BIG DATA

Comp 790-134, Fall 2013

University of North Carolina, Chapel Hill

“Learning Latent Factor Models of Human Travel”

Submitted by:

Giuseppe Dinitale

Sahil Narang

Contents

Introduction

A 'good' travel model could yield new scientific insights into human behavior, aside for being useful in numerous applications. For example, travel models can help in predicting the spread of disease; surveying tourism, traffic, and special events mobility for urban planning; geolocating with computer vision; interpreting activity from movements; and recommending travel. Despite being an inherent human behavior and having numerous applications, very little work has been done to gain scientific insights into factors that dictate human travel. This may be attributed to the lack of information in the past. However, the proliferation of digital photo-capture devices equipped with GPS and the growing practice of sharing public photos online using social media sites has resulted in a huge volume of geo-tagged photos available on the Web. This has spurned a flurry of research using a variety of techniques.

The objective of our project was to implement and possibly improve the basic latent travel model put forth by Guerzhoy et. al. [1]. Essentially, we aim to model travel probabilities as functions of spatially-varying latent properties of locations and travel distance. The latent factors represent interpretable properties: travel distance cost and desirability of locations. These latent factors are combined in a multiplicative model which easily lends itself to incorporating additional latent factors and sources of information.

Problem Statement & Previous Work

Suppose we discretize the world into J locations, and we observe an individual at location i. We wish to predict where that individual is likely to be after a certain time interval :

$$P_{ij\tau} = P(Lnext = j \,|Lcurrent = i, \Delta T = \tau)$$

The following figure illustrates this.

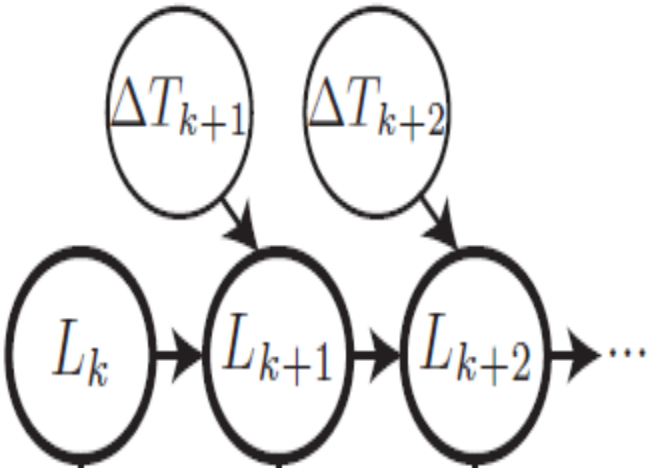

Figure 1: Our model should ideally determine the location of an individual to be L_{k+1} given that the individual was at L_k and the time interval is T_{k+1} [2]

Research into this area can be broadly classified into three distinct categories.

- First, scientists have employed the Levy flight, a stochastic process model [4]. In particular, the marginal probability distribution of traveling a distance in some fixed time interval is given by a truncated power law. This model captures the heavy-tailed nature of travel: long-distance travel is rare but not surprising. These simple parametric models can be interpreted to yield insights into travel behaviour. However, these models assume that travel probabilities depend only on the travel distance. Locations that are equidistant from one's current location are equally likely, regardless of whether they are in New York City or in the Arctic Circle.
- Second, one can build probability tables based on empirical travel histograms. For example, Kalogerakis et al. [2] use counts of actual transitions in a database of 6 million travel records. Such models are more accurate than simple stochastic process models, but require enormous datasets while revealing little about the underlying structure of the data.
- Third, there has been more recent work [1,3] that aims to combine the best of the earlier mentioned models by learning latent factors of human travel using machine learning and combining these factors to predict travel. Such techniques are useful for gaining insights into the dataset. Furthermore, this property lends such models to be more generalized.

