Where is the Soho of Rome?* Measures and algorithms for finding similar neighborhoods in cities

Géraud Le Falher[1†] Aristides Gionis[2] and Michael Mathioudakis[2]
[1] Inria Lille – Nord Europe, France
[2] Helsinki Institute for Information Technology (HIIT), Aalto University, Finland

geraud.le-falher@inria.fr, aristides.gionis@aalto.fi, michael.mathioudakis@hiit.fi

July 30, 2015

Data generated on location-aware social media provide rich information about the places (shopping malls, restaurants, cafés, etc) where citizens spend their time. That information can, in turn, be used to describe city neighborhoods in terms of the activity that takes place therein. For example, the data might reveal that citizens visit one neighborhood mainly for shopping, while another for its dining venues. In this paper, we present a methodology to analyze such data, describe neighborhoods in terms of the activity they host, and discover similar neighborhoods across cities.

Using millions of `Foursquare` check-ins from cities in Europe and the US, we conduct an extensive study on features and measures that can be used to quantify similarity of city neighborhoods. We find that the earth-mover's distance outperforms other candidate measures in finding similar neighborhoods. Subsequently, using the earth-mover's distance as our measure of choice, we address the issue of computational efficiency: given a neighborhood in one city, how to efficiently retrieve the k most similar neighborhoods in other cities. We propose a similarity-search strategy that yields significant speed improvement over the brute-force search, with minimal loss in accuracy. We conclude with a case study that compares neighborhoods of Paris to neighborhoods of other cities.

*According to our findings, in Trastevere.
†Work carried out at Aalto University.

1 Introduction

More and more people live in cities. That fact raises the challenge of making decisions about how to live in an increasingly complex environment (e.g., what neighborhood to live in, or work in, or visit as a tourist). Yet the plethora of data gathered from social networks, smart sensors and mobile devices offers the opportunity to make informed decisions about these issues. In this paper, motivated by the challenge of understanding urban environments, and seizing the opportunities created by geo-enabled social data, we address the problem of comparing neighborhoods in different cities.

The problem we study has applications to recommending locations in a city. Imagine a traveler planning a trip in a new city and deciding the neighborhood in which to book a hotel room: the methods developed in this paper allow to match each neighborhood in the new city with the most similar neighborhood in the traveler's home city, or any other city that the traveler wishes to compare. Such a comparison makes the traveler's choice easier and more intuitive.

Our methods are also applicable in the analysis of cities and urban planning. For instance, when applied to the neighborhoods of one city, our techniques allow to identify neighborhoods that are similar to each other, and thus help us understand the activity that takes place in each area, what are the hubs of different activities, how citizens experience their city, and how they utilize its resources.

Our approach is briefly described as follows. Using geo-enabled data from social-media platforms, we represent each venue with a feature vector, that accurately describes the characteristics and the overall activity of the venue. We then devise similarity measures between venues, as well as between neighborhoods, i.e., sets of venues that are geographically close to each other.

We address these two problems from a *metric-learning* point of view (Bellet, Habrard, and Sebban, 2013). We experiment with many different distance measures and with algorithms that aim to learn their parameters. To learn the parameters of the distance measures and select the optimal settings, we use ground-truth data, either present in the dataset or gathered from carefully-designed user studies.

The measure that is shown to perform best for the task of finding similar neighborhoods is the *earth-mover's distance* (EMD) (Rubner, Tomasi, and Guibas, 1998). EMD is known to be a robust measure—however, it is also expensive to compute. Motivated by this observation, we address the issue of computational efficiency. In particular, given a neighborhood R in one city, we ask how to find the k most similar neighborhood to R in another city (or a set of other cities) under EMD, and without performing brute force computation. We design a pruning strategy that yields significant speed improvement with minimal loss in accuracy.

Our study and our algorithms are based on extensive experimental evaluation in European and US cities, using activity logs gathered from `Foursquare`, a location-based social network. Yet our study can be enriched by many other types of data, such as transportation, weather, air quality, energy consumption, etc. Such an extension is left for future work.

Table 1: Number of check-ins and venues per city.

City	2010 Check-ins	2014 Check-ins	Venues
New York	408 584	369 562	67 363
San Francisco	104 363	77 294	15 580
Washington	75 984	80 274	13 168
Paris	32 952	67 382	17 231
Barcelona	21 448	44 462	11 351
Rome	9 364	24 128	7 691
14 others	643 810	988 671	206 285
Total	1 296 505	1 651 773	338 669

2 Datasets

Our dataset consists of geo-tagged activity logs from `Foursquare`. `Foursquare`is a popular location-based social network that, as of 2015, claims more than 50 million users.[1] It enables users to share their current location with friends, rate and review venues they visit, and read reviews of other users. `Foursquare` users share location information by generating "check-ins" using a dedicated mobile application[2]. Each check-in is associated with a web page that contains information about the user, the venue, and other details of the visit. Each venue is also associated with a public web page that contains information about the venue—notably its category such as *Food* or *Nightlife Spot*—and aggregates information from user check-ins.

According to `Foursquare`'s privacy policy, check-ins are private information. However, sometimes users opt to share their check-ins via `Twitter`, a popular micro-blogging platform. We were thus able to obtain `Foursquare` data by retrieving check-ins from `Twitter`, between March and July 2014. We focused on 10 US and 10 European cities, chosen for their high activity. In addition, we also used a previously released dataset (Cheng et al., 2011), with `Foursquare` check-ins generated between September 2010 and January 2011 in these cities. In total, our `Foursquare` data consists of almost 3 million check-ins, even though none of them took place in August. Details can be found in Table 1, where we focus on the 6 cities on which we carry on our main experimental evaluation.

Exploration

We highlight some high-level patterns we observe in our dataset, as they guide our later choice of features used to describe venues.

First, we cluster the venues in each city by the temporal distribution of associated check-ins during a day. Specifically, for each venue in one city we calculate the fraction of check-ins that occur at different times of the day, using 4-hour-long windows — i.e.

[1] According to *http://foursquare.com/about*.
[2] The Swarm application, *http://www.swarmapp.com*.

