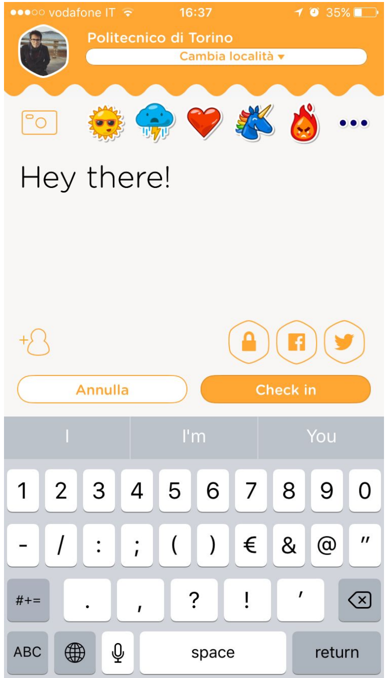
# Predicting your Next Stop-Over from Location-Based Social Network Data with Recurrent Neural Networks

***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*

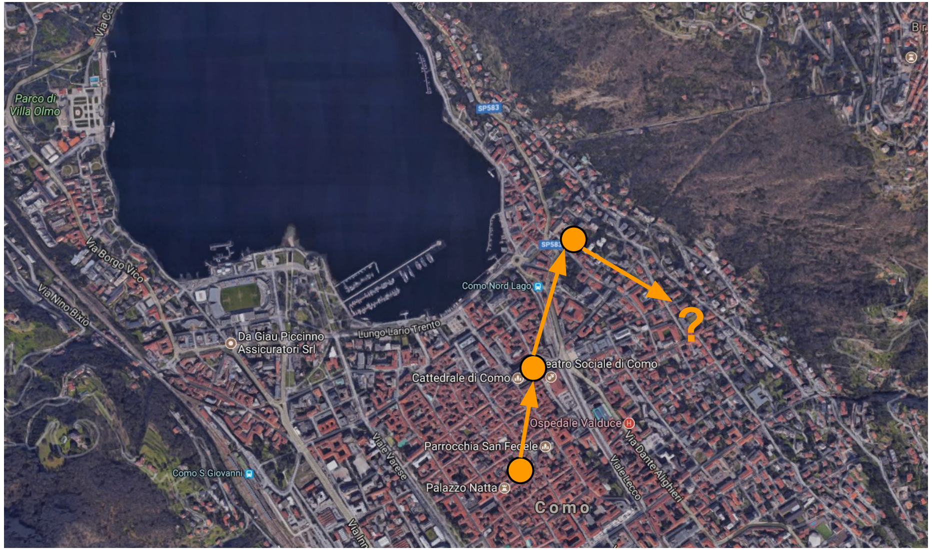
# Location-Based Social Network Data

- 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)



PASTIME

# Next-POI category prediction

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

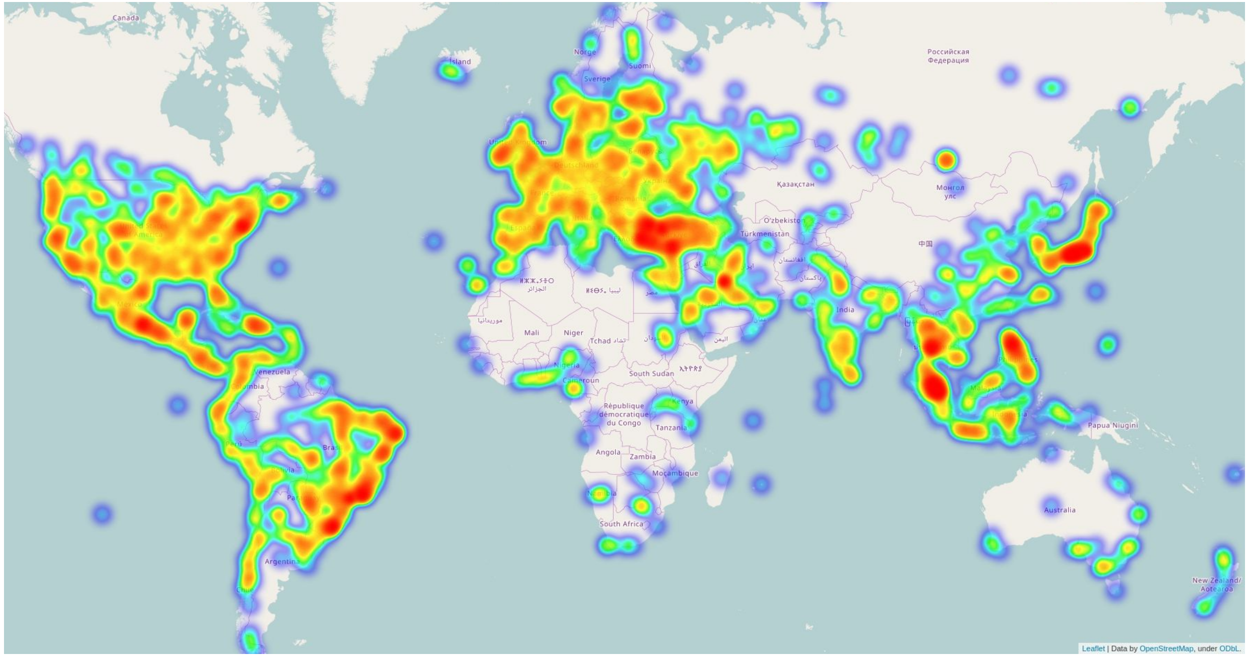
Then, having a dinner in an **Italian Restaurant**

