMs

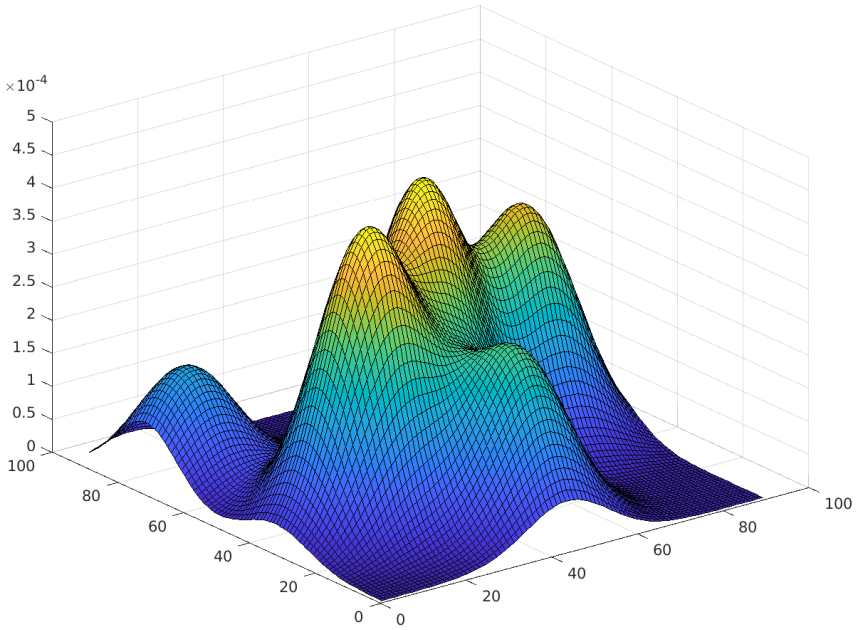

Figure: GMM 2 (9 components)

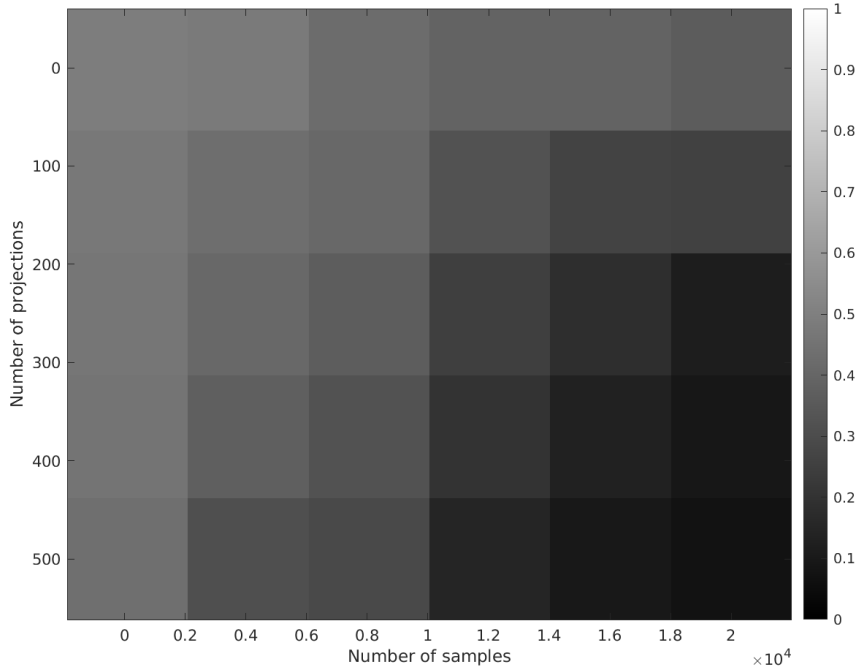

Figure: GMM 2 Errors

Joint Recovery from 3-way marginals

[Kargas et al, IEEE TSP 2018] recovered joint from 3-way marginals. We refer to this algorithm as CTF henceforth

$$\min_{\{\boldsymbol{A_n}\},\boldsymbol{\lambda}} \sum_{j} \sum_{k>j} \sum_{l>k} \|\boldsymbol{Z}_{j,k,l} - [\boldsymbol{\lambda}, \boldsymbol{A_j}, \boldsymbol{A_k}, \boldsymbol{A_l}]\|_F^2$$