Dataset

The dataset comprises the publicly-available Flickr.com image streams of 75,250 distinct individuals, collected by Kalogerakis et al. [2]; there are about 6 million images in total in the dataset. Each image is accompanied by a time-stamp, and a geo-tag specifies the location where the picture was taken. Since we are concerned only with the travel, we ignore the images themselves, and consider only the geo-tags and the time intervals between consecutive geo-tags. Furthermore, the earth was discretized into 3186 bins, each of size 400km x 400 km. The metadata along with the bin mapping was provided by Michael Guerzhoy [1].

The following figure illustrates the discretization of Earth:

Figure 2: Discretization of Earth into 3186 bins [2]

The following figure illustrates the distribution of data. One can see that while the distribution peaks around the major cities, almost the entire earth is sampled, except for a few bins in central Africa, Siberia, Antarctica and of course, the oceans.

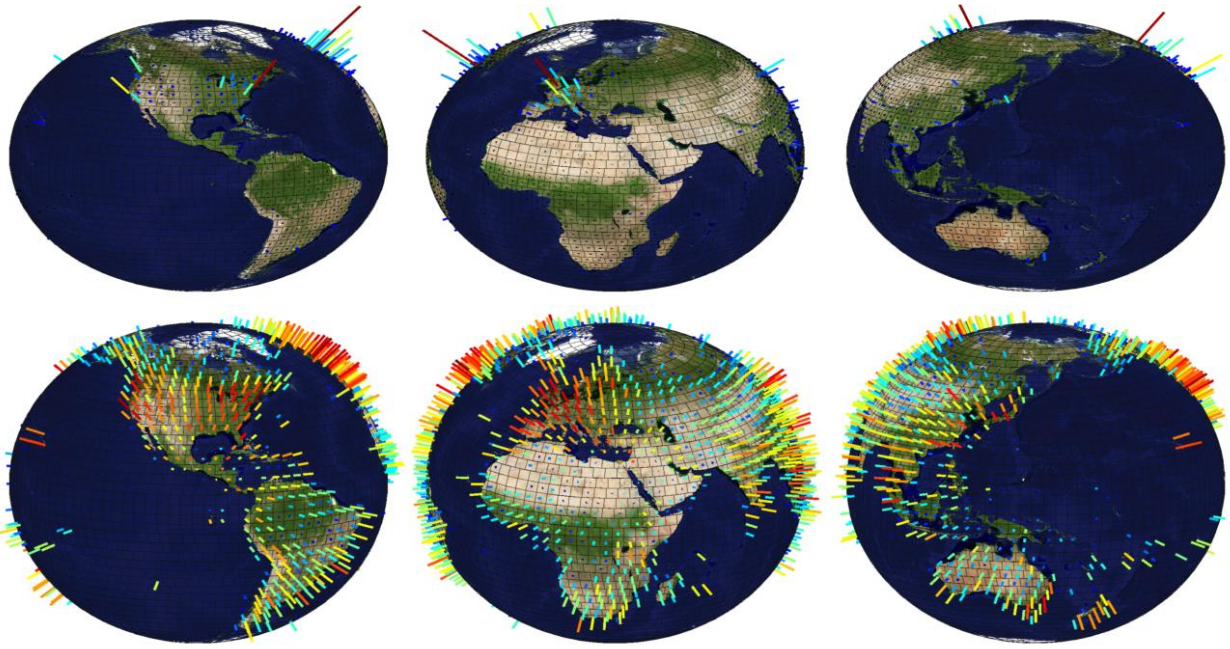

Figure 3: linear scale (top) and natural log scale (bottom). The height of each bar is proportional to the density of geotagged photos in each equal area region. Regions with zero photos have no bar. [2]

Basic Model

Given the time constraints for the project, we have considered two latent factors:

1. A distance factor r(d, τ) which captures the dependence of travel on the distance d_{ij} between locations I and j and the transition time τ. For a given τ, this term mimics the power-law property captured by the first category of techniques.
2. A spatially varying term a_j, which represents the desirability of the destination j. This term reflects the fact that some destinations attract more travel than others.