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

PASTIME

# Data Collection and Enrichment




**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: $\boldsymbol{\tau_{i+1} - \tau_i > 8}$. 'Isolated' check-ins, $\tau_{i+1} - \tau_i > 8$, $\tau_i - \tau_{i-1} > 8$, are removed from the data.
- Example: (Art_Museum, 8), (Café, 10), (French_Restaurant, 12), (Rock_Club, 22), (Cocktail_Bar, 23)

$\boldsymbol{\tau_{i+1} - \tau_i} = 10\text{h}$

# Stats

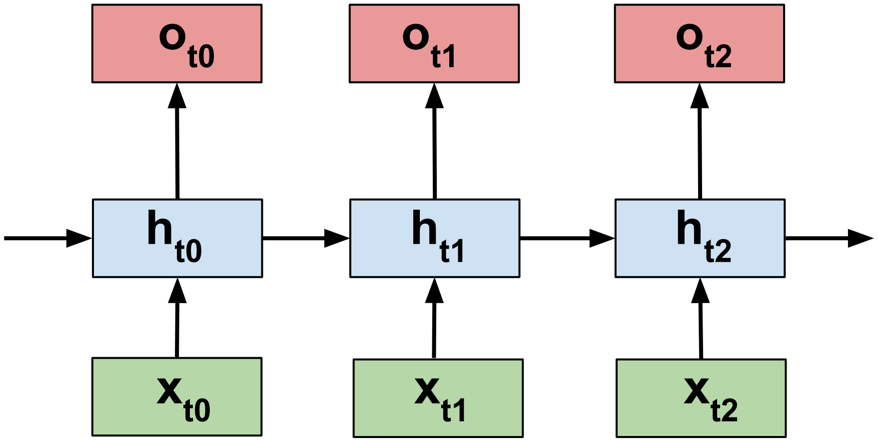
| | Users | Check-ins |
|---|---|---|
| **Collection** | 235.6 K | 1.5 M |
| **At least 10 check-ins** | 19.5 K | 400 K |
| **Without Bots** | 12.4 K | 184 K |
| **Segmentation** | 12 K | 123 K |

**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*

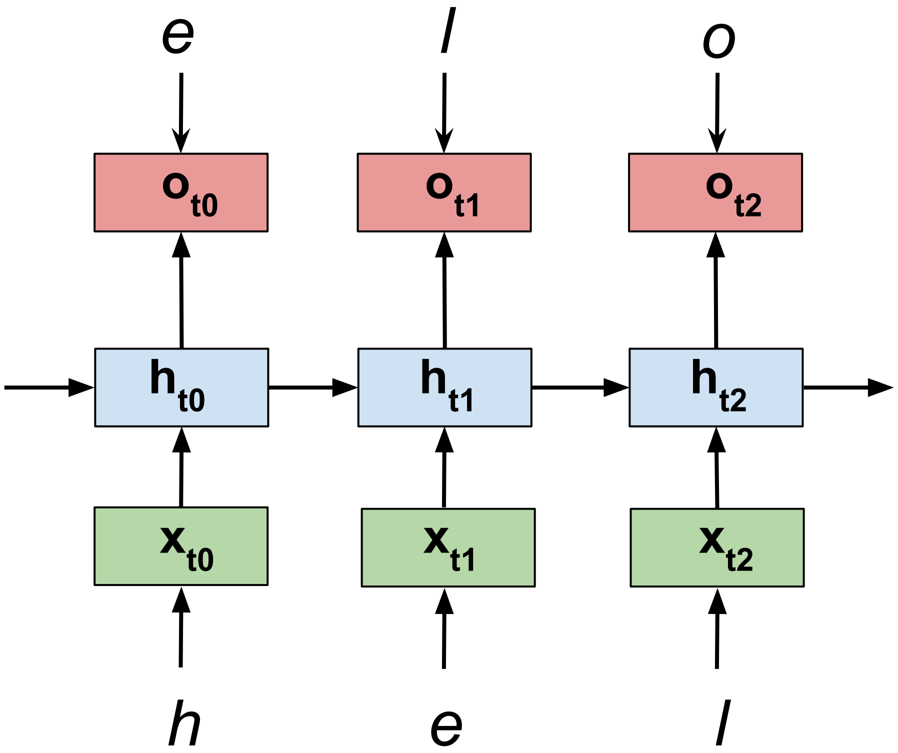
# Recurrent Neural Networks

- Model sequences
- Output layer: $\mathbf{o_t = f_o(x_t, h_t)}$
- Hidden layer: $\mathbf{h_t = f_h(x_t, h_{t-1})}$
- 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 the time into the network
- ‘hello’: ‘h’-> ‘e’, ‘e’ -> ‘l’, ‘l’ -> ‘o’
- Use the trained model to generate new text