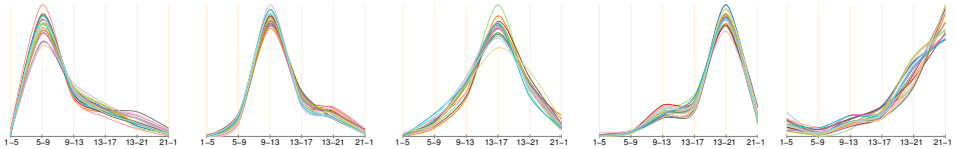


Figure 1: Results of k-means with $k = 5$ on the 4-hour-window check-in distribution of venues. Each line in each figure is the centroid of a cluster in a city. We observe that the centroids of the five clusters are almost identical in all cities.

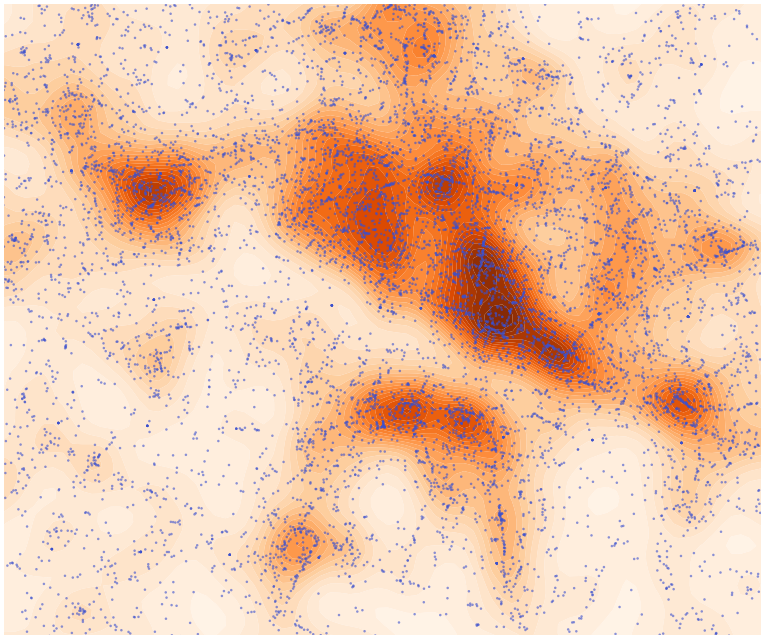


Figure 2: Venue density in Paris, computed by a Gaussian kernel.

the fraction of check-ins occurring from 1 am to 5 am, from 5 am to 9 am, and so on. We then perform k-means clustering on the computed distributions for all venues of each city, using $k = 5$. Figure 1 shows the $k = 5$ identified centroids with a different color for each city. We observe five clearly-separated clusters of venues in all cities, with clear peaks of activity within a day, that are also remarkably consistent across all cities and thus motivate this choice of k.

Then, we explore the geographic distribution of venues within a city. We observe that venues are not uniformly distributed within the city and that the density of nearby locations is a discriminative feature as well. For instance in Figure 2, we can clearly distinguish venues belonging to the city center from the others.

Finally, we notice that venues differ in the diversity of people who visit them—some having a large number of infrequent visitors, and others a smaller number of dedicated

visitors. To quantify the diversity of unique visitors for different venues, we use the measure of user entropy introduced by Cranshaw et al. (2010). Intuitively, a higher value of entropy means that check-in activity is shared among a larger number of users. Computing this value for all venues in Paris and Barcelona shows that it is a good descriptor of the "publicness" of a place. Touristic attractions like Sagrada Família and Eiffel Tower exhibit high entropy because they are visited by a large and diverse set of people. On the other hand, work offices and private houses have low entropy.

3 Data model and problem setting

In this section we introduce the notation and the problem setting we will be working with in the rest of the paper.

Cities and venues. We consider a set $\mathcal{C}$ of n cities, $\mathcal{C} = \{C_1, \ldots, C_n\}$. Each city contains a set of *venues*. A venue is a uniquely identified location that can be visited by individuals, like a restaurant, shop or park. We write $V(C)$ to denote the set of venues of the city $C \in \mathcal{C}$. We also use $\mathcal{V}$ to denote the set of all venues in all cities, i.e., $\mathcal{V} = \bigcup_{C \in \mathcal{C}} V(C)$. The description of each venue $v \in \mathcal{V}$ contains a geographic location $\mathrm{loc}(v)$, expressed as a pair of latitude and longitude coordinates.

Activities. We consider a set of *activities* $\mathcal{A} = \{a_1, \ldots, a_{|\mathcal{A}|}\}$, each one corresponding to an instance of user behavior in the cities $\mathcal{C}$. Each activity $a_j \in \mathcal{A}$ is represented by a tuple $a_j = (\mathrm{type}_j, \mathrm{u}_j, \mathrm{loc}_j, \mathrm{t}_j, \mathrm{descr}_j)$, where type_j is the activity type, u_j is a user-id, loc_j and t_j are the spatio-temporal coordinates of the activity, and descr_j provides additional information depending on the type of activity.

Although this framework supports a variety of sources, in this initial study we focus on single type of activities, namely one `Foursquare` user checking in at a particular location and time. Such check ins are naturally associated with a single venue.

Feature vectors. Given the set of activities $\mathcal{A}$ associated with venues in $\mathcal{V}$, we extract 30 *features* and describe each venue $v \in \mathcal{V}$ with a *feature vector* $\mathbf{f}(v)$ that contains information about associated activities. The features include the total number of check-ins at v, the number of unique users, entropy of the check-in distribution, number of likes, and number of visits/check-ins split down by day (weekday/Saturday/Sunday) and time of day (morning/noon/...). Additionally, since venues in a city are typically not considered in isolation, but location and context is very important, some features of the vector $\mathbf{f}(v)$ contain information about other venues around v, namely the distribution of the categories of venues around v. Specifically, the features involving the surrounding were computed from all others venues, weighted by a 2D Gaussian of arbitrary radius $r = 350$ meters. To make our data more reliable, we only consider venues with at least five check-ins by two different users.