The two latent factors can be combined using the following multiplicative model:

$$P_{ij\tau} = \frac{r(d(i,j),\tau) * a_j}{\sum_\ell r(d(i,l),\tau) * \alpha_\ell}$$

Where P_{ijt} estimates the probability of travelling from location bin i to location bin j in transition time τ. As mentioned in [1], it is more convenient to parameterize the model in the log domain. Let,

$$\rho(d,\tau) = \log(r(d,\tau))$$
$$\alpha_j = \log a_j$$

We can now write the log-probability of travelling from i to j as:

$$P_{ij\tau} = \frac{\exp(\rho(d(i,j),\tau) + \alpha_j)}{\sum_\ell \exp(\rho(d(i,\ell),\tau) + \alpha_\ell)}$$

Learning using Batch Gradient

The most important component of this project was to learn the latent factors, α and ρ. This can be done by minimizing the negative log likelihood using an appropriate learning algorithm. The negative log-likelihood of the data is:

$$\text{NLL} = -\sum_{ij\tau} N_{ij\tau} \ln P_{ij\tau}$$

Where, N_{ijt} is the number of observed transitions from i to j in time interval τ.
Given the non-convex objective function, Non Linear Conjugate Gradient method would be an appropriate choice. However, we decided to implement Batch Gradient to gauge the correctness of our implementation.
Taking derivatives with respect to the model parameters yields:

$$\frac{\partial NLL}{\partial \propto_j} = -N_j + \sum_{i\tau} N_{i\tau} \frac{\exp\left(\rho\left(d_{ij},\, \tau\right) + \propto_j\right)}{\sum_{\iota} \exp(\, \rho(d_{i\iota},\, \tau) + \propto_{\iota})} \;= -N_j + \sum_{i\tau} N_{i\tau}\, P_{ij\tau}$$

$$\frac{\partial NLL}{\partial \rho_{\tau d}} = -N_{\tau d} + \sum_{ij} N_{ij\tau} \frac{\sum_{\iota : d_{i\iota} = d} \exp(\rho(d_{i\iota},\, \tau) + \propto_{\iota})}{\sum_{\iota} \exp(\, \rho(d_{i\iota},\, \tau) + \propto_{\iota})} \;= -N_{\tau d} + \sum_{i} N_{i\tau}\, P_{i\tau d}$$

One can see that the derivatives with respect to the model parameters can be naturally interpreted as the difference between expected counts of transitions and the observed counts of transitions.

$$-N_j + \sum_{i\tau} N_{i\tau}\, P_{ij\tau} = \; -N_j + E[N_j]$$

$$-N_{\tau d} + \sum_{i} N_{i\tau}\, P_{i\tau d} = \; -N_{\tau d} + E[N_{\tau d}]$$

Implementation Issues

1. Sequential Execution.
   The huge size of the dataset compelled us to create large number of summary tables and hash tables. While this did help immensely in speeding up execution, it also prevented us from parallelizing the code. Almost all our summary tables are global variables which are shared by various functions. It seems obvious that functions should be able to perform concurrent reads on these tables in parallel. Aside from parallelizing the derivative functions from within, it seemed like an obvious choice to create separate tasks for each that could execute in parallel. However, we found that MATLAB does not deal nice with global or persistent objects within its parallel constructs. This meant we had to execute the code sequentially.

2. Slow
   Our program's inability to execute in parallel effectively meant that the execution was painfully slow. For example, we ran our code on the KillDevil Cluster on campus and it took approximately 150 sec for each iteration. Again, executing it sequentially meant that we could not take advantage of the cluster.

3. Inf=Exp(x)
   Another issue we came across was inability of MATLAB to represent huge numbers. Given that P_{ijt} has an exponential term, we had to be very careful about the step size. Even an initial α value

of 500 would cause the exp to shoot up to Inf. We considered using the Matlab Symbolic Precision Toolbox but that wasn't a viable choice since it would have prevented us from performing actual computations on those 'symbols'.

