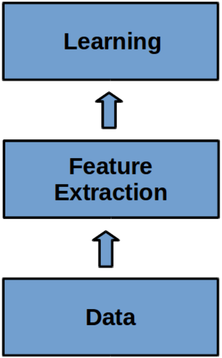
Learning from Complex and Relational Data

Piyush Rai

IIT Kanpur

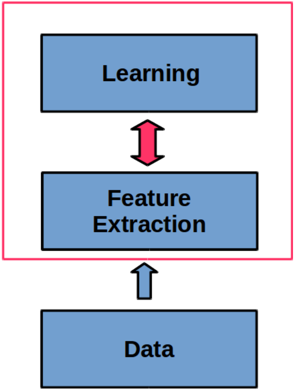
Dec 1, 2015

Learning from Data: The Traditional Way



A two-stage process. Stage 1 often hand-crafted.

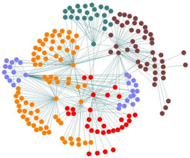
Learning via "Feature Learning"



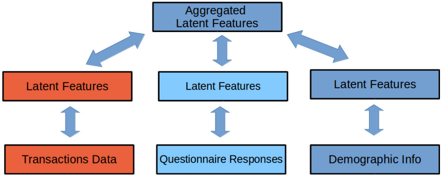
Learning features tuned for specific tasks. Lot of recent buzz (e.g., deep learning).

Why **Learn** Features?

- May not be obvious which features to use (too many, possibly noisy, features)
- Data may not have explicitly defined features (e.g., data is a graph)

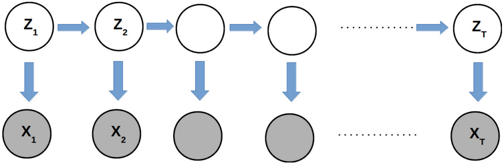


- Data may be heterogeneous and multi-modality



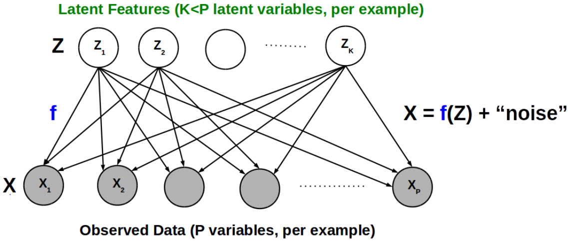
Why Learn Features?

- Data may be temporally evolving ("drifting distribution")



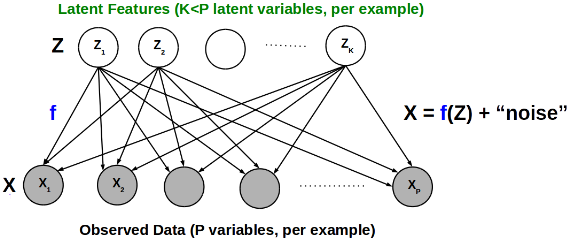
Generative Models for Feature Learning

- Assume that the observations arise from a generative process



Generative Models for Feature Learning

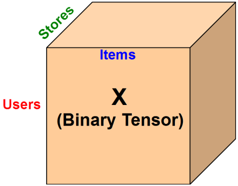
- Assume that the observations arise from a generative process



- Can be extended to multiple layers of latent features (Deep Learning)
- Many other advantanges
  - Handling missing/noisy observations
  - Learning the "right" number of latent features

Feature Learning from Tensors

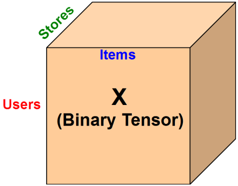
- Tensor: generalization of matrices to more than two dimensions or "ways"
- Assume $\mathbf{X}$ is a binary 3-way tensor of size $N_U \times N_I \times N_S$



- $X_{uis} = 1$ denotes a past transaction involving user u, item i and store s

Feature Learning from Tensors

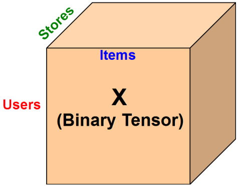
- Tensor: generalization of matrices to more than two dimensions or "ways"
- Assume $\mathbf{X}$ is a binary 3-way tensor of size $N_U \times N_I \times N_S$



- $X_{uis} = 1$ denotes a past transaction involving user u, item i and store s
- Problems one might care about:
  - How to predict if a new transaction (u,i,s) is likely?