[Yeredor et al., Sig Pro. Letters, 2019] proposed a EM framework which we refer to as RAND-EM

$$\min_{\{\boldsymbol{A_n}\},\boldsymbol{\lambda}} - \sum_{t=1}^{t=T} \log \sum_{f=1}^{f=F} \lambda_f \prod_{n=1}^{n=N} \boldsymbol{A_n}(y[t], f)$$

[Ibrahim et al., arxiv 2006.16912] recovered using 2-way marginals by proposing the SPA algorithm

$$\min_{\{\boldsymbol{A_n}\},\boldsymbol{\lambda}} \sum_{j} \sum_{k>j} \|\boldsymbol{Z}_{j,k} - \boldsymbol{A_j} \boldsymbol{D}(\boldsymbol{\lambda}) \boldsymbol{A_k^T}\|_F^2$$

SPA Algorithm

Construct the 2-way marginals $\boldsymbol{Z}_{j,k} = \boldsymbol{A_j D}(\boldsymbol{\lambda})\boldsymbol{A_k^T}$ using histogramming

Split the N variables into 2 sets with M being the index of the split. Thus we have 2 splits namely $\{1, 2, ..., M\}$ and $\{M+1, M+2, ..., N\}$

Arrange the 2-way marginals in the form a matrix $\tilde{\boldsymbol{Z}} \in \mathbb{R}^{MI \times (N-M)I}$ where the block $\tilde{\boldsymbol{Z}}_{\boldsymbol{i},\boldsymbol{j}} = \boldsymbol{Z}_{\boldsymbol{i},\boldsymbol{M}+\boldsymbol{j}}$ where NMF can be applied to get the factors

SPA (Successive Projection Algorithm) is a linear algebra based algorithm to non-negative matrix factorisation

Algorithm for high-dimensional density estimation

On fixing j and k and stacking M one-dimensional PMFs $p(\phi_m^t \boldsymbol{X_{j,k}})$ row-wise, we obtain a matrix $\boldsymbol{Y}_{j,k} \in \mathbb{R}^{M \times I}$ (thus known to us)

$$J(\{\boldsymbol{A_n}\}_{n=1}^N, \boldsymbol{\lambda}) = \sum_{j,k>j} \|\boldsymbol{Y}_{j,k} - \Re(\boldsymbol{A_j} D(\boldsymbol{\lambda}) \boldsymbol{A_k}^T)\|_F^2$$

We cannot directly optimise $J(.)$, say via gradient descent updates, as the mode factors will not be identifiable when $F > I$ as argued in Ibrahim et al. To circumvent, we introduce an auxiliary variable $\boldsymbol{Z}_{j,k}$ with the constraint $\boldsymbol{Z}_{j,k} = \boldsymbol{A_j} D(\boldsymbol{\lambda}) \boldsymbol{A_k}^T$ and transform it into an unconstrained problem by adding a penalty term, namely

$$J_1(\{\boldsymbol{A_n}\}_{n=1}^N, \boldsymbol{\lambda}, \{\boldsymbol{Z}_{j,k}\}) \triangleq \sum_{j,k>j} \|\boldsymbol{Y}_{j,k} - \Re(\boldsymbol{Z}_{j,k})\|_F^2 + \rho \|\boldsymbol{Z}_{j,k} - \boldsymbol{A_j} D(\boldsymbol{\lambda}) \boldsymbol{A_k}^T\|_F^2. \quad (1)$$

Algorithm 1 Recovering Mode Latent Factors from 1D Densities of Random Projections

procedure JUROR

Set $\{\boldsymbol{Z}_{j,k}\}_{(j,k)\in B}$ to be equal to argmin $\sum_{j,k>j} \|\boldsymbol{Y}_{j,k} - \Re(\boldsymbol{Z}_{j,k})\|_F^2$