Comparing venues. Given two venues $v_i, v_j \in \mathcal{V}$ with feature vectors $\mathbf{f}_i = \mathbf{f}(v_i)$ and $\mathbf{f}_j = \mathbf{f}(v_j)$, the simplest distance one can define between the two venues is the p-norm between their feature vectors $||\mathbf{f}_i - \mathbf{f}_j||_p$. This simple definition has a number of shortcomings, as it does not account for (i) the different scale of individual features, (ii) different scale due to variations in the cities, (iii) importance of features, and (iv)

potential correlations between features. One way to address the first two shortcomings is to ensure that each feature, aggregated at a city level, has mean equal to zero and standard deviation equal to one. To address the other two shortcomings, one well-motivated and principled approach is to consider the Mahalanobis distance

$$d_W(v_i, v_j) = ||\mathbf{f}_i - \mathbf{f}_j||_W = (\mathbf{f}_i - \mathbf{f}_j)^T W (\mathbf{f}_i - \mathbf{f}_j), \tag{1}$$

where W is the inverse of the covariance matrix of the features. Learning a more suitable matrix W, and thus the distance measure between venues, is the first challenge we face.

Objective 1 *Learn the matrix W so that the distance measure d_W captures best the human perception of similarity between venues.*

We approach this problem as a *metric-learning* problem (Bellet, Habrard, and Sebban, 2013), where we learn the matrix W with different methods to maximize consistency with a ground-truth labeling in our data.

City neighborhoods. Summarizing our discussion so far, each venue $v \in \mathcal{V}$ is described by the pair $(\text{loc}(v), \mathbf{f}(v))$, where $\text{loc}(v)$ and $\mathbf{f}(v)$ specify the location of the venue v, and its feature vector, respectively.

Now, given a city $C \in \mathcal{C}$, we define a *neighborhood* (or a *region*) R of the city C as a geographical region (a closed and connected set in the geographical plane). We abuse notation and we write $R \subseteq C$ to denote that R is a neighborhood of the city C. For a neighborhood $R \subseteq C$ we define $V(R)$ to be the set of venues of C that are contained in R, i.e.,

$$V(R) = \{v \in \mathcal{V} \mid \text{loc}(v) \in R\}.$$

We also define $F(R)$ to be the set of feature vectors of all the venues in R, that is,

$$F(R) = \{\mathbf{f}(v) \mid \text{loc}(v) \in R\}.$$

Comparing neighborhoods. Our objective is to define a meaningful distance measure between city neighborhoods. We want two neighborhoods to be similar, if they contain the same kind of venues, and in the same proportion. Thus, given two neighborhoods R_i and R_j we consider the feature vectors $F(R_i)$ and $F(R_j)$ of the venues contained the two neighborhoods, and we define the distance $\delta(R_i, R_j)$ between R_i and R_j by

$$\delta(R_i, R_j) = D(F(R_i), F(R_j)), \tag{2}$$

where D is a distance function between *sets of feature vectors*. Our second objective is stated as follows.

Objective 2 *Choose a distance measure δ between city neighborhoods as expressed by Equation (2), i.e., a distance measure D between sets of feature vectors of the venues in the two neighborhoods. The distance measure should capture best the human perception of similarity between city neighborhoods.*

In the next section we consider a number of different options for the distance function D, and we discuss our methodology for selecting the best one. Building on the optimal distance measure selected for our objective, we then consider the following *city-neighborhood search* problem.

Problem 1 *We are given a neighborhood R in a city $C \in \mathcal{C}$ and a subset of target cities $\mathcal{C}' \subseteq \mathcal{C}$. The goal is to find a small set of neighborhoods $\{R'_i\}_{i=1}^{k}$ in some city $C' \in \mathcal{C}'$ so that the distances $\delta(R, R'_i)$ are minimized.*

Two interesting special cases of Problem 1 are (i) $\mathcal{C}' = \mathcal{C}$, search for the most similar neighborhood in all cities; and (ii) $\mathcal{C}' = \{C'\}$, search for the most similar neighborhood in a given city C'. The emphasis for Problem 1 is on computational efficiency, and the aim is to improve over brute-force search.

4 Measures and evaluation methodology

In this section, we describe methods to achieve the two objectives we set above.

4.1 Measures to compare venues

We consider the following approaches to compare venues in terms of their feature vectors:

ITML Information Theoretic Metric Learning (ITML) (Davis et al., 2007) is a metric-learning method that aims to learn the matrix W in Equation (1). Its input consists of the feature vectors of venues V, along with class labels. It learns a matrix W so that the following goals are achieved: (i) the distance between two feature vectors of the same class is less than a (low) threshold t_{low}, (ii) the distance of vectors of different class exceeds a (high) threshold t_{high}, and (iii) matrix W is as close to the identity matrix as possible (in order to provide regularization). The two thresholds, t_{low} and t_{high}, are set automatically to the 5^{th} and 95^{th} percentiles of all pairwise euclidean distances of the original feature vectors.

We use the *category* of venues provided by `Foursquare`[3] as the class of each feature vector. To summarize, this method learns a matrix W that places venues of the same category in close distance of each other and venues of different categories at large distance from each other.

LMNN Large Margin Nearest Neighbor (LMNN) is also a metric-learning method that aims to learn matrix W in Equation (1). It accepts the same input as ITML (i.e. feature vectors of venues, classified according to their `Foursquare` categories), but unlike ITML, it optimizes an unconstrained cost function, defined so that, for the learned matrix W, feature vectors of the same class are as close as possible, while vectors of different class are as far as possible. In our study, we use Gradient Boosted LMNN, a state-of-the-art variant proposed by (Kedem et al., 2012).

[3] We use the top level of `Foursquare`'s category hierarchy.

t-SNE This method embeds feature vectors to a 2-d plane via t-distributed stochastic neighbor embedding (t-SNE), a state-of-the-art dimensionality-reduction method (Maaten and Hinton, 2008), and computes the Euclidean distance on the projected space

Euclidean Finally, we consider the EUCLIDEAN distance in the original space. Note, again, that feature values have been centered and normalized, so that each feature has zero mean and standard deviation equal to one at a city level.