4. Local Minima
   Given our initial choice of using Batch Gradient, this was an expected repercussion. Plotting the cost of our objective function for 1000 iterations clearly illustrated the oscillations about a local minima. Ideally, we would have liked to implement Non Linear CG to solve this problem but alas, we ran out of time.

Results

Figure 4 is a plot of a subset of the learned ρ values after running optimization with Batch Gradient Descent. Each line plot represents a different travel time interval while the y axis is ρ and the x axis represents distance in kilometers. Recall that ρ is the distance time factor which weights travel based on distance and time. As expected, travel for smaller durations and smaller distances have a much larger ρ because travel of smaller distances will occur in smaller amounts of time and much more frequently. Also worth noting, around 10,000km, there is an inflection point where the ρ values reverse order. Again, this is expected because travel over larger distances is less likely to happen in shorter time intervals, but still captures the trend that longer distance travel though rare can happen less frequently.

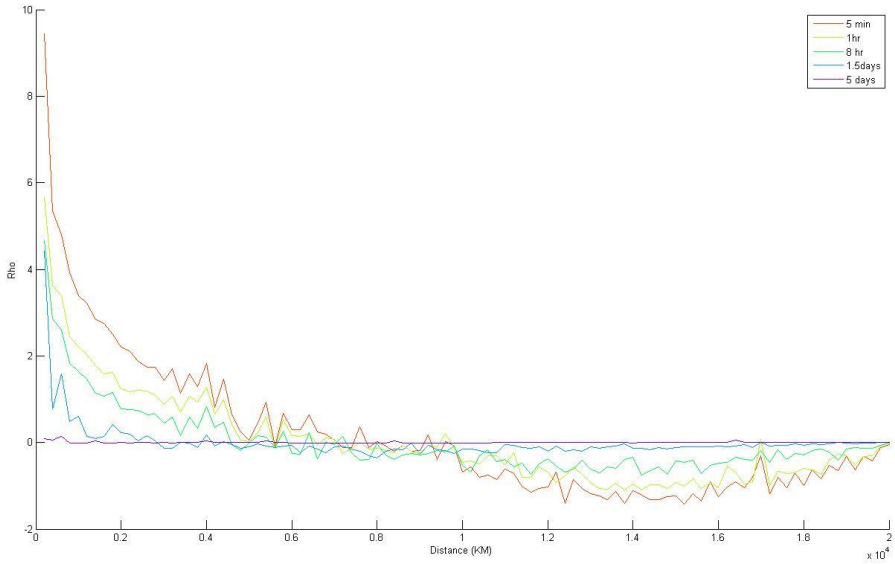

Figure 4: Plot of rho, the distance time factor, for a few select time spans

Let us examine one of the line plots as an example, say the five minute time interval. For relatively short distances, the ρ values for five minutes are larger because it is highly likely that a person will travel those distances in a short time span. Conversely, as distance gets larger, the five minute ρ values become smaller and even become negative which indicates the very small likelihood of a person travel such large distances in such a small time span.

There is a slight issue with the five minute ρ values (and likely with all the other ρ values as well) since there is an expectation of a sharp drop off as seen in [1]. The ρ values for five minutes should drop off extremely quickly since travel of extremely short distances (say 5km or less) would occur in such a small

duration. The expectation of a drop off implies that our model may not be optimal due to our use of Batch Gradient Descent.

Next, we take a look at the desirability of locations. Figure 5 is a graphic depicting the desirability of various locations on the globe according to our model. Red represents the most desirable while cyan represents the least desirable (we did not color bins that had a relative desirability of less than .2). As expected, areas around New York, parts of Europe, and a few select spots in Asia and Australia are quite desirable travel destinations.