Assemble $\widetilde{\boldsymbol{Z}}$ using $\{\boldsymbol{Z}_{j,k}\}_{(j,k)\in B}$

$\boldsymbol{W}^{(0)}, \boldsymbol{H}^{(0)} \leftarrow \text{SPA}(\widetilde{\boldsymbol{Z}})$

STAGE 1

converged $\leftarrow$ False, $\rho \leftarrow \rho_0$, $q \leftarrow 1$

while converged == False **do**

  Fetch $\{\boldsymbol{A}_n\}_{n=1}^{n=N}, \boldsymbol{\lambda}$ from $\boldsymbol{W}^{(q-1)}, \boldsymbol{H}^{(q-1)}$

  $\boldsymbol{Z}_{j,k}^{(n)} = (\Re^T\Re + \rho I)^{-1}(\Re^T\boldsymbol{Y}_{j,k} + \rho\boldsymbol{A}_j D(\boldsymbol{\lambda})\boldsymbol{A}_k^T)$

  Assemble $\widetilde{\boldsymbol{Z}}$ using $\{\boldsymbol{Z}_{j,k}^{(n)}\}_{(j,k)\in B}$

  $\boldsymbol{W}^{(q)}, \boldsymbol{H}^{(q)} \leftarrow \text{SPA}(\widetilde{\boldsymbol{Z}})$

  $q = q + 1$

  if $J_1(.) < \epsilon$ **then** converged $\leftarrow$ True

Fetch $\{\boldsymbol{A}_n^0\}_{n=1}^{n=N}, \boldsymbol{\lambda}^0$ from $\boldsymbol{W}^{(q)}, \boldsymbol{H}^{(q)}$

STAGE 2

converged$\leftarrow$ False, $q \leftarrow 1$

while converged == False **do**

  for k in 1 to N **do**

    $\boldsymbol{A}_k^{(q)} \leftarrow \text{ProjectOnSimplex}(\boldsymbol{A}_k^{(q-1)} - \eta_q \frac{\partial J_1}{\partial \boldsymbol{A}_k})$

  $\boldsymbol{\lambda}^{(q)} \leftarrow \text{ProjectOnSimplex}(\boldsymbol{\lambda}^{(q-1)} - \eta_q \frac{\partial J_1}{\partial \boldsymbol{\lambda}})$

  $q = q + 1$

  if $J(.) < \epsilon$ **then** converged $\leftarrow$ True

STAGE 3

return $\{\boldsymbol{A}_n\}_{n=1}^{n=N}, \boldsymbol{\lambda}$

Empirical Results I

N_s	100	1000	5000	10000	50000
CTF	0.339	0.294	0.233	0.172	0.103
SPA	0.295	0.264	0.196	0.143	0.084
JUROR-A	**0.253**	**0.205**	**0.174**	0.138	0.092
JUROR-B	0.267	0.229	0.182	**0.131**	**0.087**
JUROR-C	0.262	0.217	0.185	0.140	0.098
RAND-EM	0.284	0.246	0.204	0.149	0.106

Table: MSE for PMFs for $F = 25, I = 10, N = 6, \kappa = 1$

N_s = Number of samples
F = Rank of Tensor
I = Number of states
N = Dimensionality
κ = Fraction of features observed

Empirical Results II

N_s	100	1000	10000	50000
CTF	0.316	0.151	0.142	**0.092**
SPA	0.238	0.162	0.126	0.117
JUROR-A	**0.205**	**0.147**	0.131	0.103
JUROR-B	0.225	0.159	**0.118**	0.107
JUROR-C	0.218	0.154	0.126	0.114
RAND-EM	0.269	0.173	0.152	0.136

Table: MSE for PMFs for $F = 20$, $I = 15$, $N = 5$, $\kappa = 0.8$

N_s = Number of samples
F = Rank of Tensor
I = Number of states
N = Dimensionality
κ = Fraction of features observed

Empirical Results III

$M(\downarrow)$	100	1000	5000	10000	50000
200	0.253	0.205	0.174	0.138	0.092
150	0.264	0.214	0.188	0.151	0.106
100	0.289	0.267	0.223	0.184	0.129
50	0.384	0.357	0.305	0.264	0.213