4.2 Measures to compare neighborhoods

We proceed with addressing Objective 2, that is, selecting a distance measure between sets of feature vectors, in order to evaluate similarity between city neighborhoods. We consider the following options.

1. Earth mover's distance. EMD is a standard distance for vector sets that measures the *total amount of work* needed to transform (move) one vector set (total mass) to the other (Rubner, Tomasi, and Guibas, 1998). In our case, we have two neighborhoods X and Y made of feature vectors $X = \{x_i\}_{i=1}^{n}$ and $Y = \{y_j\}_{j=1}^{m}$. Assuming the distance between two vectors x_i and y_j is defined by an underlying metric as $d_{i,j}$, the distance between X and Y is the solution f of a bipartite flow problem formalized by this linear program:

$$\begin{aligned} & \min_{f} \textstyle\sum_{i,j} d_{i,j} f_{i,j} \\ \text{subject to} \quad & \textstyle\sum_j f_{i,j} = w_{x_i} \qquad \sum_i f_{i,j} = w_{y_j} \\ & \textstyle\sum_{i,j} f_{i,j} = \sum_i w_{x_i} = \sum_j w_{y_j} = 1 \end{aligned}$$

Each vector is assigned a weight (their sum being 1 in each neighborhood) and $f_{i,j}$ represents the amount of mass we bring from x_i to y_j The constraints express the requirement to move all the mass from one side to the other while minimizing the total work needed to move it. We find experimentally that uniform weighting gives more accurate results.

We experiment with the following variants of EMD: (i) using as the underlying metric the distance learned with ITML (EMD-ITML); (ii) using as the underlying metric the distance learned with LMNN (EMD-LMNN); (iii) using t-SNE as the underlying metric (EMD-t-SNE); (iv) using as the underlying metric the Euclidean distance (EMD-EUCL); (v) using as the underlying metric the Euclidean distance and requiring only a certain fraction of the smaller feature vector set ($F(X)$ or $F(Y)$) to be matched (EMD-PARTIAL). The rationale of EMD-PARTIAL is to provide more flexibility by allowing for two neighborhoods to have a fraction of venues that are completely different. For the fraction of vectors to match, we used 80% in our experiments.

2. Jensen–Shannon divergence. JSD is a symmetrized and smoothed version of the Kullback–Leibler divergence, a function for measuring distance between distributions. JSD can be computed for multivariate distributions, however, in our setting we have a relatively small number of samples (a typical neighborhood contains around 100 venues, and never more than 700) and a high dimensions (30) so we cannot estimate an accurate joint probability distribution. To account for this problem, we opt for computing the

JSD independently for each feature. In particular, we compute $\text{JSD}_1(F^{(i)}, G^{(i)})$, where JSD_1 is univariate JSD, while $F^{(i)}$ and $G^{(i)}$ are the distributions of the i-th feature for the vector sets F and G, respectively. We then aggregate over all features by

$$\text{JSD}(F, G) = \sum_i \theta_i \cdot \text{JSD}_1(F^{(i)}, G^{(i)}), \tag{3}$$

with $\sum_i |\theta_i| = 1$. To calibrate θ_i, we sample different neighborhoods and label each pair of neighborhoods as "similar" or "not similar." The labeling of the neighborhoods is based on our ground-truth, which we describe shortly. We then compute θ_i to maximize the sum of the JSD values over similar pairs minus the sum of JSD values over non similar pairs.

3. Minimum cost matching distance of set centroids. A very simple way to compute a distance between two sets of feature vectors is to compute the *centroid* of each set and then compute the distance between the two centroids. We extend this simple definition with k centroids. Given a set of feature vectors we perform k-means clustering, and represent the set with k centroids. Then, given two sets of feature vectors, and their corresponding k-set centroids, we compute the distance of a min-cost matching, using the Hungarian algorithms (Munkres, 1957). We experiment with different values of k and report the best results, obtained for $k = 3$.

Evaluation methodology. We now describe our evaluation process for selecting the best function for measuring distance over feature vector sets. Consider a neighborhood R in a source city C, and a target city C'. Assume that we have obtained k ground-truth neighborhoods $R_1, \ldots, R_k$ in C', which are the most similar to R. For example, if R is a neighborhood with many offices, companies, and financial services in C, so are neighborhoods $R_1, \ldots, R_k$ in C'.

Given a distance measure δ we want to evaluate, we can then compute the distance $\delta(F(R), F(R'))$ for each possible neighborhood R' of C' and rank all those neighborhoods in order of increasing distance. We can evaluate the quality of this ranking by checking the position that the ground-truth neighborhoods $R_1, \ldots, R_k$ appear in the ranking—if appearing at all. The higher we find a match, the better the ranking, and thus, the better the distance measure δ.

Since we do not have any a-priori neighborhood boundaries (and in fact we do not want to use any, since a neighborhood may be defined in a dynamic way, different than what administrative boundaries would give), any subset of venues that corresponds to a closed and connected region is a candidate neighborhood. As there are exponentially many such subsets, we restrict our search to regions of a certain shape. We consider neighborhoods R' to be *circles* (v', r), centered at a venue v' and with radius r. We take as v' regularly spaced venues in C' and $r \in \{200, 275, 350, 425, 500\}$ meters, with the additional constraint that the resulting circle should contain at least 20 venues.

After ranking all possible such circular neighborhoods R' in order of increasing distance $\delta(F(R), F(R'))$ we remove overlapping neighborhoods (in an Eratosthenes-sieve way) so that the resulting ranking does not contain overlapping neighborhoods.

