The Web of Cities and Mobility

Anastasios Noulas
Bruno Goncalves
Konstantinos Pelechrinis
Daniele Quercia

Figures used in the presentation are obtained from the cited publications
Objectives

- Point-out some of the open challenges in the field

- Introduce basic analytical techniques used to mine urban data

- Provide pointers to urban informatics studies that have relied on these techniques
  - More information can be found in these references
Challenges with urban data analysis

- **Urban data sources are highly heterogeneous**
  - Mutually reinforced knowledge

- **Current established data analysis and modeling techniques usually handle homogenous data**
  - Equally treating data from different sources can lead to suboptimal performance [1]
  - Heterogeneity increases also the dimensionality of the space to be explored → intensifies sparsity problems
Challenges with urban data analysis

● Many of the urban data can also be represented through multi-modal graphs
  ▪ Composite networks

● Different types of nodes and/or edges
  ▪ E.g., dwellers and locations
  ▪ Friendship and visitation relations
  ▪ Time evolution is also present

● Typical solution is projecting the multi-mode topology to one type of nodes → information loss
Challenges with urban data analysis

- Many of the urban data can also be represented through multi-modal graphs
  - Composite networks

- Different types of nodes and/or edges
  - E.g., dwellers and locations
  - Friendship and visitation relations

- Time evolution is also present

- Typical solution is projecting the multi-mode topology to one type of nodes → information loss
Other challenges include...

- **Data acquisition**
  - Quality and quantity
  - Privacy

- **Visualization of heterogeneous data is still an open challenge**

- **Any data-driven solution to urban problems relies on systems that connect the physical and digital space**
  - Design, implementation and deployment of such systems is much more challenging
Why tensors?

- Tensors are a generalization of a matrix and can capture a wealth of relationships
  - E.g., adding time evolution to traditional user-item relationships

- Different dimensions of a tensor can represent different types of nodes
  - Natural extensions of adjacency matrix for composite networks

- Coupled tensor-matrix can further model additional features of the entities and enhance/facilitate inference tasks
Tensors in urban informatics

- Taxi refueling behavior and city-wide gas consumption was studied using tensor analysis [2]
  - Three-mode tensor to model refueling events (RE)
    - Gas-station, day of the week, time-of-day (hourly granularity)

- Estimation of wait times in gas stations, total gas consumption, refueling recommendations, energy consumption analysis etc.
A fine-grained analysis of noise pollution in NYC was provided through tensor decomposition in [3]

- Three-mode tensor
  - Location, type of noise complaint from 311 data and time
Tensors in urban informatics

- Tensor decomposition was used in [4] to obtain real-time estimation of travel time of the road network
  - Three-dimensional tensor
    - Time, driver, and road segment
Data sparsity

- One of the problems when building the “urban” tensor is the fact that data can be sparse (i.e., missing)
  - Some areas might be underrepresented for 311 calls for noise complaints
    - Absence of a noise complaint does not mean there is not noise
  - GPS-enabled cars/taxis might not go through specific street segments and hence the corresponding entry at the tensor can be “zero”
    - This does not mean there is no traffic on this segment
  - Similar situation with gas-stations and GPS-enabled car-sensors
A typical approach used is to rely on matrix factorization methods and/or collaborative filter techniques to “fill in” the missing tensor values

- Exploit contextual information in a coupled with the tensor matrix
- Perform coupled tensor-matrix factorization

Matrix factorization is also useful when dealing with (sparse) urban data that can be represented as a matrix

- E.g., location-POIs category [5], dwellers-POIs [6] etc.
Matrix factorization

- Decompose the matrix of interest into a product of matrices
  - LU decomposition
  - QR decomposition
  - Cholesky decomposition
  - Eigendecomposition
  - Schur decomposition
  - SVD decomposition
  - CUR decomposition
  - Non-negative matrix factorization
  - ...

13
Singular Value Decomposition

- **Singular Value Decomposition (SVD)** is the most frequently used matrix factorization technique for collaborative filtering.