# Next POI category prediction

Input Target

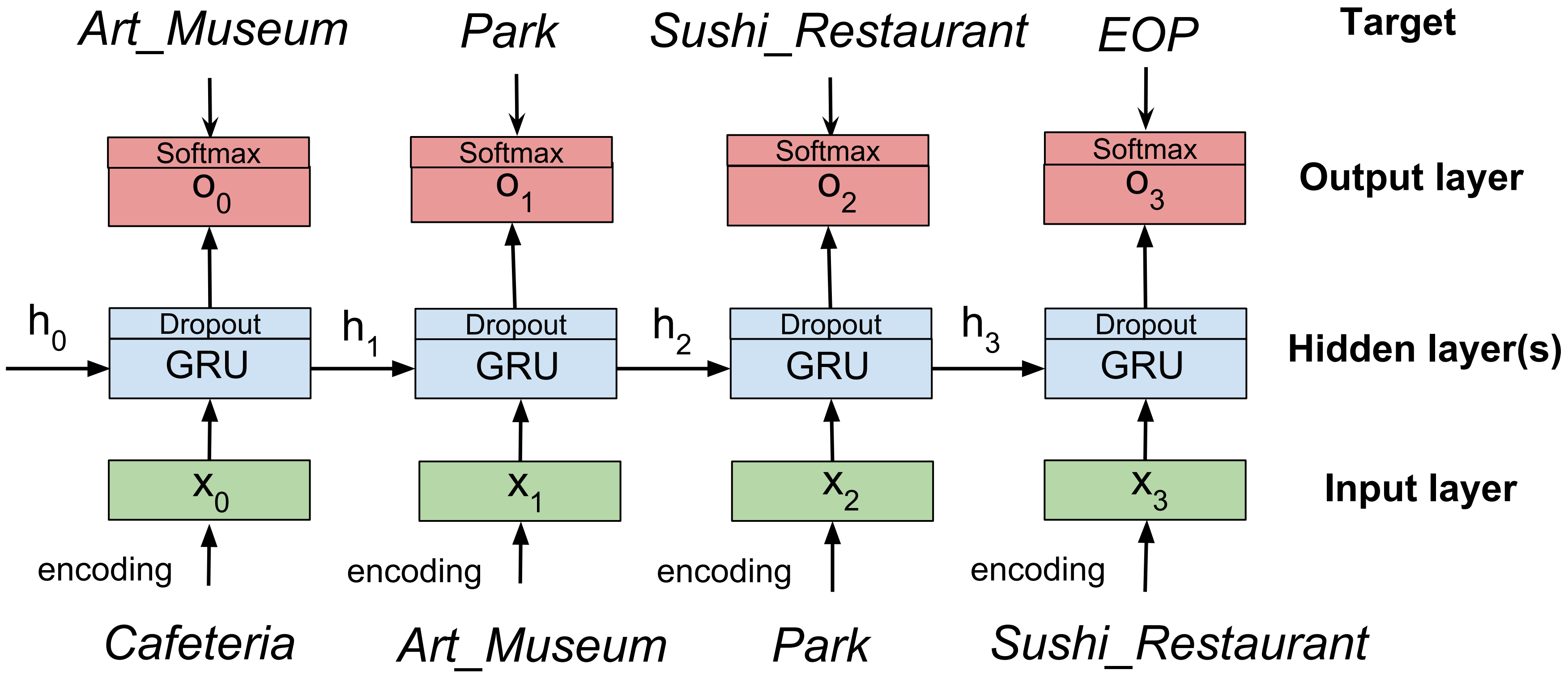
*Café, Art Museum, Monument, Italian Restaurant, Irish Pub, Night Club, EOP*

*Chinese Restaurant, Pub, Wine Bar, Karaoke Bar, Rock Club, EOP*

*Beach, Gym, Cocktail Bar, EOP*

*Cafeteria, Vegetarian Restaurant, Café, Shopping Mall, Smoothie Shop, Spa, Wine Bar, Opera House, EOP*

