Predicting your Next Stop-Over from Location-Based Social Network Data with Recurrent Neural Networks

RecTour workshop 2017
RecSys 2017, Como, Italy

Enrico Palumbo, ISMB, Italy, Turin
Giuseppe Rizzo, ISMB, Italy, Turin
Raphaël Troncy, EURECOM, France, Sophia Antipolis
Elena Baralis, Politecnico di Torino, Italy, Turin
Location-Based Social Network Data

- Allow users to **check-in** in a **venue** and share information about their whereabouts
- Venues are **categorized** according to consistent taxonomies
- User profiles allow to **group users** according to demographic information
Next-POI prediction problem

- Predict where a user will go next depending on past check-ins and profile
- Modeling **sequential information**: where is a user likely to go after an *Art Museum* and a *Sushi Restaurant*?
- Modeling **user profile** information (e.g. *Vegetarian Restaurant and Steakhouse* do not go together)
Frédéric, French man in his 40's, is going to visit Tate Modern in London this afternoon.

Then, he can be interested in:
- Taking a beer in an Irish Pub, first
- Then, having a dinner in an Italian Restaurant
- And, lately, attending an event in a Jazz Club

path, ie sequence, of categories of Points of Interest that Frédéric can be interested to go after having seen the Tate Modern gallery.
1.5M check-ins from 235.6K Swarmapp users worldwide in a week via Twitter API
Collect venue category, user gender and language with Foursquare API
Data cleansing and path extraction

- **At least 10 check-ins:** prefiltering, select active users who are likely to generate paths
- **Bot removal:** if more than twice two check-ins in a minute, it is a bot (p~1 on a sample of 50 examples)
- **Segmentation:** split a stream of checkins for a user \((c_1, \tau_1), \ldots, (c_n, \tau_n)\) when the temporal interval is more than 8 hours: \(\tau_{i+1} - \tau_i > 8\). ‘Isolated’ check-ins, \(\tau_{i+1} - \tau_i > 8, \tau_i - \tau_{i-1} > 8\), are removed from the data.
- **Example:** \((\text{Art_Museum}, 8), (\text{Café}, 10), (\text{French_Restaurant}, 12), (\text{Rock_Club}, 22), (\text{Cocktail_Bar}, 23)\)

\(\tau_{i+1} - \tau_i = 10h\)
## Stats

<table>
<thead>
<tr>
<th></th>
<th>Users</th>
<th>Check-ins</th>
</tr>
</thead>
<tbody>
<tr>
<td>Collection</td>
<td>235.6 K</td>
<td>1.5 M</td>
</tr>
<tr>
<td>At least 10 check-ins</td>
<td>19.5 K</td>
<td>400 K</td>
</tr>
<tr>
<td>Without Bots</td>
<td>12.4 K</td>
<td>184 K</td>
</tr>
<tr>
<td>Segmentation</td>
<td>12 K</td>
<td>123 K</td>
</tr>
</tbody>
</table>

**29.5K paths, with a max length of 50 and an avg length of 4.2**

**Path:** Cafeteria, Vegetarian Restaurant, Café, Shopping Mall, Smoothie Shop, Spa, Wine Bar, Opera House, EOP
Recurrent Neural Networks

- Model sequences
- Output layer: $o_t = f_o(x_t, h_t)$
- Hidden layer: $h_t = f_h(x_t, h_{t-1})$
- No Markov assumption of fixed memory window
- Very popular in NLP, language modeling, speech processing, semantic image analysis
RNN application: next character prediction

- Train the network to predict the next character in a sentence
- Feed one character at a time into the network
- Use the trained model to generate new text
<table>
<thead>
<tr>
<th>Input</th>
<th>Target</th>
</tr>
</thead>
<tbody>
<tr>
<td>Café, Art Museum, Monument, Italian Restaurant, Irish Pub, Night Club, EOP</td>
<td>Chinese Restaurant, Pub, Wine Bar, Karaoke Bar, Rock Club, EOP</td>
</tr>
<tr>
<td>Beach, Gym, Cocktail Bar, EOP</td>
<td>Cafeteria, Vegetarian Restaurant, Café, Shopping Mall, Smoothie Shop, Spa, Wine Bar, Opera House, EOP</td>
</tr>
</tbody>
</table>
Architecture

Art_Museum  Park  Sushi_Restaurant  EOP  Target

Input layer  Hidden layer(s)  Output layer

Target

Cafeteria  Art_Museum  Park  Sushi_Restaurant

GRU  GRU  GRU  GRU

Softmax  Softmax  Softmax  Softmax

h_0  h_1  h_2  h_3

x_0  x_1  x_2  x_3

encoding  encoding  encoding  encoding

encoding  encoding  encoding  encoding

Dropout  Dropout  Dropout  Dropout
One-hot encoding

- **Encoding**: a strategy to turn categories $c$ into vectors $x$
- **One-hot encoding**:
  - assign to each category $c$ a unique index $\alpha = \alpha(c)$.
  - $x_i(c) = 1$ if $i = \alpha$ else 0
  - $|x_i(c)| = |C|$ where $c \in C$
- Simple and intuitive
Shortcomings