- SVD is also typically used for dimensionality reduction.
  - For instance let’s consider the following rank-2 matrix that represents some dataset:
    - We can use the two linear independent rows/columns to form a new base and describe the data using 2D vectors instead of 3D vectors.

\[
A = \begin{pmatrix}
2 & 4 & 2 \\
-2 & -3 & 1 \\
9 & 15 & 0
\end{pmatrix}
\]
Singular Value Decomposition

\[ A_{[m \times n]} = U_{[m \times r]} \Sigma_{[r \times r]} (V_{[n \times r]})^T \]

- **A**: Input (data) matrix
  - \( m \times n \) matrix (e.g., \( m \) dwellers, \( n \) POIs)

- **U**: Left singular vectors
  - \( m \times r \) matrix (e.g., \( m \) dwellers, \( r \) ‘urban concepts’)

- **\( \Sigma \)**: Singular values
  - \( r \times r \) diagonal matrix (strength of each ‘urban concept’)
    - \( r \): rank of matrix \( A \)

- **V**: Right singular vectors
  - \( n \times r \) matrix (\( n \) POIs, \( r \) ‘urban concepts’)

15
Singular Value Decomposition

\[ A = UΣV^T = \sum_i \sigma_i \vec{u}_i \odot \vec{v}_i^T \]
Singular Value Decomposition

\[ A = U \Sigma V^T = \sum_i \sigma_i \tilde{u}_i \circ \tilde{v}_i^T \]

\[ A = \begin{pmatrix} \sigma_1 \tilde{u}_1 \tilde{v}_1 \\ \sigma_2 \tilde{u}_2 \tilde{v}_2 \end{pmatrix} \]

\( \sigma_i \): scalar
\( \tilde{u}_i \): vector
\( \tilde{v}_i \): vector
SVD Theorem

- It is **always** possible to decompose a real matrix $A$ into $A = UΣV^T$, where:
  - $U$, $Σ$, $V$: unique
  - $U$, $V$: column orthonormal
    - $U^TU = I$ and $V^TV = I$
  - $Σ$: diagonal
    - Entries are positive and sorted in decreasing order ($σ_{11} ≥ σ_{22} ≥ ... ≥ 0$)
Example

Let’s consider 5 venues in a city and 8 city dwellers

- The dwellers-venues matrix $A$ captures the number of times a dweller has visited the corresponding venues.

$$
A = \begin{pmatrix}
5 & 5 & 0 & 0 & 0 & 0 & 0 \\
4 & 4 & 0 & 0 & 0 & 0 & 0 \\
5 & 5 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 4 & 4 & 0 & 0 & 0 \\
0 & 0 & 3 & 3 & 0 & 0 & 0 \\
0 & 0 & 4 & 4 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 5 & 5 & 0 \\
0 & 0 & 0 & 0 & 4 & 4 & 0 \\
\end{pmatrix}
$$
Example

\[ A = \begin{pmatrix} 5 & 5 & 0 & 0 & 0 & 0 & 0 \\ 4 & 4 & 0 & 0 & 0 & 0 & 0 \\ 5 & 5 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 4 & 4 & 0 & 0 & 0 \\ 0 & 0 & 3 & 3 & 0 & 0 & 0 \\ 0 & 0 & 4 & 4 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 5 & 5 \\ 0 & 0 & 0 & 0 & 4 & 4 & 0 \end{pmatrix} \]

\[ U = \begin{pmatrix} -0.6155 & 0 & 0 & 0 \\ -0.4924 & 0 & 0 & 0 \\ -0.6155 & 0 & 0 & 0 \\ 0 & -0.6217 & 0 & -0.5035 \\ 0 & -0.4663 & 0 & -0.3776 \\ 0 & -0.6293 & 0 & 0.7771 \\ 0 & 0 & -0.7809 & 0 \\ 0 & 0 & -0.6247 & 0 \end{pmatrix} \]

\[ \Sigma = \begin{pmatrix} 11.4891 & 0 & 0 & 0 \\ 0 & 9.0771 & 0 & 0 \\ 0 & 0 & 9.0554 & 0 \\ 0 & 0 & 0 & 0.7790 \end{pmatrix} \]