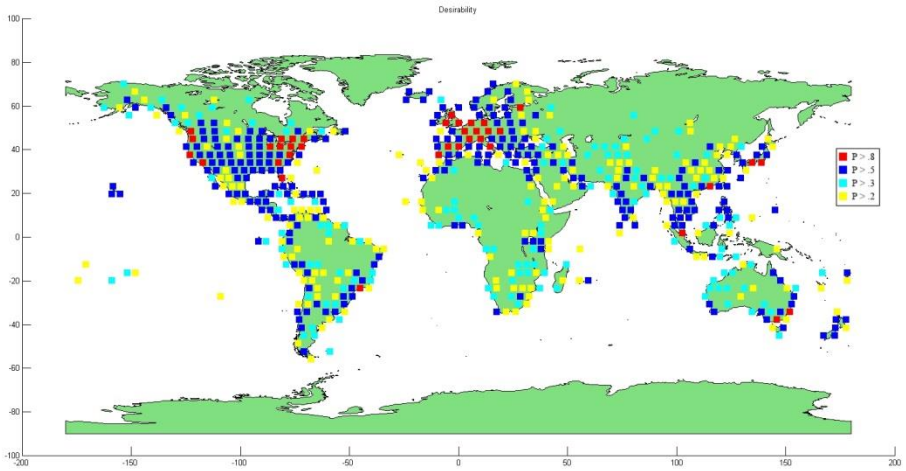

Figure 5: A map displaying the desirability of various locations

Figure 6 below shows an example of the output from our learned model. The source bin is the one containing London, England and the top destinations are shown with varying travel times. As expected, when the travel time is five minutes or below, the expectation is to not leave the bin. As the travel time increases, say to one hour, travel to neighboring and relatively nearby bins can potentially occur. Given more than a couple of days of travel, it is perfectly reasonable to be on another continent, as the figure also depicts.