To evaluate the resulting ranking, we need a *relevance score* for each neighborhood R' in the ranking with respect to the ground-truth neighborhoods $R_1, \ldots, R_k$. Note

that R' may not be identical to any of the ground-truth neighborhoods (for one, R' is circular, while the ground-truth neighborhoods can have arbitrary shapes) but there may have significant overlap with some of them. To account for such overlap, we define the relevance of each R' as the best *overlap* of R' with a ground-truth neighborhood $R_1, \ldots, R_k$, where the overlap is measured using the *Jaccard coefficient* on the sets of venues of two neighborhoods.

$$\text{rel}(R' \mid R_1, \ldots, R_k) = \max_{i=1}^{k} \frac{|V(R') \cap V(R_i)|}{|V(R') \cup V(R_i)|},$$

Having assigned a relevance score for each neighborhood R' in the ranking, we evaluate the quality of the ranking using *discounted cumulative gain* (DCG) Sakai (2007). The gain is a measure of relevance and we accumulate them (or sum them) but discount results that came too far in the ranking according to

$$\text{DCG} = \sum_{i=1} \frac{2^{rel(R_i)} - 1}{\log_2(i+1)}.$$

Evaluation results. To perform the evaluation described above we need ground-truth information regarding neighborhoods in cities. Since ground truth requires concrete knowledge of a city, we conducted a user study. Our study involved six cities: Barcelona, New York, Paris, Rome, San Francisco, and Washington DC. The participants in the study are international friends and colleagues of ours, who have lived for many years in at least one of those cities. The participants were given textual description of neighborhoods and were asked to mark some neighborhoods in their own city that matched best that description. The descriptions are shown in Table 2. The table also shows matching neighborhoods for Paris, as chosen by the first author of this paper who happens to have lived in that city. The participants provided their answers via a graphical interface.[4] The answers were curated, so that if more than one person provided answers for one city, the answers were merged. To give a better sense of the typical size and shape of these neighborhoods, we present in Figure 3 three of them in Paris and in Barcelona.

Starting with each of the 6 cities as a source city and for each of the 8 query neighborhoods, we compute the most similar neighborhoods in all other cities, using each of the distance measures that we want to evaluate.[5] The results are shown in Table 3. Each row corresponds to a source city. DCG scores are averaged over all target cities and all 8 query neighborhoods.

We see that EMD-Eucl is the best-performing measure, while JSD and EMD-t-SNE perform rather poorly. EMD-LMNN performs only slightly worse than EMD-Eucl. Another observation is that the absolute DCG scores of all measures are relatively low. One reason for the low scores is that many neighborhoods in the ground truth have

[4] *http://where-would-you.herokuapp.com/*

[5] Note that we experiment with 203 queries instead of $6 \times (6-1) \times 8 = 240$, as in a couple of cases (pairs of city - neighborhood description) our study did not yield a ground-truth neighborhood with enough venues inside.

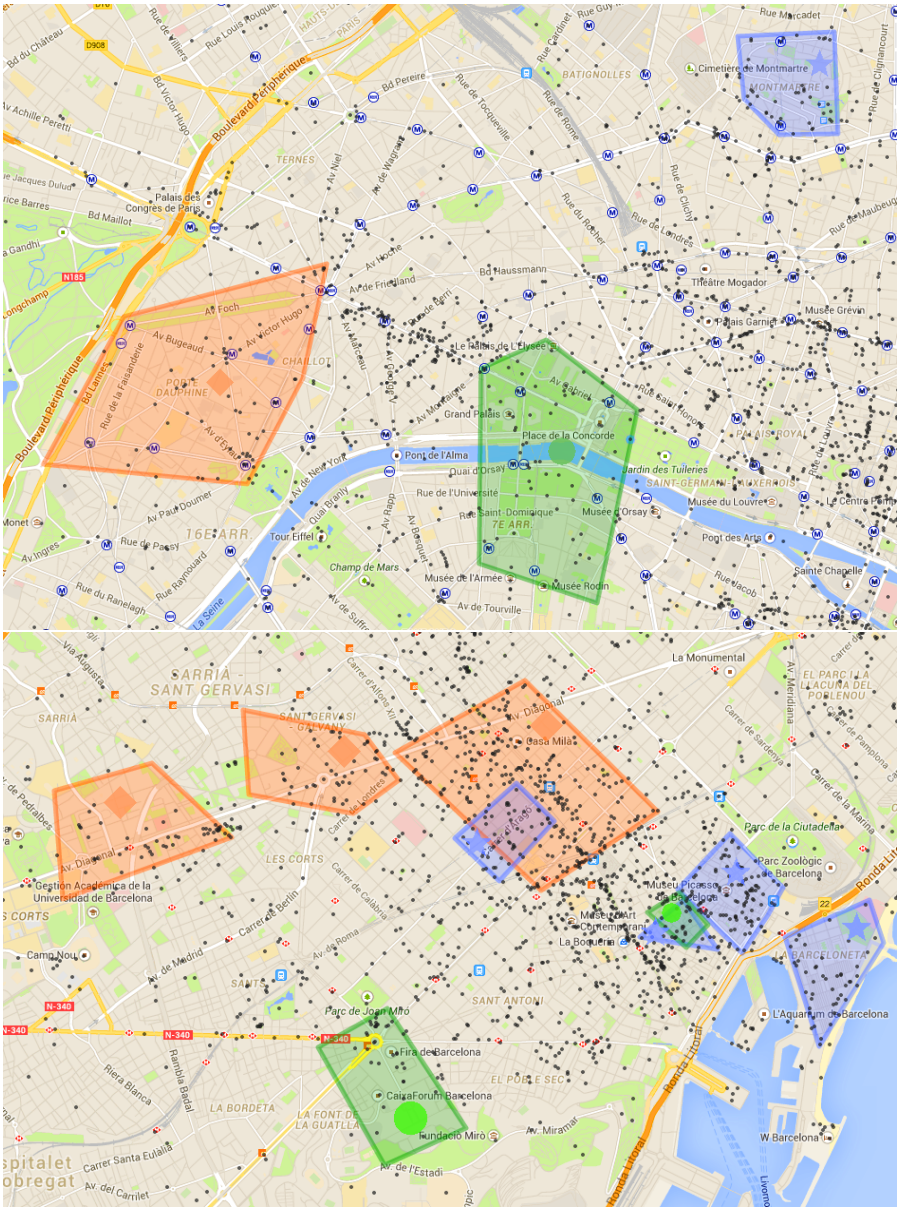


Figure 3: Three neighborhoods selected in Paris (top) and the corresponding annotations from Barcelona (bottom) experts: 16$^{\text{th}}$ *arrondissement* ◆ Montmartre ★ , and Official ● . The black dots on each map represent the venues of our dataset.