Feature Learning from Tensors

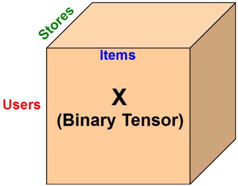
- Tensor: generalization of matrices to more than two dimensions or "ways"
- Assume $\mathbf{X}$ is a binary 3-way tensor of size $N_U \times N_I \times N_S$



- $X_{uis} = 1$ denotes a past transaction involving user u, item i and store s
- Problems one might care about:
  - How to predict if a new transaction (u,i,s) is likely?
  - How to cluster users/items/stores?

Feature Learning from Tensors

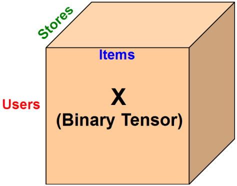
- Tensor: generalization of matrices to more than two dimensions or "ways"
- Assume $\mathbf{X}$ is a binary 3-way tensor of size $N_U \times N_I \times N_S$



- $X_{uis} = 1$ denotes a past transaction involving user u, item i and store s
- Problems one might care about:
  - How to predict if a new transaction (u,i,s) is likely?
  - How to cluster users/items/stores?
  - How to rank users/items/stores?

Feature Learning from Tensors

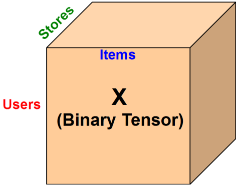
- Tensor: generalization of matrices to more than two dimensions or "ways"
- Assume **X** is a binary 3-way tensor of size $N_U \times N_I \times N_S$



- $X_{uis} = 1$ denotes a past transaction involving user u, item i and store s
- Problems one might care about:
  - How to predict if a new transaction (u,i,s) is likely?
  - How to cluster users/items/stores?
  - How to rank users/items/stores?
  - How to make new recommendations (e.g., user-stores, user-items)?

Feature Learning from Tensors

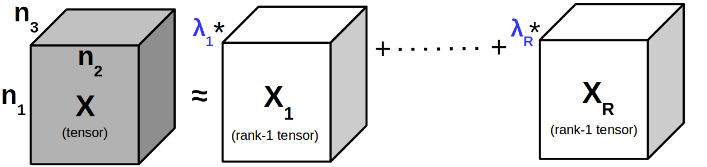
- Tensor: generalization of matrices to more than two dimensions or "ways"
- Assume $\mathbf{X}$ is a binary 3-way tensor of size $N_U \times N_I \times N_S$



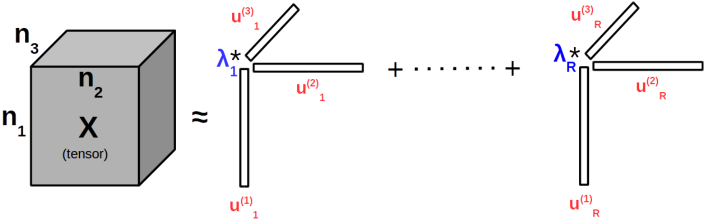
- $X_{uis} = 1$ denotes a past transaction involving user u, item i and store s
- Problems one might care about:
    - How to predict if a new transaction (u,i,s) is likely?
    - How to cluster users/items/stores?
    - How to rank users/items/stores?
    - How to make new recommendations (e.g., user-stores, user-items)?
- Learning features for users/items/stores is the key

Tensor Decomposition

Low-Rank approximation: Express a tensor as a weighted sum of rank-1 tensors



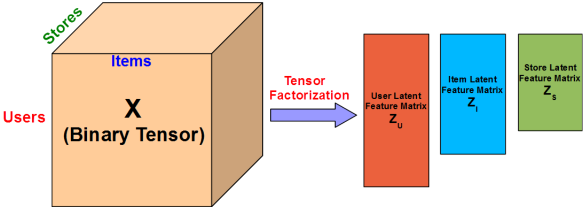
Each rank-1 tensor is an outer product of a set of column vectors (factors)