Table: MSE for PMFs for $F = 25, I = 10, N = 6, \kappa = 1$

M = Number of random projection
F = Rank of Tensor
I = Number of states
N = Dimensionality
κ = Fraction of features observed

Classification Experiments

UCI Repository Datasets

Car Dataset
Number of Samples - 1728 (70% used for training)
Attributes: 6 categorical like maintenance, passengers, safety etc.
Number of Classes: 4

Mushroom Dataset
Number of Samples - 8124 (50% used for training)
Attributes: 22 categorical
Number of Classes: 4

The classification is done by assigning class label
$\hat{y} = \text{argmax}_y p(y|\boldsymbol{x}) = \text{argmax}_y p(\boldsymbol{x}, y)/p(\boldsymbol{x})$ to $\boldsymbol{x}$

Empirical Results IV

Algorithm	Car	Mushroom
CTF	84.92	95.13
SPA	86.45	96.01
JUROR-A	87.59	**96.72**
JUROR-B	86.37	95.12
JUROR-C	**88.38**	95.79
RAND-EM	82.68	94.65
LOGISTIC REGRESSION	82.37	95.86
SVM-RBF	78.32	95.74
NAIVE BAYES	84.39	89.67
NEURAL NET	85.16	96.37

Table: Classification Accuracies on real-world datasets

Prior Literature

One may note that mixture of smooth product distributions is essentially the CPD model where each column of the mode factors is a pdf instead of a pmf

Learning Mixtures of Smooth Product Distributions[Kargas et al., AISTATS 2019]
This particular work assumes that each mode factor CDF is smooth and band-limited and thus it can be reconstructed from sufficiently closely spaced discrete samples followed by invoking the Nyquist sampling theorem.

A Low-Rank Characteristic Function Approach for Density Estimation [Amridi et al., Arxiv 2008.12315]

In this work, they model the characteristic function (fourier transform of the PDF) as a low rank tensor from which they try to recover the original PDF.

Prior Literature

Sliced Wasserstein Distance for Learning Gaussian Mixtures[Kolouri et al., CVPR 2018]

1. Randomly project the samples on 1D space where the direction of projection will have all N entries in it (basically, not sparse)
2. Fit either a kernel density estimator on the projected 1D samples or have point masses on the projections to get the base density I_y
3. This density I_y is approximated by a Radon transformed GMM where the distance used in approximation is the 1D Wasserstein distance. The parameters of the GMM are updated so that the 1D Wasserstein distance of the Radon transformed GMM is closer to I_y

This leads to a smoother optimization landscape

But their number of random projections required for a good high dimensional recovery scales exponentially in the dimensionality of the data d

Learning GMMs from 1D Marginals

Recalling the CPD model, if we let $A_n(:, f) = \mathcal{N}(\mu_i, \Sigma_i)$, then the density we recover is a GMM where each component has a diagonal covariance matrix.

As done before, estimated densities on one-dimensional space for all the $\binom{d}{2}$ pairs of random variables. Now, for each of these pairs, we can invoke the sliced wasserstein module in order to get two-way marginals of the form $p(X_i, X_j)$ which will also be a GMM.

Note that this is not expensive as we are just going to 2-way form 1-way marginals. This method is polynomial in the dimensionality d as opposed to the exponential in d cost decribed earlier.

How to go to d dimensions now?

Learning GMMs from 1D Marginals

$$P = \sum_i \alpha_i \mathcal{N}(\mu_i, \Sigma_i) \text{ and } Q = \sum_i \beta_i \mathcal{N}(m_i, S_i)$$

The MMD distance between the two GMMs P and Q is defined as below:

$$MMD(P, Q) = \sqrt{K(P, P) + K(Q, Q) - 2K(P, Q)}$$

For GMMs P and Q, we define the $K(.,.)$ function as follows:

$$K(P, Q) = \sum_{i,j} \alpha_i \beta_j K(P_i, Q_j) \text{ where } X_i \text{ is the ith component of GMM } X$$

$$K(P_i, Q_j) = (2\pi\sigma^2)^{d/2} \mathcal{N}(\mu; \hat{\mu}, \Sigma + \hat{\Sigma} + \sigma^2 I)$$

Learning GMMs from 1D Marginals

We minimize the following loss function:

$$J = \sum_k MMD^2(P_k, \hat{P}_k)$$

where k is an index covering $\binom{N}{2}$ values, P_k is the parameterised (which are to estimated) 2-way marginal and $\hat{P}_k$ is the estimated 2-way marginal.