Table 2: Neighborhood description used in user study and corresponding Paris neighborhoods. Participants in the study were asked to identify up to 5 most similar districts in their own city.

1 Fashion shops, luxurious places	Golden triangle
2 College & student neighborhood	Quartier Latin
3 Red light district	Pigalle
4 Touristic and artsy district	Montmartre
5 Government buildings	Official
6 LGBT neighborhood	Le Marais
7 Expensive residences	16^{th} *arrondissement*
8 Parks & leisure	The banks of Seine

Table 3: Average score of each metric from a given city. The best metric in each city is **highlighted** and the last row is the average score over all cities.

Query Source	Min cost matching	EMD-Ucl	EMD-LMNN	EMD-ITML	EMD-*t*-SNE	JSD	EMD-Partial
Barcelona	.083	.078	**.084**	.033	.028	.042	.078
New York	**.059**	**.059**	**.059**	.049	.026	.057	.053
Paris	.061	**.091**	.078	.021	.044	.045	.061
Rome	.024	.042	.039	**.055**	.038	.021	.029
San Franc.	.045	.045	.040	**.060**	.042	.033	.044
Wash. DC	**.043**	.034	.038	.035	.026	.033	.038
Average	*.052*	***.058***	*.056*	*.042*	*.033*	*.038*	*.051*

non-circular shapes (e.g., a long street of luxurious shops). Thus, even if our measures discover an area very close to the ground truth, due to its circular shape it can have low overlap with the ground truth, and low relevance score.

5 Searching for similar neighborhoods efficiently

We now turn our attention to the efficiency aspects of the neighborhood similarity-search problem, i.e., the Problem 1 defined in page 7. Following our evaluation from the previous section, we focus on the EMD distance. The brute-force approach to solve this problem is an exhaustive search algorithm, as the one used above for the evaluation of neighborhood distance measures. Namely, given neighborhood R, consider all candidate neighborhoods R' of a certain shape (circle, rectangle, or other) in the target city C', evaluate the distance $\text{EMD}(F(R), F(R'))$, and return the neighborhood that achieves the smallest distance.

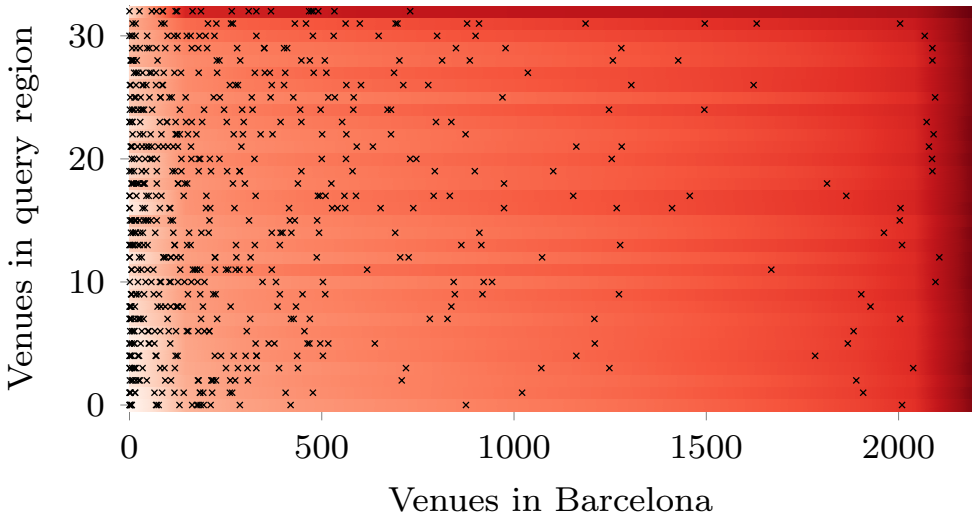


Figure 4: Intuition behind our pruning strategy: for two neighborhoods with small EMD, the venues of one neighborhood are in the k-NN set of the venues of the other.

In this section, we show how to speedup the search task significantly, with very little loss in accuracy. Our solution relies on the following observation: the EMD between two sets of feature vectors $F(R) = F$ and $F(R') = F'$ is zero, if all feature vectors in F and F' coincide. Relaxing this condition, the EMD is small, if for many vectors in F there is some *near* vector in F'. Put differently, when one feature vector in F is *far away* from all vectors in F', it contributes a large cost to EMD.

Therefore we can reduce the search space by preprocessing all venues in the target city and keeping only those venues whose feature vectors are close to feature vectors of venues in the query neighborhood. The venues kept in this preprocessing step can be used as *anchors*. We can then look for areas in the target city that are dense in anchor venues, and group them in candidate neighborhoods, for which we calculate the actual EMD.

To see how this idea works, consider Figure 4, where we search in Barcelona to find the neighborhood that is most similar to Pigalle (Paris). Each row in the figure corresponds to one venue v in Pigalle, and contains the ranking of all venues in Barcelona sorted by distance to v. There is a cross (`x`) in i-th position of the ranking if the i-th ranked venue in Barcelona belongs to the ground truth neighborhood (in this case, el Raval, which we know from our user study). We see that if we restrict ourselves to the 100 nearest neighbors of each venue in the query neighborhood, we recover most of the venues in the ground-truth neighborhood.

Our algorithm works as follows. Starting from the query neighborhood R and target city C' (or cities), we obtain the set of k-nearest neighbors $N_k(v) \subseteq V(C')$ for each venue $v \in V(R)$. All venues found in at least one k-NN set form the set of anchor

venues $V_A = \bigcup_{v \in V(R)} N_k(v)$. The set of anchor venues V_A is then treated with respect to its geographic coordinates: the DBSCAN algorithm is applied and areas with low density in anchor venues are discarded. DBSCAN also produces a clustering of venues, which are treated as candidate neighborhoods. For the candidate neighborhoods the exact EMD is computed. If a candidate neighborhood is too large, the exact EMD is computed for sliding subareas. Finally, to account for misses that may happen during the pruning phase, each area considered is extended by adding to its radius a distance of $j \times 50$ meters, $j = 0, \ldots, \ell$, and the extended area is also treated as a candidate neighborhood. In the end of the process, the algorithm returns the the neighborhood with the smallest distance (or top-m smallest distances).