- **Independence**: all categories are considered as orthogonal and equidistant.
- **Size**: the size of the input vector depends on the size of the ‘vocabulary’, i.e. the set of categories.
node2vec

- Feature learning from networks
- Adaptation of word2vec on graph structures using random walks
- Maps nodes in a graph into an euclidean space preserving the graph structure
node2vec workflow

Graph

Random Walk

Text file with ‘sentences’

node1, node3, node2, node1, node5 ...

node2, node3, node1, node2, node5, node4 ...

node3, node1, node2, node1, node3 ...

node4, node3, node4, node1, node5 ...

Word2vec

Node embeddings
node2vec on Foursquare taxonomy

- Foursquare taxonomy is a graph that models dependencies among venue categories
- Using node2vec we obtain category vectors that models dependencies

http://projector.tensorflow.org/?config=https://gist.githubusercontent.com/enricopal/9a2683de61f5fb16c4f59ae295e3fef7/raw/159df72f47e881d0f314096fcc8ea561fb7132b9/projector_config.json
GRU units

- Hidden layer with **Gated Recurrent Units (GRU)**
- $h_{t+1} = \text{GRU}(x_t, h_t)$
- Able to effectively model **long-term dependencies**
- Recently they have proven to be easier to train w.r.t. to Long Short-Term Memory Networks (LSTM)

Dropout

● **Regularization** by randomly switching off neurons at training time
● **Prevent** neurons from co-adapting and **overfitting**
● Can be seen as an ensemble of slightly different networks
● $k = 0.2$
Softmax

- Output layer has length equal to $|C|$.
- Turn activation values into a probability distribution over the categories.

$$\text{softmax}(o_t^\alpha) = \frac{e^{o_t^\alpha}}{\sum_{k=1}^{\left|C\right|} e^{o_t^k}}$$
Loss function

- **Cross entropy** between the model probability distribution and the observed distribution
- One-hot encoding for target vectors
- Target $\rightarrow \alpha$
- $L = - \log (\text{softmax}(o^\alpha))$
Evaluation

- Train, validation and test set split
- Accuracy weights error in the same way (e.g. predicting ‘Art Museum’ when the true category is ‘History Museum’ counts the same as predicting ‘Karaoke Bar’)
- **Perplexity (ppl)**: borrowed from neural language modeling, measures how much the model is ‘surprised’ in observing the test set, the lower the better: \( \text{ppl} = 2^L = 2^H(p,q) \) where \( p \) is the model distribution and \( q \) is the category distribution in the test set
- A random uniform model has \( \text{ppl} = |C| \)
- \( \text{ppl} \) can be interpreted as the number of categories among which a random uniform model would have to choose
## Hyper parameters optimization

<table>
<thead>
<tr>
<th>rank</th>
<th>n_hidden</th>
<th>l_r</th>
<th>epochs</th>
<th>n_layers</th>
<th>ppl</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>64</td>
<td>10^{-4}</td>
<td>5</td>
<td>3</td>
<td>71.333</td>
</tr>
<tr>
<td>2</td>
<td>64</td>
<td>10^{-4}</td>
<td>5</td>
<td>2</td>
<td>71.609</td>
</tr>
<tr>
<td>3</td>
<td>64</td>
<td>10^{-4}</td>
<td>2</td>
<td>3</td>
<td>71.630</td>
</tr>
<tr>
<td>4</td>
<td>128</td>
<td>10^{-4}</td>
<td>2</td>
<td>2</td>
<td>71.645</td>
</tr>
<tr>
<td>5</td>
<td>128</td>
<td>10^{-4}</td>
<td>5</td>
<td>2</td>
<td>72.048</td>
</tr>
<tr>
<td>range</td>
<td>[64, 128]</td>
<td>[10^{-4}, 5*10^{-4}, 10^{-3}]</td>
<td>[1, 2, 5]</td>
<td>[2, 3]</td>
<td>-</td>
</tr>
</tbody>
</table>

Grid-search of hyper-parameters and perplexity on the validation set, top 5 configurations
### Results

RNN with node2vec encoding, one-hot encoding vs bigram and random model.