# Architecture



# One-hot encoding

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

**α** = *4*

| | | | | | |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 0 | 0 |

α (*Art Museum*) = *4*

c = *Art Museum*

PASTIME

# 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

*Art Museum*

| | | | | | |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 0 | 0 |

*Café*

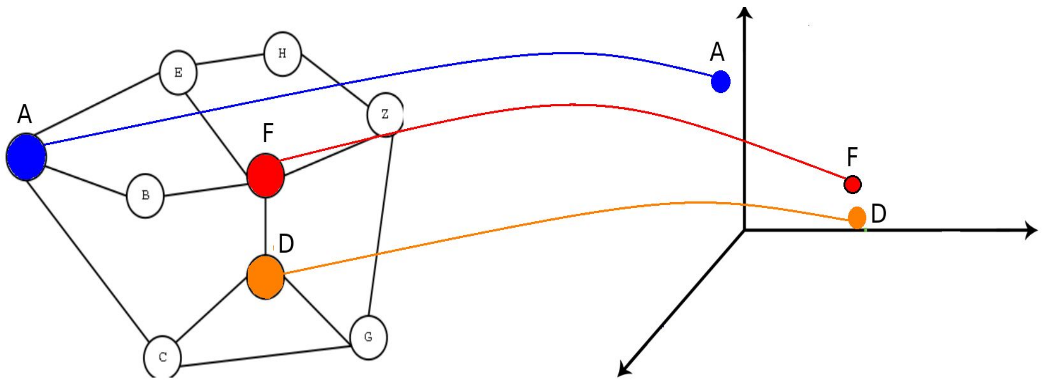
| | | | | | |
|---|---|---|---|---|---|
| 0 | 0 | 1 | 0 | 0 | 0 |

*History Museum*

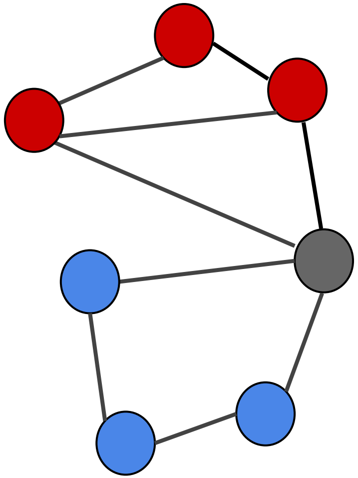
| | | | | | |
|---|---|---|---|---|---|
| 0 | 1 | 0 | 0 | 0 | 0 |

# 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

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

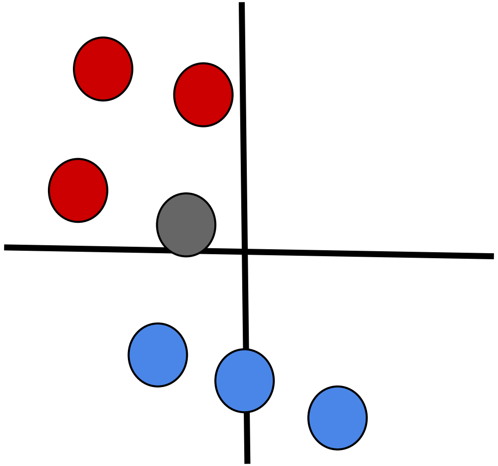
node2, node3, node1, node2, node5, node4 …

node3, node1, node2, node1, node3 …

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

Text file with ‘sentences’