Simply do a GD over the parameters?
a. We need to ensure that the convariance matrix is positive-semi-definite after each iteration and this requires projection on the SPD manifold.

b. The mixing weights of the components are also projected to ensure that they follow the simplex constraints - they should be non-negative and should sum to one.

Empirical Results

Random Initialization gets stuck in spurious minimas!

Hence, intialize the parameters using the estimated marginals (not possible for GMMs as association is not known). Hence tried with only Gaussians as of now.

Experimental Conditions: Dimensionality of the dataset: 3
Number of Samples: 1000
Final Value of Loss Function: 0.157
True Log-Likelihood of the Samples: -113.46
Log-Likelihood of the Samples from the estimated Gauss: -115.34

Experimental Conditions: Dimensionality of the dataset: 8
Number of Samples: 50000
Final Value of Loss Function: 0.229
True Log-Likelihood of the Samples: -538.92
Log-Likelihood of the Samples from the estimated Gauss: -542.18

Future Work

1. Improvise the CPD model in order to make it more expressive (such as a graph structure) as opposed to a simple naive bayes model and see if the joint recovery from marginals theory applies there.

2. Extend the Radon theory to joint PDF recovery from marginals for a large variety of mixture model (GMMs are just a starting point). The broad long term goal could be to design a framework so that we can handle a big class of density estimators and standard methods like GMM learning, KDE etc. come out to be as special cases of this framework.

3. Scale the algorithm to very high dimensional data sets(for example 500) which is not possible in the current framework because of the limitations of the SPA algorithm (in the discrete setting).

References

1. K. P. Murphy, Machine learning: a probabilistic perspective. MIT press.
2. N. Kargas, N. D. Sidiropoulos, and X. Fu, "Tensors, learning, and 'Kolmogorov extension'for finite-alphabet random vectors," IEEE Trans. Signal Process., vol. 66, no. 18, pp. 4854–4868, 2018.
3. A. Yeredor and M. Haardt, "Maximum likelihood estimation of a lowrank probability mass tensor from partial observations," IEEE Signal Process. Lett., vol. 26, no. 10, pp. 1551–1555, Oct 2019
4. S. Ibrahim, X. Fu, Recovering Joint Probability of Discrete Random Variables from Pairwise Marginals - https://arxiv.org/abs/2006.16912
5. Tensors and Probability: An Intriguing Union - N. Sidiropoulos, N. Kargas, X. Fu, GlobalSIP 2018 Keynote
6. Finbarr O'Sullivan and Yudi Pawitan, Multidimensional Density Estimation by Tomography, Journal of the Royal Statistical Society. Series B (Methodological) , 1993, Vol. 55, No. 2 (1993), pp. 509-521

References

1. S. Dasgupta. Experiments with random projection. Sixteenth Conference on Uncertainty in Artificial Intelligence (UAI), 2000
2. M Amiridi, N Kargas, ND Sidiropoulos, Nonparametric Multivariate Density Estimation: A Low-Rank Characteristic Function Approach - arXiv preprint arXiv:2008.12315, 2020 - arxiv.org
3. Kolouri, Soheil, Gustavo K. Rohde, and Heiko Hoffmann. "Sliced Wasserstein Distance for Learning Gaussian Mixture Models." 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (2018)
4. N. Kargas and N. D. Sidiropoulos, Learning Mixtures of Smooth Product Distributions: Identifiability and Algorithm, International Conference on Artificial Intelligence and Statistics (AISTATS), 2019
5. Zhi Gao, Yuwei Wu, Yunde Jia, Mehrtash Harandi; Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 7700-7709

Questions?