Bayesian Nonparametrics
+
Graph Neural Networks

Matt Gormley
Lecture 25
Dec. 7, 2022
Reminders

• 10-618 Mini-Project
  – Team Formation Due: Tue, Nov 29
  – Proposal Due: Thu, Dec 1
  – Summary & Code Due: Fri, Dec 9

• Practice Problems 2
  – Out: Wed, Dec 8

• Exam 2:
  – Thu, Dec 15, 5:30 – 7:30 PM
Chinese Restaurant Process & Stick-breaking Constructions

DIRICHLET PROCESS
Dirichlet Process

Ferguson Definition

- Parameters of a DP:
  1. Base distribution, $H$, is a probability distribution over $\Theta$
  2. Strength parameter, $\alpha \in \mathcal{R}$

- We say $G \sim \text{DP}(\alpha, H)$ if for any partition $A_1 \cup A_2 \cup \ldots \cup A_K = \Theta$ we have:
  
  $$(G(A_1), \ldots, G(A_K)) \sim \text{Dirichlet}(\alpha H(A_1), \ldots, \alpha H(A_K))$$

In English: the DP is a distribution over probability measures s.t. marginals on finite partitions are Dirichlet distributed
Chinese Restaurant Process

• Imagine a Chinese restaurant with an infinite number of tables
• Each customer enters and sits down at a table
  – The first customer sits at the first unoccupied table
  – Each subsequent customer chooses a table according to the following probability distribution:

\[ p(k\text{th occupied table}) \propto n_k \]
\[ p(\text{next unoccupied table}) \propto \alpha \]

where \( n_k \) is the number of people sitting at the table \( k \)
DIRICHLET PROCESS MIXTURE MODEL
CRP Mixture Model

• Draw n cluster indices from a CRP:
  \( z_1, z_2, \ldots, z_n \sim CRP(\alpha) \)
• For each of the resulting K clusters:
  \( \theta_k^* \sim H \)
  where \( H \) is a base distribution
• Draw n observations:
  \[ x_i \sim p(x_i \mid \theta^*_{z_i}) \]

Customer \( i \) orders a dish \( x_i \) (observation) from a table-specific distribution over dishes \( \theta_k^* \) (cluster parameters)

(color denotes different values of \( x_i \))
CRP Mixture Model

- Draw n cluster indices from a CRP: 
  \( z_1, z_2, \ldots, z_n \sim CRP(\alpha) \)
- For each of the resulting K clusters: 
  \( \theta^*_k \sim H \)
  where \( H \) is a base distribution
- Draw n observations: 
  \( x_i \sim p(x_i \mid \theta^*_z) \)

- The Gibbs sampler is easy thanks to exchangeability
- For each observation, we remove the customer / dish from the restaurant and resample as if they were the last to enter
- If we collapse out the parameters, the Gibbs sampler draws from the conditionals:
  \[ z_i \sim p(z_i \mid z_{-i}, x) \]

(color denotes different values of \( x_i \))
Overview of 3 Gibbs Samplers for Conjugate Priors

- **Alg. 1: (uncollapsed)**
  - Markov chain state: per-customer parameters $\theta_1, \ldots, \theta_n$
  - For $i = 1, \ldots, n$: Draw $\theta_i \sim p(\theta_i \mid \theta_{-i}, x)$

- **Alg. 2: (uncollapsed)**
  - Markov chain state: per-customer cluster indices $z_1, \ldots, z_n$ and per-cluster parameters $\theta_1^*, \ldots, \theta_k^*$
  - For $i = 1, \ldots, n$: Draw $z_i \sim p(z_i \mid z_{-i}, x, \theta^*)$
  - Set $K = \text{number of clusters in } z$
  - For $k = 1, \ldots, K$: Draw $\theta_k^* \sim p(\theta_k^* \mid \{x_i : z_i = k\})$

- **Alg. 3: (collapsed)**
  - Markov chain state: per-customer cluster indices $z_1, \ldots, z_n$
  - For $i = 1, \ldots, n$: Draw $z_i \sim p(z_i \mid z_{-i}, x)$
CRP Mixture Model

• Q: How can the Alg. 2 Gibbs samplers permit an infinite set of clusters in finite space?