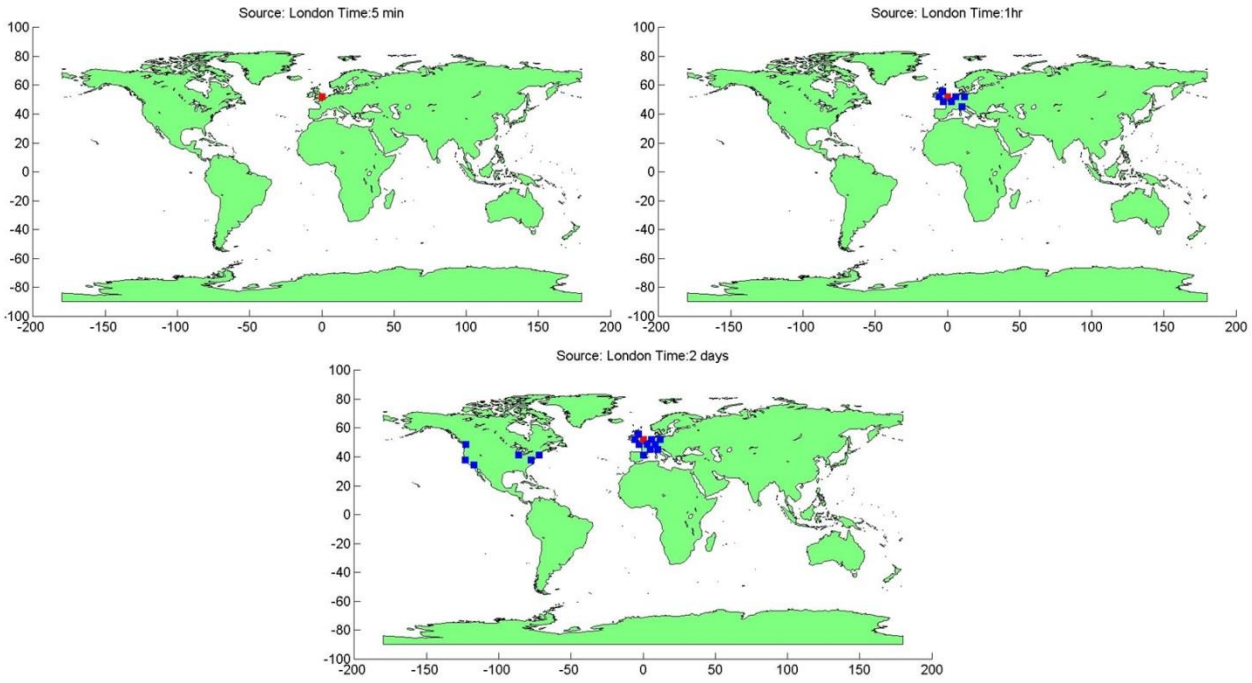

Figure 6: Graphical depictions of our learned model. The source bin is the bin containing London, England.

Popular Destination (Actual)	Desirable Destinations [1]	Desirable Destination (Our Model)
London, GB	London, GB	New York
New York, US	New York, US	London
San Francisco/San Jose, US	Brussels, BE	San Francisco
Paris, FR	San Francisco/San Jose, US	Seattle
Milan, IT	Paris, FR	Washington D.C.
Washington DC/Baltimore, US	Frankfurt, DE	Vancouver
Vancouver, CA	Sydney, AU	Los Angeles
Chicago, US	Melbourne, AU	Chicago
Los Angeles, US	Tokyo, JP	Milan
Brussels, BE	Dublin, IE	Glasgow
Berlin, DE	Shanghai, CN	Berlin
Tokyo, JP	Washington DC/Baltimore, US	Tokyo
Rome, IT	Berlin, DE	Naples
Glasgow, GB	Toronto, CA	Barcelona
Frankfurt, DE	Hilo, US	Amsterdam
Barcelona, ES	Marseille, FR	Paris
		Sydney

Table 1: Lists of popular locations, desirable locations based on [1], and our learned desirable locations

We can also take a look at the locations which are popular compared to what the model learns is desirable. The popularity of a destination can be estimated by counting the number of users who visit the destination. Our model allows us to also estimate the *desirability* of a destination: how popular the destination might have been if not for factors such as distance. For example, we see that Sydney is more desirable than it is popular: it is in the top 16 most-desirable destinations, but not in the top 16 most-popular destinations, due to it being far away from most population centers. At present, our model predicts 8 of the 16 most desirable destinations correctly in comparison to [1].

Finally, we will take a look at the cost per iteration to determine how optimized our model is. Figure 7 shows the Negative Log Likelihood (NLL) for each iteration of Batch Gradient Descent. As can be seen, the cost continuously spikes. The oscillations imply that near convergence is occurring at a local minima which means a better starting point must be chosen or Batch Gradient Descent is not suitable for optimizing the NLL. Regardless, this means our model is not being properly optimized which limits the accuracy of predicting travel patterns. In [1], the minimum NLL after training their model was 1.05×10^6 while our model, at best, would have a training NLL of approximately 5×10^6. We leave finding better optimization techniques to future work.