<table>
<thead>
<tr>
<th>System</th>
<th>ppl</th>
</tr>
</thead>
<tbody>
<tr>
<td>RNN-node2vec</td>
<td>75.271</td>
</tr>
<tr>
<td>RNN-one-hot</td>
<td>76.472</td>
</tr>
<tr>
<td>bigram</td>
<td>125.361</td>
</tr>
<tr>
<td>random</td>
<td>741</td>
</tr>
</tbody>
</table>

As if we were to choose randomly among ~75 categories.

Bigram model, 1-st order transition probabilities estimated, Laplace add-one smoothing.

This is exactly the number of categories |C| in the training data.
User clustering results

- **Paths** depend on user preferences
- Not enough data to learn a personalized model
- **Groups of users** according to user profile (no cold start)
- **Gender, language** extracted from Twitter and Foursquare

<table>
<thead>
<tr>
<th>Gender</th>
<th>Language</th>
<th>Paths</th>
<th>ppl</th>
</tr>
</thead>
<tbody>
<tr>
<td>M</td>
<td>All</td>
<td>18718</td>
<td>75.478</td>
</tr>
<tr>
<td>F</td>
<td>All</td>
<td>8741</td>
<td>73.024</td>
</tr>
<tr>
<td>All</td>
<td>En</td>
<td>8955</td>
<td>71.343</td>
</tr>
<tr>
<td>All</td>
<td>Ar</td>
<td>439</td>
<td>100.772</td>
</tr>
<tr>
<td>F</td>
<td>En</td>
<td>2636</td>
<td>74.999</td>
</tr>
<tr>
<td>M</td>
<td>En</td>
<td>5771</td>
<td>69.481</td>
</tr>
<tr>
<td>F</td>
<td>Th</td>
<td>234</td>
<td>83.601</td>
</tr>
<tr>
<td>M</td>
<td>Th</td>
<td>524</td>
<td>94.132</td>
</tr>
<tr>
<td>All</td>
<td>All</td>
<td>29465</td>
<td>75.271</td>
</tr>
</tbody>
</table>
Correlation with training set size

- **Perplexity** vs number of paths in the user cluster training set
- **Spearmann correlation** = -0.48
- **p value** = 0.02
Conclusions

● Novel approach to next POI prediction passing through POI categories

● **Recurrent Neural Network** to model sequences of POI categories for next POI prediction

● **Category encoding** with *node2vec* vs one-hot encoding

● **Perplexity** as an evaluation metric

● **User clustering** based on demographics

● Will be integrated with entity2rec and user context to provide next POI instance from the POI category
Thank you!

- Enrico Palumbo, Data Scientist, ISMB, PhD Student, Eurecom - PoliTo
- Mail: palumbo@ismb.it, enrico.palumbo@eurecom.fr
- Personal page: www.enricopal.github.io
- Slides: http://www.slideshare.net/EnricoPalumbo2
- Twitter: https://twitter.com/enricopalumbo91
Outline

- Next-POI prediction problem
- Location-Based Social Network (LBSN) Data
- Recurrent Neural Networks
- Model
- User groups
- Data collection
- Category encoding
- Evaluation
- Results
Architecture

**Art_Museum**

- $c_1$
- Softmax
- $O_0$
- $W_o$
- $h_0$
- Dropout
- GRU
- $x_0$
- encoding

**Park**

- $c_2$
- Softmax
- $O_1$
- $W_o$
- $h_1$
- Dropout
- GRU
- $x_1$
- encoding

**Sushi_Restaurant**

- $c_3$
- Softmax
- $O_2$
- $W_o$
- $h_2$
- Dropout
- GRU
- $x_2$
- encoding

**EOP**

- $c_4$
- Softmax
- $O_3$
- $W_o$
- $h_3$
- Dropout
- GRU
- $x_3$
- encoding

**Target**

- Output layer
- Hidden layer(s)
- Input layer

**Cafeteria** **Art_Museum** **Park** **Sushi_Restaurant**
GRU equations

\[
r_t = \text{sigmoid}(W_r h_{t-1} + W_r x_t + b_r)
\]
\[
h'_t = \tanh(W_i (r_t \otimes h_{t-1}) + W_i x_t + b_i)
\]
\[
z_t = \text{sigmoid}(W_z h_{t-1} + W_z x_t + b_z)
\]
\[
h_t = z_t \otimes h' + (1 - z_t) \otimes h_{t-1}
\]
Optimizer

- **Adam Optimizer**: adaptive per neuron learning rate (neurons that are frequently active have small learning rate, neurons that are not frequently active have large learning rate) + momentum (add part of previous weight change to current weight change)
Objective

- **Understanding** temporal correlations among venue categories in check-ins
- **Next POI category prediction**
- More general and more easily portable results
- **Path** = sequences of venue categories
- Category can then be turned into a specific venue by querying a database with a variety of parameters (e.g. user location, time, user preferences, weather...)