The two parameters of the algorithm, k and ℓ, offer an accuracy vs. efficiency trade-off. Our experiments in the next Section , as detailed in Section 6, suggest that the algorithm produces very accurate results even for small values of the parameters (which also give the highest efficiency). With respect to k, we found that in our data, $k = 50$ works very well, as in most cases, this value returns around 50% of the venues in the ground truth, while covering only 33% of the city. With respect to ℓ, even $\ell = 0$ (no extension) works quite well. While the robustness of EMD (and more generally optimal transport based distance) has recently lead to active research about speeding up their computation (e.g. Pele and Werman, 2009; Cuturi, 2013; Tang et al., 2013), we find that our simple filtering approach is more than adequate for our purposes.

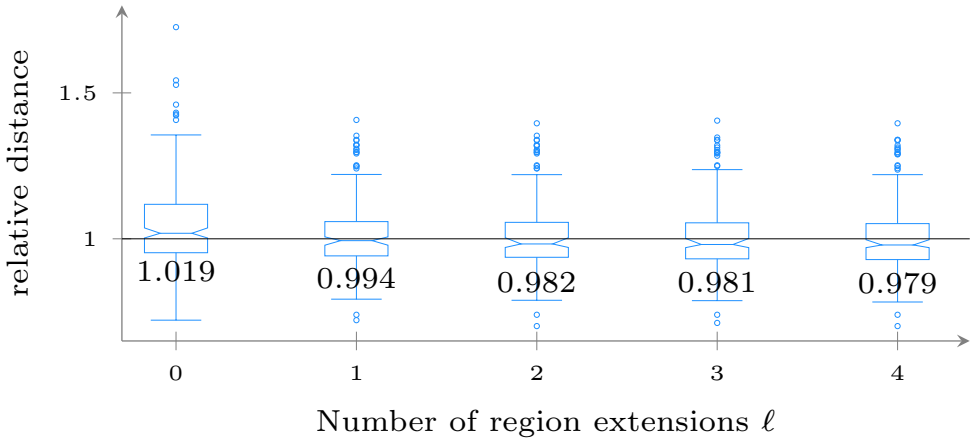
6 Scalability experiments

In this section we quantify how well our proposed method approximates a brute-force EMD search. We conduct our performance evaluation on the 203 query triplets (C, R, C') that were used in the analysis shown in Table 3.

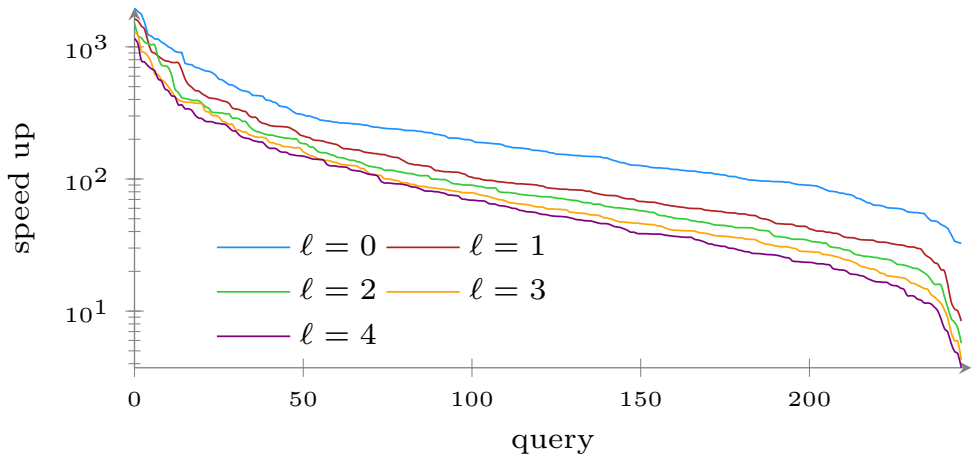
For each of these queries, we execute the brute-force search ; namely, we compute the EMD for all circles (v', r) centered at a venue v' of C' and with radius $r \in \{200, 275, 350, 425, 500\}$ meters. We also execute the neighborhood similarity-search algorithm, described in the previous section. We compare these two methods in terms of *execution time* and *quality* of solution found. In particular, given a query triple (C, R, C') let R_{BF} be the most similar neighborhood found by the brute force and let R_A be the most similar neighborhood found by the approximation method. Let the corresponding closest distances be $D_{BF} = \text{EMD}(F(R), F(R_{BF}))$ and $D_A = \text{EMD}(F(R), F(R_A))$, respectively. We define the *relative distance* ρ for that query as

$$\rho = \frac{D_A}{D_{BF}} \tag{4}$$

The smaller is ρ, the better the approximation. We would in fact expect that ρ is greater than 1, as values less than 1 indicate that the approximation method is better than the brute force. However, as the brute force is constrained to circular neighborhoods, it is possible that the approximation method finds better solutions. Removing this constraint from the brute force implies searching over other shape families (rectangles, diamonds, etc.), which will increase its running time even more.



(a) Box plot of ρ defined in Equation 4 as ℓ varies. Values smaller than 1 indicate that the approximation method finds a neighborhood with smaller distance than the brute force.



(b) Time speed-up for each query, in descending order. Original brute force searches take between 19 and 2129 seconds. Note the logarithmic scale.

Figure 5: Approximation performance.

Table 4: First, second and third quantile of relative distance as k varies.

k	8	25	50	80	160
1	0.942	0.940	0.942	0.946	0.944
2	1.003	0.998	0.994	0.994	0.986
3	1.093	1.070	1.059	1.067	1.061

Overall, our results show that for the range of parameters we experiment with, the approximation method is much faster than the brute force—in most cases by at least one order of magnitude, while often by two or even three. At the same time the relative distance is very close to 1, often below 1, and rarely above 1.5.