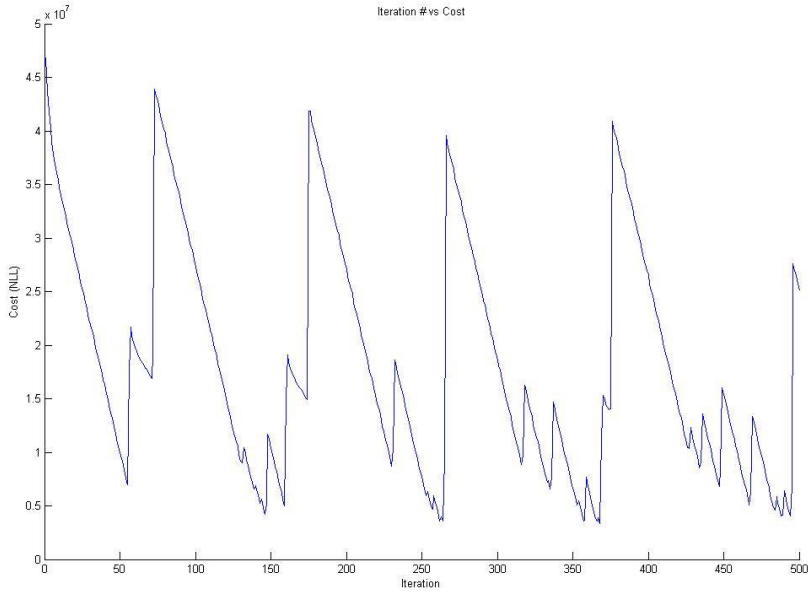

Figure 7: Plot of Iteration # vs Cost for Batch Gradient Descent

Parametric Models vs Histogram based Methods

The main reason for the improvement due to the parametric methods is that the histogram-based methods are very statistically inefficient, whereas the factored models can generalize.

1. Let us consider the set of parameters. Our model has 5486 parameters. These are the only values that need to be stored. In comparison, a histogram based method would require more than 2 million values to be stored even when taking advantage of sparsity of the data set.
2. Parametric models have been shown to have better generalization power than the histogram based methods. For example, analysis done by Guerzhoy et. al. [1] showed that the test NLL error for the histogram based method described in [2] was 5.27e05 while it was about 2.23e05 for the basic latent factor parametric model.
3. On a qualitative note, the histogram based methods require enormous datasets but reveal little about the underlying patterns. On the other hand, parametric models learn meaningful concepts which can be used to explain travel patterns.

Conclusion

Overall, compared to [1], our model exhibits "decent" predictive power. Our model is not optimal since we used Batch Gradient to learn our model. As mentioned above, by Guerzhoy et. al. [1] were able to achieve a more optimal solution using Conjugate Gradient Descent. We will leave better optimization as future work.

By implementing the travel model as a parametric model, we were able to spare quite a bit of memory and achieve a model which generalizes well. Using an empirical model would waste more space and would still perform worse than a parametric model [1]. The parametric travel model allows us to learn more meaningful factors as well.

Future Work

As mentioned previously, our model is not optimally optimized. As future work, we would like to explore replacing Batch Gradient Descent with a non-linear Conjugate Gradient Descent technique. Again, better optimization will yield a more powerful and accurate model.

We could also incorporate affinity between two locations as another latent factor [1]. For example, we expect that people are more likely to stay in their own countries than to cross borders, and, when leaving their country, they are more likely to visit countries where the same language is spoken as in their home country.

Furthermore, one could also adopt a Clustered Model [1] which can be used to cluster individuals based on prior travel habits. We can then assume that each individual travels according to the parameters of one cluster. The currently implemented model is a global model and cannot predict individual travel patterns.

Finally, we would like to experiment with adding a factor for "season" of travel to a location's desirability. We believe that time of travel has a big impact on where people travel and thus we would like to determine if such a factor can be learned.

References

[1] M. Guerzhoy and A. Hertzmann. Learning latent factor models of human travel. In *NIPS Workshop on Social Network and Social Media Analysis: Methods, Models and Applications*, 2012.

[2] E. Kalogerakis, O. Vesselova, J. Hays, A. Efros, and A. Hertzmann. Image Sequence Geolocation with Human Travel Priors. *In Proc. ICCV, 2009*.

[3] T. Kurashima, T. Iwata, G. Irie, and K. Fujimura. Travel route recommendation using geotags in photo sharing sites. *In Proc. CIKM, 2010*.

[4] D. Brockmann, L. Hufnagel, and T. Geisel. The Scaling Laws of Human Travel. *Nature, 439(7075):462–465, 2006.*

[5] P. D. Hoff. Multiplicative latent factor models for description and prediction of social networks. *Comp.& Math. Org. Theory, 15:261–272, 2009.*