• A: Easy!
  – We are only representing a finite number of clusters at a time – those to which the data have been assigned
  – We can always bring back the parameters for the “next unoccupied table” if we need them
Dirichlet Process: For both the CRP and stick-breaking constructions, if we marginalize out $G$, we have the following predictive distribution:

$$
\theta_{n+1}|\theta_1, \ldots, \theta_n \sim \frac{1}{\alpha + n} \left( \alpha H + \sum_{i=1}^{n} \delta_{\theta_i} \right)
$$

(Blackwell-MacQueen Urn Scheme)

The **Chinese Restaurant Process Mixture Model** is just a different construction of the **Dirichlet Process Mixture Model** where we have marginalized out $G$.
Graphical Models for DPMMs

The Pólya urn construction

The Stick-breaking construction
Example: DP Gaussian Mixture Model

Figure 2: The approximate predictive distribution given by variational inference at different stages of the algorithm. The data are 100 points generated by a Gaussian DP mixture model with fixed diagonal covariance.
Example: DP Gaussian Mixture Model

Figure 3: Mean convergence time and standard error across ten data sets per dimension for variational inference, TDP Gibbs sampling, and the collapsed Gibbs sampler.
GMM VS. DPMM EXAMPLE
Example: Dataset
Example: GMM

Clustering with GMM (k=6, init=random, cov=full, iter=0)
Example: GMM

Clustering with GMM (k=6, init=random, cov=full, iter=5)
Example: GMM

Clustering with GMM (k=6, init=random, cov=full, iter=10)
Example: GMM

Clustering with GMM (k=6, init=random, cov=full, iter=15)
Example: GMM

Clustering with GMM (k=6, init=random, cov=full, iter=20)
Example: GMM

Clustering with GMM (k=6, init=random, cov=full, iter=25)
Example: GMM

Clustering with GMM (k=6, init=random, cov=full, iter=30)
Example: GMM

Clustering with GMM (k=6, init=random, cov=full, iter=35)
Example: GMM

Clustering with GMM (k=6, init=random, cov=full, iter=39)
Example: DPMM

Clustering with DPMM (k=6, init=random, cov=full, iter=0)
Example: DPMM

Clustering with DPMM (k=6, init=random, cov=full, iter=1)
Example: DPMM

Clustering with DPMM (k=6, init=random, cov=full, iter=2)
Example: DPMM

Clustering with DPMM (k=6, init=random, cov=full, iter=3)
Example: DPMM

Clustering with DPMM (k=6, init=random, cov=full, iter=4)
Example: DPMM

Clustering with DPMM (k=6, init=random, cov=full, iter=5)
Example: DPMM

Clustering with DPMM (k=6, init=random, cov=full, iter=6)
Example: DPMM

Clustering with DPMM (k=6, init=random, cov=full, iter=7)
Example: DPMM

Clustering with DPMM (k=6, init=random, cov=full, iter=8)
Example: DPMM

Clustering with DPMM (k=6, init=random, cov=full, iter=9)
Example: DPMM

Clustering with DPMM (k=6, init=random, cov=full, iter=10)
Example: DPMM

Clustering with DPMM (k=6, init=random, cov=full, iter=11)
Example: DPMM

Clustering with DPMM (k=6, init=random, cov=full, iter=12)
Example: DPMM

Clustering with DPMM (k=6, init=random, cov=full, iter=13)
Example: DPMM

Clustering with DPMM (k=6, init=random, cov=full, iter=14)
Example: DPMM

Clustering with DPMM (k=6, init=random, cov=full, iter=15)
Example: DPMM

Clustering with DPMM (k=6, init=random, cov=full, iter=16)
Example: DPMM

Clustering with DPMM (k=6, init=random, cov=full, iter=17)
Example: DPMM

Clustering with DPMM (k=6, init=random, cov=full, iter=18)
Example: DPMM