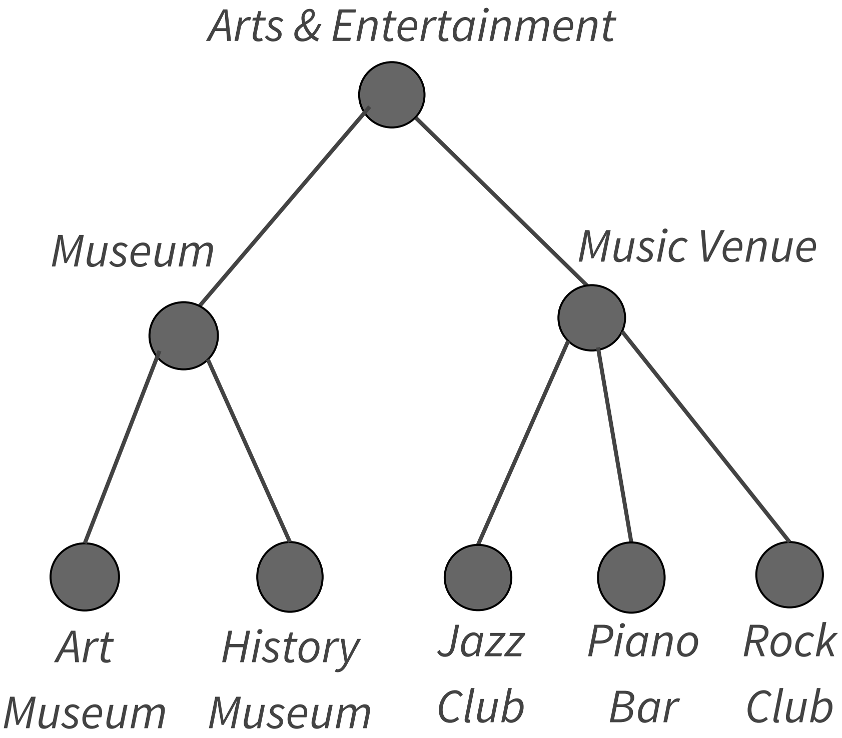
Word2vec



Node embeddings

# node2vec on Foursquare taxonomy

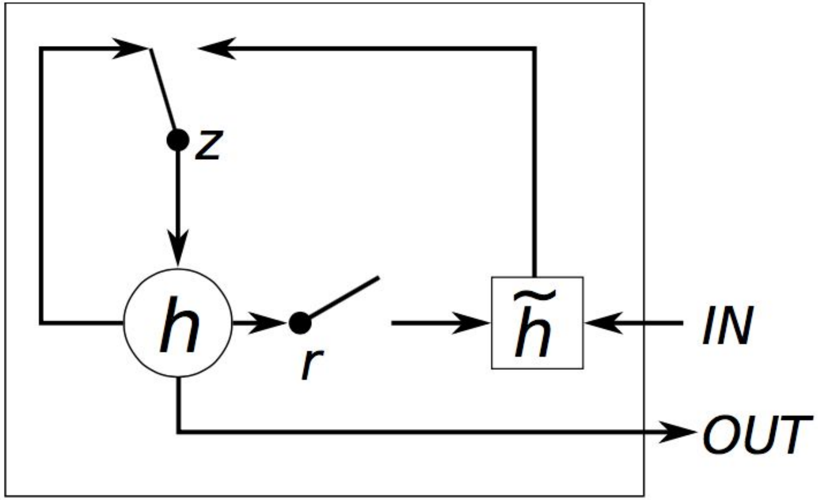
- Foursquare taxonomy is a graph that models dependencies among venue categories
- Using node2vec we obtain category vectors that models dependencies




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

# GRU units

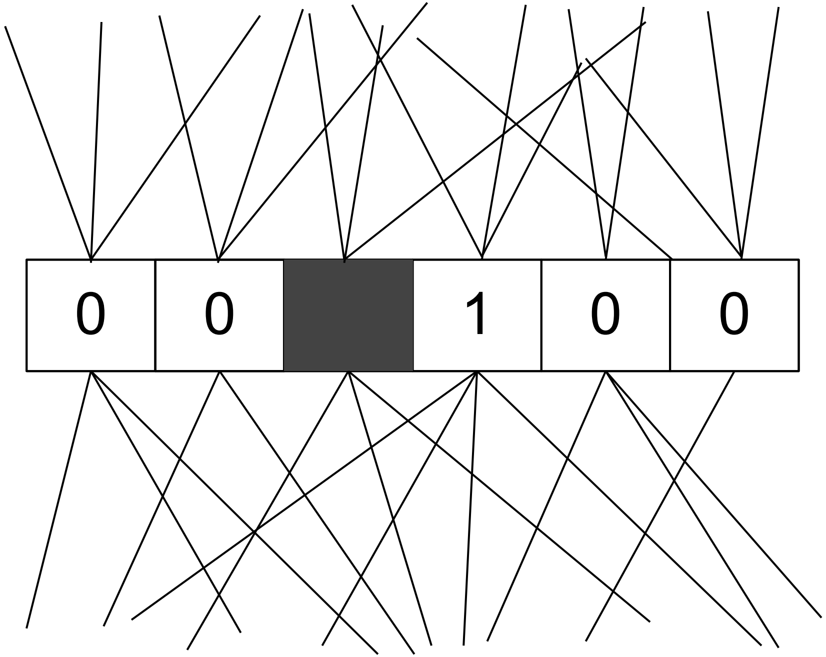
- Hidden layer with **Gated Recurrent Units (GRU)**
- $\mathbf{h_{t+1} = GRU(\, x_t, h_t)}$
- Able to effectively model **long-term dependencies**
- Recently they have proven to be easier to train w.r.t. to Long Short-Term Memory Networks (LSTM)




img source:
http://www.wildml.com/2015/10/recurrent-neural-network-tutorial-part-4-implementing-a-grulstm-rnn-with-python-and-theano/

