Wikipedia Traffic Forecasting with Graph Embeddings
Anthony Miyaguchi, Shaon Chakrabarti, and Nicolai Garcia
CS229 - Fall 2019

Summary
We build a model to forecast the number of visits to a random, connected subset of Wikipedia articles over a 7-day window.
- We mine unsupervised features from article hyperlinks using spectral graph methods.
- Through ablation analysis on n=10 trials, we find no significant improvement in our model performance when incorporating these features.

Introduction
Web traffic patterns evolve in response to information demand. Models of these patterns can be used to analyze trends across pages or to forecast future demand. Can the hyperlinked structure of Wikipedia be incorporated to improve performance?

Data
The data is processed using [4] on the Wikipedia SQL, dump from August 2019 and 60-weeks of page view data from 2018-01 to 2019-03. There are $5.9 \times 10^6$ pages and $4.9 \times 10^8$ directed hyperlinks. We randomly sample a fraction of articles, compute the induced graph, and use the largest connected component for experimentation.

Experiments and Features
In experiment 1, we observe random, connected graphs with size $|V| = 3.7 \times 10^4$ and $|E| = 1.7 \times 10^5$ and compute PageRank and a spectral embedding [2]. These are concatenated with the historical page views.

![Figure 2: Spectral embedding of a sampled sub-graph.](Image)

In experiment 2, we recursively cluster a sub-graph of size $|V| = 2.9 \times 10^6$ and $|E| = 5.2 \times 10^7$ to generate sign and vector features.

![Figure 3: Sample record from experiment 2 with metadata, degree, sign, vector, and page view columns.](Image)

Models
We report the performance of the following regression models for page view traffic.
- Persistence: $\hat{y}^{(i)} = x(t)$
- Mean: $\hat{y}^{(i)} = \frac{1}{|E|} \sum_{(e)} x(t)$
- Linear Regression: $\hat{y}^{(i)} = \sum_{t=1}^T \beta(t) x(t)$
- Ridge Regression: $\hat{y}^{(i)} = \sum_{t=1}^T \beta(t) x(t) - \frac{2}{T} \sum_{t=1}^T \beta(t)$
- Neural Network: Fully connected, feedforward, $L_2$ regularization
- TRMF[3]: Latent space model using matrix methods

Metrics
Mean Absolute Precision Error (MAPE):
$$MAPE = \frac{1}{nT} \sum_{i=1}^n \sum_{t=1}^T \left| \frac{y_i(t) - \hat{y}_i(t)}{y_i(t)} \right| \times 100\%$$

Normalized Root Mean Square Error (NRMSE):
$$NRMSE = \frac{1}{\sqrt{nT}} \sqrt{\frac{1}{nT} \sum_{i=1}^n \sum_{t=1}^T (y_i(t) - \hat{y}_i(t))^2}$$

Results
All results were gathered from 10 trials.

<table>
<thead>
<tr>
<th>Model</th>
<th>Exp 1</th>
<th>Exp 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>persistence</td>
<td>7.2 ± 1</td>
<td>7.0 ± 0</td>
</tr>
<tr>
<td>mean</td>
<td>5.7 ± 0.6</td>
<td>7.2 ± 0</td>
</tr>
<tr>
<td>linear reg.</td>
<td>6.1 ± 0.6</td>
<td>7.6 ± 0.3</td>
</tr>
<tr>
<td>ridge reg.</td>
<td>6.0 ± 0.6</td>
<td>7.5 ± 0.3</td>
</tr>
<tr>
<td>neural network</td>
<td>5.9 ± 0.7</td>
<td>6.9 ± 0.3</td>
</tr>
<tr>
<td>trmf</td>
<td>-</td>
<td>6.5 ± 0.3</td>
</tr>
</tbody>
</table>

Ablation Analysis

<table>
<thead>
<tr>
<th>Experiment</th>
<th>NRMSE</th>
<th>MAPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>neural network</td>
<td>5.9 ± 1.0</td>
<td>6.9 ± 0.9</td>
</tr>
<tr>
<td>without history</td>
<td>0.2 ± 0.0</td>
<td>-0.4 ± 0.4</td>
</tr>
<tr>
<td>without pagerank</td>
<td>0.1 ± 0.4</td>
<td>0.0 ± 0.4</td>
</tr>
<tr>
<td>without embedding</td>
<td>0.0 ± 0.4</td>
<td>0.0 ± 0.4</td>
</tr>
</tbody>
</table>

Discussion
We were not able to build a model that consistently outperformed simple persistence and linear models. This may be due to limited resolution of the data, as shown by lackluster performance of TRMF. We determined that the embedding features do not improve performance at daily granularity. In the second experiment, we note some reliance on graph features by observing the greater increase in MAPE by dropping historical page views than the sign or vector features.

Future
We would proceed next by building a convolutional neural network to exploit symmetries in the time and graph signals, as well as forecasting over a longer window (months) or higher frequency (hours). Additionally, we would attempt to scale the first few cuts of our partitioning implementation increase our training set for non-linear models.

References