Coffee lovers
Mediterranean food
Asian food
Fast food

\[ V^T = \begin{pmatrix} -0.7071 & -0.7071 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -0.7054 & -0.7054 & 0 & 0 & -0.0693 \\ 0 & 0 & 0 & 0 & -0.7071 & -0.7071 & 0 \\ 0 & 0 & -0.0490 & -0.0490 & 0 & 0 & 0.9967 \end{pmatrix} \]
Matrix $U$ maps each user to the urban concept space

Matrix $V$ maps each venue to the urban concept space

Matrix $\Sigma$ represents the strength of each concept in our dataset

- The "Fast food" concept is rather weak in our dataset
  - This concept does not capture any interesting variation in the urban activity encoded by our dataset
- Do we really need to consider this concept? Can we reduce the dimensionality of the data?
Best low rank approximation

How do we do dimensionality reduction?

- By setting the k smallest singular values to 0
  - This essentially removes the contribution of the corresponding left and right singular vectors as well

\[ A \approx U \Sigma V^T \]

B is the best approximation of A (in the sense of the Frobenius norm)
Best low rank approximation

- **Theorem:** Let $A = U\Sigma V^T$ where $\Sigma: \sigma_1 \geq \sigma_2 \geq \ldots$, and rank$(A)=r$. Then $B=USV^T$ is the best rank-$k$ approximation to $A$ where: $S$ diagonal $r \times r$ matrix where $s_i=\sigma_i$ ($i=1,\ldots,k$) else $s_i=0$

- **What does “best” mean?**
  - B is a solution to: $\min_B \|A - B\|_F$ where rank$(B) = k$

- **How many singular values to keep?**
  - Heuristic: keep approximately 90% of the energy ($\sum \sigma_i^2$)
Low-rank approximation of urban concepts

- We can ignore the fourth concept since the singular value is small

\[
\begin{pmatrix}
5 & 5 & 0 & 0 & 0 & 0 & 0 \\
4 & 4 & 0 & 0 & 0 & 0 & 0 \\
5 & 5 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 4 & 4 & 0 & 0 & 0 \\
0 & 0 & 3 & 3 & 0 & 0 & 0 \\
0 & 0 & 4 & 4 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 5 & 5 & 0 \\
0 & 0 & 0 & 0 & 4 & 4 & 0 \\
\end{pmatrix}
\approx
\begin{pmatrix}
-0.6155 & 0 & 0 & 0 \\
-0.4924 & 0 & 0 & 0 \\
-0.6155 & 0 & 0 & 0 \\
0 & -0.6217 & 0 & 0 \\
0 & -0.4663 & 0 & 0 \\
0 & -0.6293 & 0 & 0 \\
0 & 0 & -0.7809 & 0 \\
0 & 0 & -0.6247 & 0 \\
\end{pmatrix}
\begin{pmatrix}
11.4891 & 0 & 0 & 0 \\
0 & 9.0771 & 0 & 0 \\
0 & 0 & 9.0554 & 0 \\
0 & 0 & 0 & 0 \\
\end{pmatrix}
\begin{pmatrix}
-0.7071 & -0.7071 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -0.7054 & -0.7054 & 0 & 0 & -0.0693 \\
0 & 0 & 0 & 0 & -0.7071 & -0.7071 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\end{pmatrix}
\]
> Urban concepts for POI recommendation

- Let’s assume that a user has visited “Costa café” 5 times and wants to know what other venues he might enjoy
  - We form a query vector: \( q^T = [0 \ 5 \ 0 \ 0 \ 0 \ 0 \ 0] \)

- We can project this query vector on the latent urban concept space using the right singular vectors
  - Venue-to-urban concept matrix \( V \)
  - \([q^TV_1 \ q^TV_2 \ q^TV_3] = [-3.5355 \ 0 \ 0] \)
    - This dweller *likes* the first urban concept (i.e., coffee shops)
    - Using this fact we can now perform POI recommendation for this dweller (e.g., by identifying venues that *belong* to the first urban concept)
Which city is this?
Now...?
OK…how about now?
I guess now you know…