# Dropout

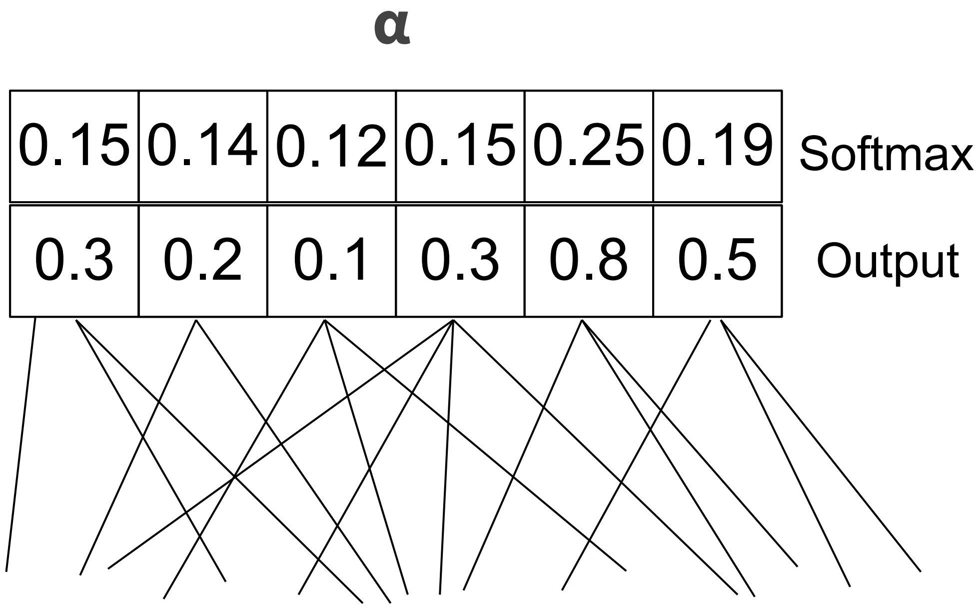
- **Regularization** by randomly switching off neurons at training time
- **Prevent** neurons from co-adapting and **overfitting**
- Can be seen as an ensemble of slightly different networks
- **k = 0.2**



# Softmax

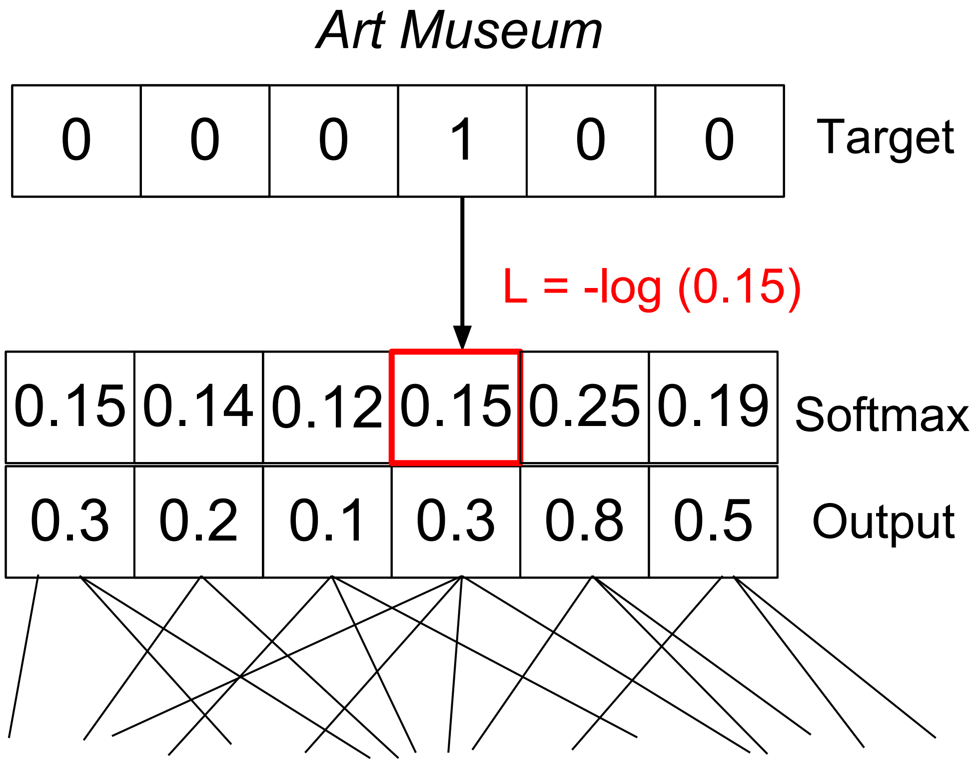
- Output layer has length equal to |C|
- Turn activation values into a **probability distribution** over the categories

$$softmax(o_t^\alpha) = \frac{e^{o_t^\alpha}}{\sum_{k=1}^{|C|} e^{o_t^k}}$$



# Loss function

- **Cross entropy** between the model probability distribution and the observed distribution
- One-hot encoding for target vectors
- Target -> **α**
- **L = - log (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**: **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 ppl = $|C|$
- ppl can be interpreted as the number of categories among which a random uniform model would have to choose