Clustering with DPMM (k=6, init=random, cov=full, iter=19)
Example: DPMM

Clustering with DPMM (k=6, init=random, cov=full, iter=20)
Example: DPMM

Clustering with DPMM (k=6, init=random, cov=full, iter=21)
Example: DPMM

Clustering with DPMM (k=6, init=random, cov=full, iter=22)
Example: DPMM

Clustering with DPMM (k=6, init=random, cov=full, iter=23)
Example: DPMM

Clustering with DPMM (k=6, init=random, cov=full, iter=24)
Example: DPMM

Clustering with DPMM (k=6, init=random, cov=full, iter=25)
Summary of DP and DP-MM

• **DP** has many **different representations:**
  – Chinese Restaurant Process
  – Stick-breaking construction
  – Blackwell-MacQueen Urn Scheme
  – Limit of finite mixtures
  – etc.

• These representations give rise to a variety of **inference techniques** for the **DP-MM** and related models
  – Gibbs sampler (CRP)
  – Gibbs sampler (stick-breaking)
  – Variational inference (stick-breaking)
  – etc.
HIERARCHICAL DIRICHLET PROCESS (HDP)
Related Models

• Hierarchical Dirichlet Process Mixture Model (HDP-MM)
• Infinite HMM
• Infinite PCFG
HDP-MM

- In LDA, we have $M$ independent samples from a Dirichlet distribution.
- The weights are different, but the topics are fixed to be the same.
- If we replace the Dirichlet distributions with Dirichlet processes, each atom of each Dirichlet process will pick a topic independently of the other topics.
- Because the base measure is continuous, we have zero probability of picking the same topic twice.
- If we want to pick the same topic twice, we need to use a discrete base measure.
- For example, if we chose the base measure to be
  \[ H = \sum_{k=1}^{K} \alpha_k \delta_{\beta_k} \]  
  then we would have LDA again.
- We want there to be an infinite number of topics, so we want an infinite, discrete base measure.
- We want the location of the topics to be random, so we want an infinite, discrete, random base measure.
Hierarchical Dirichlet process:

\[ G_0 | \gamma, H \sim DP(\gamma, H) \]
\[ G_j | \alpha, G_0 \sim DP(\alpha, G_0) \]
\[ \theta_{ji} | G_j \sim G_j \]
HDP-MM

Figure 6: (Left) Comparison of latent Dirichlet allocation and the hierarchical Dirichlet process mixture. Results are averaged over 10 runs; the error bars are one standard error. (Right) Histogram of the number of topics for the hierarchical Dirichlet process mixture over 100 posterior samples.
HDP-HMM (Infinite HMM)

Number of hidden states in Infinite HMM is countably infinite

Figure 9: A hierarchical Bayesian model for the infinite hidden Markov model.

Figure 10: Comparing the infinite hidden Markov model (solid horizontal line) with ML, MAP and VB trained hidden Markov models. The error bars represent one standard error (those for the HDP-HMM are too small to see).
HDP-PCFG (Infinite PCFG)

HDP-PCFG

\[ \beta \sim \text{GEM}(\alpha) \]  
[draw top-level symbol weights]

For each grammar symbol \( z \in \{1, 2, \ldots \} \):

\[ \phi_z^T \sim \text{Dirichlet}(\alpha^T) \]  
[draw rule type parameters]

\[ \phi_z^E \sim \text{Dirichlet}(\alpha^E) \]  
[draw emission parameters]

\[ \phi_z^B \sim \text{DP}(\alpha^B, \beta \beta^T) \]  
[draw binary production parameters]

For each node \( i \) in the parse tree:

\[ t_i \sim \text{Multinomial}(\phi_{zi}^T) \]  
[choose rule type]

If \( t_i = \text{EMISSION} \):

\[ x_i \sim \text{Multinomial}(\phi_{zi}^E) \]  
[emit terminal symbol]