Reconstructed image from first 60 singular values
Compare with the original
Reconstruction error

![Graph showing the relationship between singular value and reconstruction error. The x-axis represents the number of singular values used, ranging from 0 to 100, while the y-axis represents the reconstruction error, ranging from 0 to 2500. The graph includes a downward trend indicating a decrease in reconstruction error as more singular values are used.](image-url)
Other latent factorization methods

- Matrix factorization is not the only method to identify latent urban patterns
- Graphical models are also useful. E.g.,
  - Latent Dirichlet Allocation (LDA) [8]

Cranshaw and Yano [7]
Other latent factorization methods

- Matrix factorization is not the only method to identify latent urban patterns
- Graphical models are also useful. E.g.,
  - Dirichlet Multinominal Regression [9] with LDA

Yuan et al. [5]
Other latent factorization methods

- SVD is also not the only matrix factorization approach to identify latent urban factors/concepts
  - Poussevin et al. [11] utilize non-negative matrix factorization to identify latent patterns in urban transportation data
    - Data are represented through a passenger-time matrix $X$, that captures the probability of public transportation usage in the specific time-slot
      - Weekly periodicity is assumed and hence, data are aggregated in a “representative” week

$$\mathcal{L}(X, A, D) = \frac{1}{m} \|X - A.D\|^2 + \lambda |A|$$

$$C(D) : D \geq 0, \forall i, \sum_{j < t_{day}} D_{ij} = 1, \sum_{t_{day} \leq j < t_{week}} D_{ij} = 1$$
Beyond 2-dimensional information

- More detailed models can be built if we have information beyond simple counts of location visitations or ratings.

- For example, in the last scenario the public transport station was not part of the factorization process!
  - This can be represented either through a three-way array (i.e., tensor) or through two coupled matrices.

- In general, additional contextual information can be utilized to deal with sparse/missing data and identify more accurate latent urban patterns.
Coupled matrix factorization

- Better modeling of latent urban factors can be achieved if we have additional information to the dweller-location/venues matrix

- For instance, matrix A can be driven by check-ins data
  - However, we might have another matrix B that includes additional information for the venues from Yelp
    - E.g., type of venue, rating, price range etc.

- A and B are coupled in the spatial dimension and we could use the additional information provided in matrix B to complete matrix A and/or provide recommendations
Coupled matrix factorization

- This low-rank factorization will allow us to reconstruct the matrix $A$
  - Our objective is to minimize the reconstruction error
    - We factor $A$ as $XY^T$ and $B$ as $XZ^T$
      - Common factor $X \rightarrow$ assumption of common low-rank space
    - Regularization to avoid overfitting,
      $$\min_{X,Y,Z} a \left\| A - XY^T \right\|_F^2 + b \left\| B - XZ^T \right\|_F^2$$
      $$\min_{X,Y,Z} a \left\| A - XY^T \right\|_F^2 + b \left\| B - XZ^T \right\|_F^2 + \lambda_x \left\| X \right\|_F^2 + \lambda_y \left\| Y \right\|_F^2 + \lambda_z \left\| Z \right\|_F^2$$
In [6] the authors use a coupled matrix factorization technique to provide POI and urban activity recommendations

- X is a location-activity matrix $\Rightarrow$ frequency of activities in locations (location here represents a geographic location)
- Y is location-category of POIs
- Z captures the correlations between two different activities
A tensor is a generalization of a matrix to higher dimensions

For instance, for 3 dimensions, tensor $X$ is a three-way array that describes the dataset and is indexed by three indices $(i,j,k) \rightarrow X(i,j,k)$

Generalizing of the notion of rank

For two vectors $a$ ($I \times 1$) and $b$ ($J \times 1$), the outer product $\circ$ is an $I \times J$ rank-one matrix with $(i,j)$-th element $a(i)b(j)$ ($a \circ b = ab^T$)

For three vectors $a$ ($I \times 1$), $b$ ($J \times 1$) and $c$ ($K \times 1$), $a \circ b \circ c$ is an $I \times J \times K$ rank-one three-way array (i.e., tensor) with $(i,j,k)$-th element $a(i)b(j)c(k)$

The rank of a tensor $X$ is the smallest number of outer products needed to synthesize $X$
Can we use techniques that generalize matrix factorization to tensors for analyzing multi-dimensional (and/or temporal) urban data?