PASTIME

# Hyper parameters optimization

| rank | n_hidden | l_r | epochs | n_layers | ppl |
|---|---|---|---|---|---|
| 1 | 64 | $10^{-4}$ | 5 | 3 | 71.333 |
| 2 | 64 | $10^{-4}$ | 5 | 2 | 71.609 |
| 3 | 64 | $10^{-4}$ | 2 | 3 | 71.630 |
| 4 | 128 | $10^{-4}$ | 2 | 2 | 71.645 |
| 5 | 128 | $10^{-4}$ | 5 | 2 | 72.048 |
| **range** | [64, 128] | [$10^{-4}$, $5*10^{-4}$, $10^{-3}$] | [1, 2, 5] | [2, 3] | - |

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

# Results

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

| System | ppl |
|---|---|
| RNN-node2vec | 75.271 |
| RNN-one-hot | 76.472 |
| bigram | 125.361 |
| random | 741 |

As if we were to choose randomly among ~75 categories

This is exactly the number of categories |C| in the training data

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

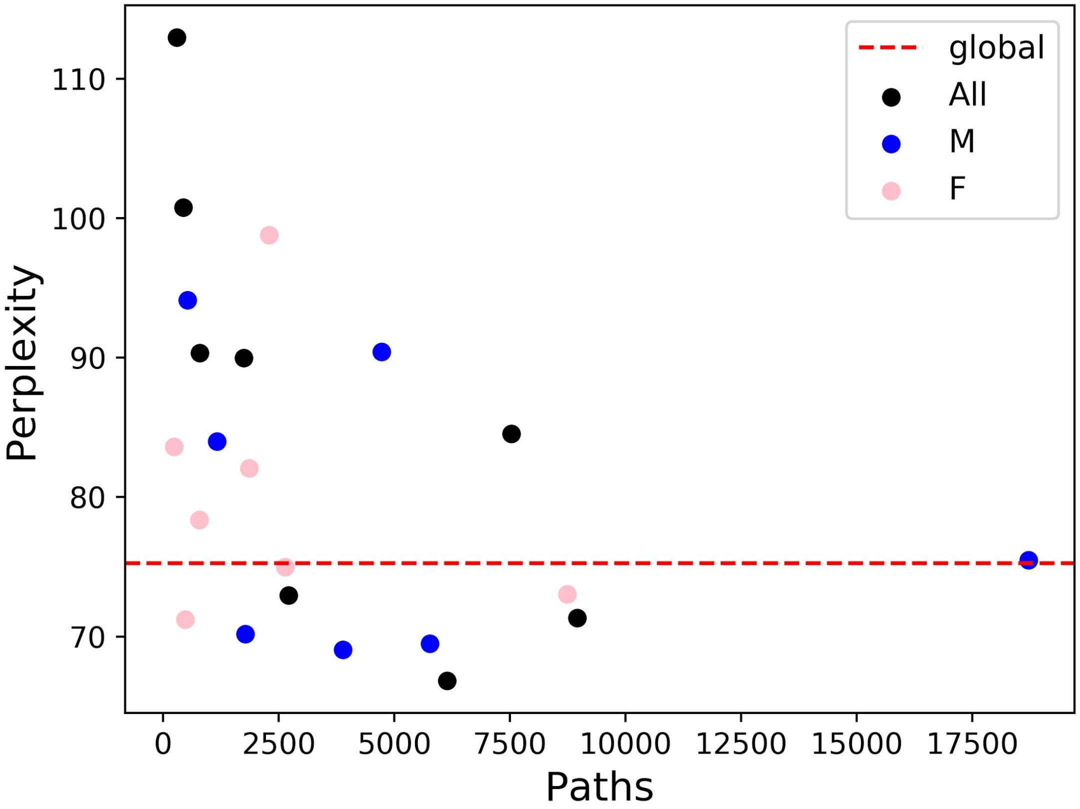
# User clustering results

- **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

| Gender | Language | Paths | ppl |
|---|---|---|---|
| M | All | 18718 | 75.478 |
| F | All | 8741 | 73.024 |
| All | En | 8955 | 71.343 |
| All | Ar | 439 | 100.772 |
| F | En | 2636 | 74.999 |
| M | En | 5771 | 69.481 |
| F | Th | 234 | 83.601 |
| M | Th | 524 | 94.132 |
| All | All | 29465 | 75.271 |