Akin to **Singular Value Decomposition** (SVD) used for matrix factorization

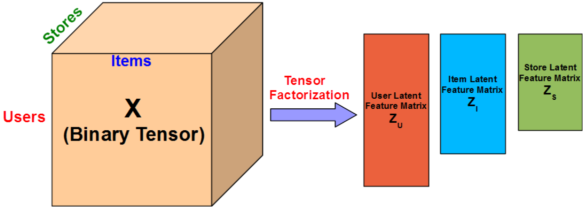
Feature Learning via Tensor Decomposition

Original tensor decomposed into factor matrices (one for each tensor dimension)



Feature Learning via Tensor Decomposition

Original tensor decomposed into factor matrices (one for each tensor dimension)

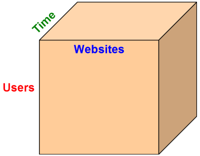


Some of our recent work on generative models for tensor decomposition:

- Interpretability: non-negativity & sparsity of the eigenvectors (latent factors)
- Simultaneous ranking/clustering of entities along with tensor decomposition
- Scalability for massive-sized tensors (computations scale w.r.t. nonzeros)
- Loss functions that model "rareness" in the data
- Online Learning

[ICML 2014, AAAI 2014, IJCAI 2015, ECML 2015, UAI 2015]

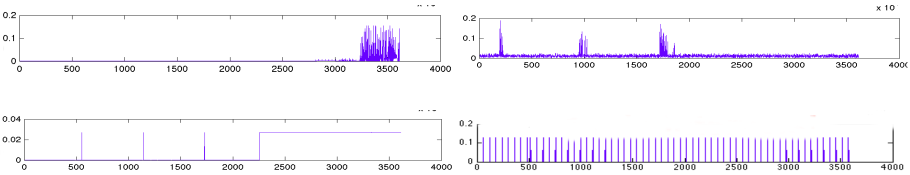
Example: Anomaly Detection



A look at what time-dimension eigenvectors might reveal

Top row: two eigenvectors capturing normal behaviors

Bottom row: two eigenvectors capturing bot-like behaviors



Can detect anomalous users by looking at the corresponding eigenvectors of users

Some Other Focus Areas

- Learning to jointly predict multiple (correlated) outputs

Some Other Focus Areas

- Learning to jointly predict multiple (correlated) outputs
- Predicting the arrival times (e.g., on an e-commerce portal)
  - When will the next visit happen?
  - How many visits between time t and $t + L$?
  - How to jointly learn arrival patterns of multiple customers?
  - All the above can be modeled using Point Processes

Some Other Focus Areas

- Learning to jointly predict multiple (correlated) outputs
- Predicting the arrival times (e.g., on an e-commerce portal)
  - When will the next visit happen?
  - How many visits between time t and $t + L$?
  - How to jointly learn arrival patterns of multiple customers?
  - All the above can be modeled using Point Processes
- Learning when data is continuously arriving (fast training, fast prediction)
  - How to learn and adapt to the "drifts" in the data distribution?

Some Other Focus Areas

- Learning to jointly predict multiple (correlated) outputs
- Predicting the arrival times (e.g., on an e-commerce portal)
  - When will the next visit happen?
  - How many visits between time t and $t + L$?
  - How to jointly learn arrival patterns of multiple customers?
  - All the above can be modeled using Point Processes
- Learning when data is continuously arriving (fast training, fast prediction)
  - How to learn and adapt to the "drifts" in the data distribution?
- Learning from mislabeled data (adversary might label some frauds as legit)

Some Other Focus Areas

- Learning to jointly predict multiple (correlated) outputs
- Predicting the arrival times (e.g., on an e-commerce portal)
  - When will the next visit happen?
  - How many visits between time t and $t + L$?
  - How to jointly learn arrival patterns of multiple customers?
  - All the above can be modeled using Point Processes
- Learning when data is continuously arriving (fast training, fast prediction)
  - How to learn and adapt to the "drifts" in the data distribution?
- Learning from mislabeled data (adversary might label some frauds as legit)
- How to leverage knowledge from other (related) data sources
  - Especially useful if we have a new learning problem with no training data
  - Algorithms: Transfer or Multitask Learning, Domain Adaptation

Thanks! Questions?