- Yes!
PARAFAC decomposition

- PARAFAC decomposition is a generalization of SVD for tensors

- PARAFAC decomposes tensor $X$ to a sum of rank-one tensors:

$$X \approx \sum_{f=1}^{F} a_f \circ b_f \circ c_f$$

  - Each component is essentially a triplet of vectors each of which corresponds to one of the tensor dimensions
  - Each such triplet represents a latent urban pattern in our data
PARAFAC decomposition example

- Insight on the ability of tensors to simplify urban pattern mining
  - Data can be summarized into a set of intuitively interpretable pieces

- Foursquare checkins between September 2010 – January 2011 [12]
  - User, location and time
  - The presence of time allows to identify temporal dynamics of the urban pulse
PARAFAC decomposition example

Exemplary latent urban patterns [13]
Further processing the raw urban latent patterns can reveal a wealth of information:

- E.g., matrices \( A \) and \( B \) are constructed from the factor vectors \( a_f \) and \( b_f \) as columns – i.e., they provide a low-rank embedding of the users and venues in the latent space.
- We can then cluster users and/or venues using the rows of these matrices.
  - Loosely this is a generalization of spectral clustering.
Processing tensor components

- One can also cluster the components directly using as features the vectors \( a, b \) and \( c \)
  - Group similar patterns to identify proto-patterns

- In general, tensor decomposition can generalize spectral techniques that have been used to identify urban and behavioral patterns

Calabrese et al. [14]
Processing tensor components

- One can also cluster the components directly using as features the vectors a, b and c
  - Group similar patterns to identify proto-patterns

- In general, tensor decomposition can generalize spectral techniques that have been used to identify urban and behavioral patterns

Eagle et al. [15]
Processing tensor components

- One can also cluster the components directly using as features the vectors $a$, $b$ and $c$
  - Group similar patterns to identify proto-patterns

- In general, tensor decomposition can generalize spectral techniques that have been used to identify urban and behavioral patterns

Cranshaw et al. [16]
Intuition behind tensor decomposition

- Tensor decomposition attempts to summarize the given data into a reduced rank representation

- Dense groups that associate all dimensions of the data are favored during this process
  - These groups need not be immediately visible by inspection of the data
    - **PARAFAC is not affected by permutations of mode indices**
  - We expect near-bipartite cores starting from the most dense all the way to the sparsest
    - E.g., people who go to specific locations at specific times
Assessing the model quality

- As with every model, the performance of PARAFAC decomposition can range from perfectly capturing the data, to performing rather poorly.

- The main question with regards to the quality of modeling is whether the data are amenable to PARAFAC decomposition and to what extent.

- CORCONDIA [17] is a diagnostic tool that serves as an indicator on whether the PARAFAC model describes the data well.
Assessing the model quality

• If CORCONDIA gives a bad score, this could be caused either because
  ▪ the chosen rank $F$ is not appropriate or
  ▪ the data cannot be modeled well with PARAFAC – irrespective of the rank
    ✓ Incrementally increasing the rank can provide us with a choice of a good value for the rank to be used

• **Open challenge**: computationally expensive metric!
Other tensor decomposition methods

- **PARAFAC is not the only tensor decomposition method**
  - However, under certain conditions PARAFAC decomposition is unique, which makes it appealing.