If \( t_i = \text{BINARY-PRODUCTION} \):

\[ (z_{L(i)}, z_{R(i)}) \sim \text{Multinomial}(\phi_{zi}^B) \]  
[generate children symbols]

\[ \beta \sim \text{GEM}(\alpha) \]

\[ \beta \beta^T \]

\[ \phi_z^B \sim \text{DP}(\beta \beta^T) \]
## Parametric vs. Nonparametric

<table>
<thead>
<tr>
<th>Type of Model</th>
<th>Parametric Example</th>
<th>Nonparametric Example</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Construction #1</td>
</tr>
<tr>
<td>distribution over</td>
<td>Dirichlet-Multinomial Model</td>
<td>Dirichlet Process (DP)</td>
</tr>
<tr>
<td>counts</td>
<td></td>
<td>Chinese Restaurant Process (CRP)</td>
</tr>
<tr>
<td>mixture</td>
<td>Gaussian Mixture Model (GMM)</td>
<td>Dirichlet Process Mixture Model (DPMM)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CRP Mixture Model</td>
</tr>
<tr>
<td>admixture</td>
<td>Latent Dirichlet Allocation (LDA)</td>
<td>Hierarchical Dirichlet Process Mixture Model (HDPMM)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Chinese Restaurant Franchise</td>
</tr>
</tbody>
</table>
Summary of DP and DP-MM

• **DP** has many **different representations:**
  – Chinese Restaurant Process
  – Stick-breaking construction
  – Blackwell-MacQueen Urn Scheme
  – Limit of finite mixtures
  – etc.

• These representations give rise to a variety of **inference techniques** for the **DP-MM** and related models
  – Gibbs sampler (CRP)
  – Gibbs sampler (stick-breaking)
  – Variational inference (stick-breaking)
  – etc.
GRAPH NEURAL NETWORKS
Background: Graphs

- **Def:** a graph $G = (V,E)$ consists of vertices $V$ and edges $E$
  - vertices are also called nodes
  - Let **node** $v_i \in V$ and $|V| = N$
  - Let **edge** $(v_i, v_j) \in E$ and $|E| = M$

- **Def:** an **adjacency matrix** $A$ for graph $G$ is a binary matrix such that:
  - $A_{i,j} = 1$ if $(v_i, v_j) \in E$
  - $A_{i,j} = 0$ if $(v_i, v_j) \not\in E$

- **Def:** an **adjacency list** is simply an ordered version of the set of edges, e.g. list$(E)$

- **Def:** the **neighbors** $N(v_j)$ of a **node** $v_j$ are all nodes $v_i$ such that $(v_i, v_j) \in E$
The graph we just defined is a **directed graph** because each edge \((v_i, v_j) \in E\) is an **ordered pair**.

For an undirected graph: 

\[(v_i, v_j) \in E \Rightarrow (v_j, v_i) \in E\] 

each undirected edge is just two directed edges.

An **undirected graph** is a special case in which the adjacency matrix is **symmetric**.
Background: Graphs

- The graph we just defined is a **directed graph** because each edge \((v_i, v_j) \in E\) is an **ordered pair**
- **Def**: a **self-loop** \((v_i, v_i) \in E\) is an edge from a node to itself
- A self-loop corresponds to a diagonal entry in the adjacency matrix

- For an undirected graph: \((v_i, v_j) \in E \implies (v_j, v_i) \in E\) each undirected edge is just two directed edges
- An **undirected graph** is a special case in which the adjacency matrix is **symmetric**
- (An undirected self-loop is only one directed edge)
Data as Graphs

- citation networks
Data as Graphs

• molecules

Figure from https://chemrxiv.org/engage/chemrxiv/article-details/6246035d3affe4aa143c3848
Data as Graphs

• semantic parsing

Figure from https://ai.googleblog.com/2017/11/sling-natural-language-frame-semantic.html
Data as Graphs

- social networks

Figure from https://distill.pub/2021/gnn-intro/
Data as Graphs

• images

In (b), above, the original image (a) has been segmented into five entities: each of the fighters, the referee, the audience and the mat. (C) shows the relationships between these entities.

