Inference and Introspection in Deep Generative Models of Sparse Data

Rahul Krishnan
NYU

Matthew D. Hoffman
Adobe Research

(Presenting)
Background: Variational Autoencoders (VAEs)
VAEs/DLGMs are almost exclusively applied to images, not text or other high-dimensional, sparse data (some exceptions: Miao et al., 2016; Bowman et al., 2016).

Why? Sparse "texty" data is everywhere! (Documents, social networks, ratings/views/listens, medical diagnoses, etc., etc.)
Our Hypothesis: Local Optima

Many observed words are rare.
The inference network needs to see each rare word a few times to learn to interpret that word.
But if the inference network's inferences are bad, the generative network's gradient signal will be bad.
Our Approach

Use the inference network as an initializer and optimize from there. (Cf. Hjelm et al. 2016)
Experimental Results: Held-Out Perplexity

Deeper is better for large datasets.

TF-IDF often helps, never hurts.

Secondary optimization *at training time* always helps deeper models.

Secondary optimization *at test time* is very important.

Come to the poster for details.
Introspection and Interpretability

What makes "shallow" models (e.g., LDA) interpretable?

A hypothesis: It's because the parameters encode a linear relationship between latent vectors $z$ and observations $x$:

$$
\mathbb{E}[x|z] = \theta z; \quad \frac{\partial \mathbb{E}[x|z]}{\partial z} = \theta.
$$
Introspection and Interpretability

What makes "shallow" models (e.g., LDA) interpretable?

A hypothesis: It's because the parameters encode a linear relationship between latent vectors $z$ and observations $x$:

$$E[x|z] = \theta z; \quad \frac{\partial E[x|z]}{\partial z} = \theta.$$

For nonlinear models this Jacobian depends on $z$, so we average over samples of $z$:

$$\mathcal{J}_{\text{prob}}^{\text{mean}} \triangleq \int_z p(z) \frac{\partial E[x|z]}{\partial z} dz; \quad \mathcal{J}_{\text{log}}^{\text{mean}} \triangleq \int_z p(z) \frac{\partial \log E[x|z]}{\partial z} dz.$$
Diagnosing Pruning

The singular value spectrum of $J$ tells us how many latent dimensions are being used.

We find that secondary optimization dramatically reduces overpruning in deep models.
**Word Embeddings**

We can use the rows of $J$ as word embeddings, with sensible results:

<table>
<thead>
<tr>
<th>Query</th>
<th>Neighborhood</th>
</tr>
</thead>
<tbody>
<tr>
<td>intelligence</td>
<td>espionage, secrecy, interrogation</td>
</tr>
<tr>
<td>zen</td>
<td>dharma, buddhism, buddhas, meditation</td>
</tr>
<tr>
<td>artificial</td>
<td>artificially, molecules, synthetic, soluble</td>
</tr>
<tr>
<td>military</td>
<td>civilian, army, commanders, infantry</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Query</th>
<th>Neighborhood</th>
</tr>
</thead>
<tbody>
<tr>
<td>Superman II</td>
<td>Superman: The Movie, Superman III, Superman IV: The Quest for Peace</td>
</tr>
<tr>
<td>Casablanca</td>
<td>Citizen Kane, The Treasure of the Sierra Madre, Working with Orson Welles, The Millionairess</td>
</tr>
<tr>
<td>The Princess Bride</td>
<td>The Breakfast Club, Sixteen Candles, Groundhog Day, Beetlejuice</td>
</tr>
<tr>
<td>12 Angry Men</td>
<td>To Kill a Mockingbird, Rear Window, Mr. Smith Goes to Washington, Inherit the Wind</td>
</tr>
</tbody>
</table>
Contextual Word Embeddings

Using the Jacobian matrix for a particular $z$ gives context-dependent embeddings:

<table>
<thead>
<tr>
<th>Word</th>
<th>Context</th>
<th>Neighboring Words</th>
</tr>
</thead>
<tbody>
<tr>
<td>crane</td>
<td>construction, bird</td>
<td>lifting, usaa, spanned, crushed, lift erected, parkland, locally, farmland</td>
</tr>
<tr>
<td>bank</td>
<td>river, money</td>
<td>watershed, footpath, confluence, drains banking, government, bankers, comptroller</td>
</tr>
<tr>
<td>fires</td>
<td>burn, layoff</td>
<td>ignition, combustion, engines, fuel, engine thunderstorm, grassy, surrounded, walkway</td>
</tr>
</tbody>
</table>
Summary

We addressed some issues with fitting VAEs to sparse, high-dimensional "texty" data.

We proposed a simple way to examine what deep models of text are learning.

Come to the poster and let's talk about details, extensions, and applications!