- **An alternative tensor decomposition is Tucker [18]** (HOSVD [19])
  - Non-unique except for very special cases
  - HOSVD has been used in [20] to provide recommendations of friends, locations and activities using spatial datasets

- **Rule of thumb:** Use Tucker/HOSVD for subspace estimation (e.g., compression applications) and PARAFAC for latent parameter estimation.
Coupled matrix-tensor decomposition

- Additional contextual information associated with one (or more) of the tensor dimensions can be represented through a matrix $Y$
  - For example, matrix $Y_v$ can capture information about the locations in a city and matrix $Y_u$ can capture demographic information about the users

- In general, a matrix $Y$ and a tensor $T$ are coupled if they share a common mode
  - An $n$-dimensional tensor can be coupled with at most $n$ matrices $Y_i$
Coupled matrix-tensor factorization

- We can decompose the tensor and the coupled matrices simultaneously

- The idea behind this joint factorization is to decompose $I$ and $Y_i$ to latent factors that are coupled in the shared dimension
  - Low dimensional embedding of the data in the common contextual subspace
    - Useful for predictive tasks
Coupled matrix-tensor factorization

- The CMTF is the solution to the following optimization problem:
  \[
  \min_{A_1,...,A_n,D_1,...,D_n} \| T - \sum_k a_{1k} \circ a_{2k} \circ ... \circ a_{nk} \|^2_F + \| Y_1 - A_1 D_1^T \|^2_F + ... + \| Y_n - A_n D_n^T \|^2_F
  \]
  - \( a_{ik} \) is the \( k^{th} \) column of \( A_i \)

- We can also add regularization terms for \( A_i \) and \( D_i \) to avoid over-fitting

- CMTF is especially useful when the tensor is extremely sparse and hence, pure PARAFAC cannot accurately reconstruct the original tensor and/or fill missing values
In [21] the authors provide an urban activity recommendation engine by utilizing CMTF

- Matrix **C**: each row captures the distribution of POIs at a location
- Matrix **D**: activity correlations (obtained through web-search engines)
- Matrix **B**: user-user similarity – e.g., via demographic information about dwellers
- Matrix **E**: dweller – location *transactions*
CMTF for location activity recommendation

- $X$, $Y$, $Z$ and $V$ are the user, location, activity and location features latent factor matrices

$$\mathcal{L}(X, Y, Z, V) = \sum_{(i,j,k)\in D_A} \left( \sum_{l=1}^{d} x_{il} y_{jl} z_{kl} - A_{ijk} \right)^2 + \lambda_1 \sum_{(i,j)\in D_B} B_{ij} \|x_i - x_j\|^2 + \lambda_2 \sum_{(j,l)\in D_C} (y_j \cdot v_l - C_{jl})^2 + \lambda_3 \sum_{(k,l)\in D_D} D_{kl} \|z_k - z_l\|^2 + \lambda_4 \sum_{(i,j)\in D_E} (x_i \cdot y_j - E_{ij})^2 + \lambda_5 (\|X\|^2 + \|Y\|^2 + \|Z\|^2 + \|V\|^2).$$

- **PARAFAC decomposition**
- **User regularization**
- **Activity regularization**
- **Coupled decomposition of $A$ and $C$**
- **Coupled decomposition of $A$ and $E$**
- **Regularization**
Zheng et al. [3] use 311 data from NYC to analyze and model the noise pollution of the city.
CMTF for urban noise modeling

- **Matrix X** captures features of the different regions considered
  - Two types of features: POIs and road network
- **Matrix Z** captures (spatial) correlations between different noise categories
  - Learnt from 311 data
- **Matrix Y** captures dweller mobility features
  - Check-in data
Open challenges

- **Online tensor decomposition for real-time urban applications**

- **Automated ways to classify latent patterns**
  - E.g., anomalous or regular

- **Beyond matrices and tensors**
  - Visualization is crucial for these applications – especially for outreach to the involved parties
  - Data collections
    - Noise data, privacy etc.
References

References

[20] Symeonidis et al., “Geo-social recommendations based on incremental tensor reduction and local path traversal”, in ACM SIGSPATIAL workshops, 2011

A recent survey on urban computing can be found at:


Acknowledgements

Evangelos Papalexakis

Theodoros Lappas

Aristides Gionis

Christos Faloutsos

Evimaria Terzi

Michalis Raptis
Thank you!