Figure from https://distill.pub/2021/gnn-intro/
Graph Neural Networks

Decomposition of tasks for GNNs

• **Node-level**
  – *node classification*: predict a label for each node
  – *node regression*: predict a value for each node

• **Edge-level**
  – *edge classification*: predict a label for each edge
  – *link prediction*: predict presence/absence/strength of an edge

• **Graph-level**
  – *graph classification*: predict a label for the entire graph
  – *graph regression*: predict a value for the entire graph
Types of GNNs

A Taxonomy of Graph Neural Networks (GNNs) from Wu et al. (2020):

1. Recurrent GNNs
2. Convolutional GNNs
   a. Spectral-based
   b. Spatial-based
3. Graph autoencoders
   a. for network embedding
   b. for graph generation
4. Spatial-temporal GNNs

Node and Edge Representations

- **Def:** each node \( v \) has a **node feature vector** \( \mathbf{x}_v \in \mathbb{R}^M \)
- **Def:** each edge \( e \) has an **edge feature vector** \( \mathbf{x}_e \in \mathbb{R}^{M'} \)
- For undirected graphs, we (usually) assume there is only **one vector per undirected edge** (i.e. not one for each of the two corresponding directed edges)

| \( \mathbf{x}_A \) | 0.1 | -7 | ... | 2 |
| \( \mathbf{x}_B \) | 4 | -2 | ... | 3 |
| \( \mathbf{x}_C \) | -5 | 1 | ... | 1 |
| \( \mathbf{x}_D \) | 0 | 3 | ... | 0.5 |
| \( \mathbf{x}_E \) | 0.6 | -3 | ... | 0.1 |

| \( \mathbf{x}_{A,B} \) | 2 | 0.4 | ... | -2 |
| \( \mathbf{x}_{A,D} \) | 0.3 | 0.1 | ... | -0.5 |
| \( \mathbf{x}_{A,E} \) | -5 | 1 | ... | 4 |
| \( \mathbf{x}_{B,D} \) | 0.9 | 9 | ... | -9 |
| \( \mathbf{x}_{B,E} \) | 1 | -4 | ... | 7 |
| \( \mathbf{x}_{C,E} \) | 6 | -2 | ... | 0 |
RECURRENT GRAPH NEURAL NETWORKS
Recurrent GNNs

• Some of the early GNNs capitalized on acyclic graphs (or acyclic substructure of graphs)

• This is akin to how Loopy Belief Propagation and Tree Reweighted Belief Propagation (two variational message passing techniques that came long before) are implemented

• The backbone of these Recurrent GNNs was a variant of the LSTM
Tree LSTMs

- **Two types:**
  - Child-Sum TreeLSTM (handles binary trees)
  - N-ary TreeLSTM (handles arbitrary trees)
- **Key insight:**
  - generalize the LSTM from chains to trees
  - the **hidden unit** for a non-terminal node is a **parameterized function of its children**

```
NP    VP
W_{DT,NN}   W_{V,NN}
```

```
The [movie] showed [peace]
```
Tree LSTMs

Standard LSTM on a chain

Tree LSTM on an n-ary tree

Graph LSTMs

- The Graph LSTM (Peng et al., 2017) decomposes a directed cyclic graph into two directed acyclic graphs.
- The computation graph first runs a TreeLSTM left-to-right along the first acyclic graph, then right-to-left through the second acyclic graph.

All patients were treated with gefitinib and showed a partial response.

Figures from [https://aclanthology.org/Q17-1008](https://aclanthology.org/Q17-1008)
SPATIAL GRAPH NEURAL NETWORKS
Spatial Graph Neural Networks

Whiteboard:

– Basic node-only GNN
– Basic neighbor-only GNN
– Visualizing the k-hop neighborhood computation graph
– Incorporating self-loops
– Normalization techniques
– Adding edge features