# 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!

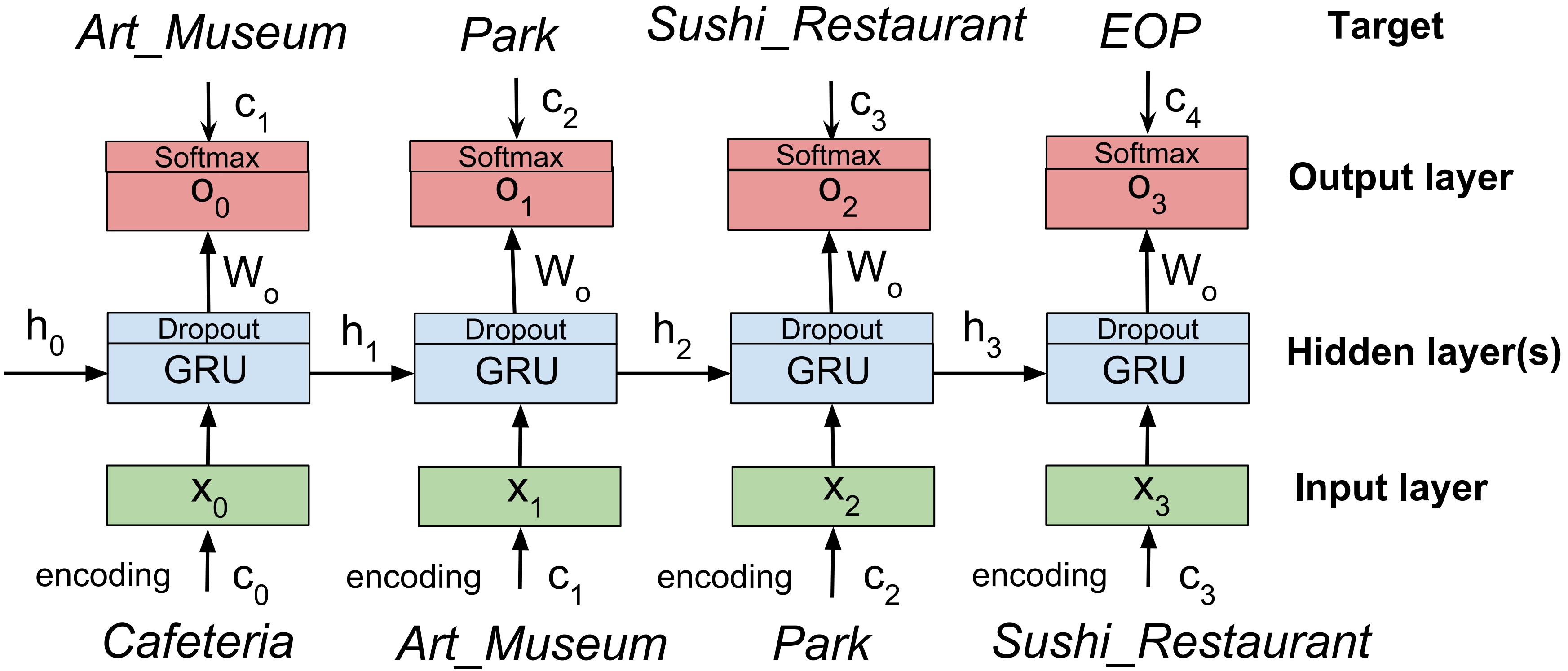
- Enrico Palumbo, Data Scientist, ISMB, PhD Student, Eurecom - PoliTo
- Mail: palumbo@ismb.it, enrico.palumbo@eurecom.fr
- Personal page: www.enricopal.github.io
- Slides: http://www.slideshare.net/EnricoPalumbo2
- Github: www.github.com/enricopal, www.github.com/MultimediaSemantics
- Twitter: https://twitter.com/enricopalumbo91

Turin, Italy, www.ismb.it | Sophia Antipolis, France, www.eurecom.fr | Turin, Italy, www.polito.it

# Outline

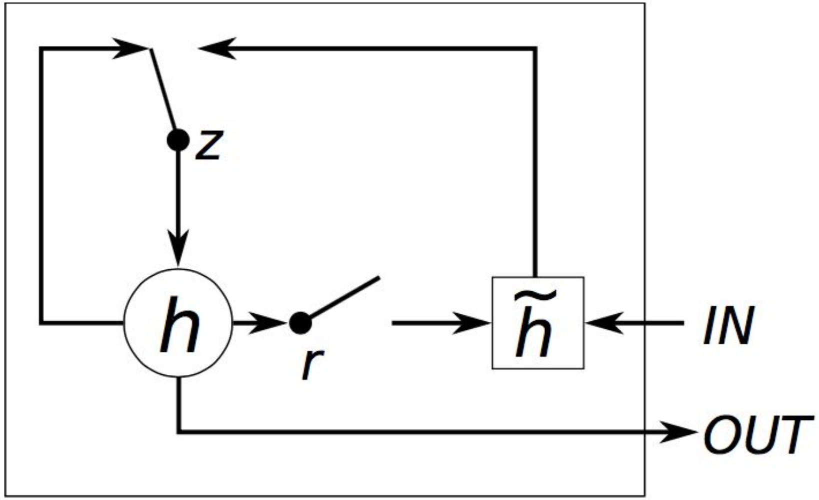
- Next-POI prediction problem
- Location-Based Social Network (LBSN) Data
- Recurrent Neural Networks
- Model
- User groups
- Data collection
- Category encoding
- Evaluation
- Results

# Architecture



# GRU equations

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



# 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…)