In more detail, we first analyze the effect of ℓ, the number of times we extend the initial regions found after clustering, while using $k = 50$. As ℓ increases, more EMD computations are required, but the chances to find a more similar region increase. Indeed, as we see in Figure 5a, the relative distance decreases as ℓ increases, while the computation becomes more expensive (Figure 5b). We also note that after $\ell = 1$, the gain is small, suggesting that the initial regions are already relevant enough.

We perform the same experiment for $k \in \{8, 25, 50, 80, 160\}$ with $\ell = 1$. As shown in Table 4, the relative distance is very small for all values of k, showing the robustness of the method. At the same time, k does not affect much the running time of the algorithm (results not shown), as the bottleneck is the computation of EMD.

7 Empirical Study

A natural way to assess our method is to compare its results with *randomly selected neighborhoods*. In this section, we employ our approximate search algorithm to find matching neighborhoods between cities, compare them with randomly selected ones, and report indicative results.

Specifically, we run approximate search on the 203 query triplets (C, R, C') that were used to generate Table 3. For each query triplet, we retrieve the list of top $k = 5$ neighborhoods returned by approximate search, order them by distance to the query neighborhood, and measure their overlap with the ground-truth neighborhoods in target city C'. Moreover, for each query triplet, we generate 2000 lists of randomly selected and ordered neighborhoods in C', each of size $k = 5$. To produce a random list of $k = 5$ neighborhoods, we first sample a venue from city C', and then select a neighborhood around it of size approximately equal to the query neighborhood, repeating the process $k = 5$ times.

Subsequently, we measure the overlap of each result list with the corresponding ground-truth neighborhoods in the target city. To measure overlap, we again use the DCG score of result lists against the ground-truth regions.

In Table 5, we report the percentile at which our method's DCG score ranks compared to the random neighborhoods. Higher percentile means that our result lists have higher overlap with ground-truth than more random neighborhoods. In most cases, it performs significantly better than random. Even when we miss completely the ground truth

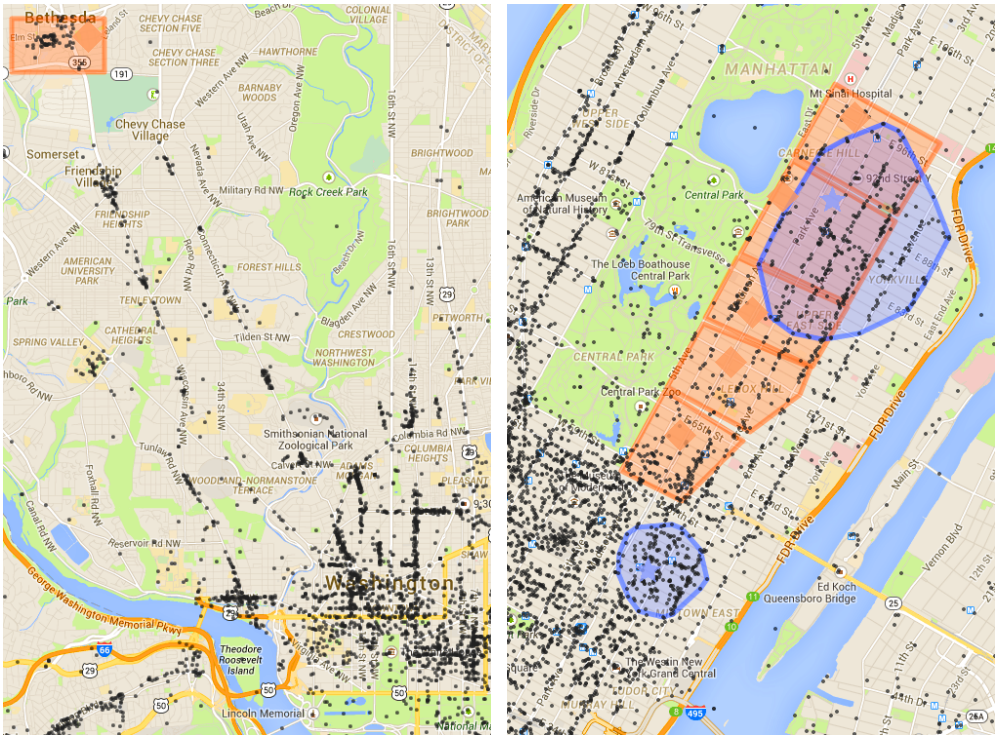


Figure 6: A query in Washington and its result in New-York.

(entries marked with grey font), the same happens with a large fraction or even the majority of the 2000 random lists – which means that the ground truth is intrinsically difficult to recover because of its size or location in the city.

Examples To illustrate the results produced by our method, we present examples of findings, produced by the best-performing measure, EMD-EUCL.

For the first example, we select Bethesda as our query neighborhood. Its location is shown in Figure 6-left ◆. Bethesda is a neighborhood in Washington D.C. and was identified in our user-study as a neighborhood with 'expensive residences' (Table 2). Subsequently, we ask to find the most similar neighborhoods in New York City. The most similar neighborhoods suggested by our method are shown in blue in Figure 6-right (★), and they coincide with lower- and upper-east-side Manhattan. Note that east-side Manhattan was identified in our user study as the neighborhood with 'expensive residences' (◆), which indeed is an area of high-cost apartments. This is especially encouraging since our `Foursquare` data does not include any information about price.

For the second example, we submit two queries. As a first query, we select Montmartre, the "touristic and artsy" neighborhood of Paris, famous for the Basilica of the Sacred Heart, which sits on top of the highest point in the city and thus offer a stunning view to million of tourists. Many artists have worked here, such as Monet, van Gogh, Picasso,

and Dalí, which matches its bohemian atmosphere. As a second query, we pick the *La Ribera* district in Barcelona, identified as a "touristic and artsy" in our user study. Indeed, it is a popular place for expats and home to many art boutiques. Subsequently, we ask for the most similar neighborhood in Rome. The most similar neighborhood is Trastevere and is shown in Figure 7. Indeed it is a neighborhood in Rome that is known to match the description of 'touristic and artsy' neighborhood. Furthermore, we observe that besides Trastevere, both queries returned the same areas of Rome, a good indication that the aforementioned neighborhoods share similar features.

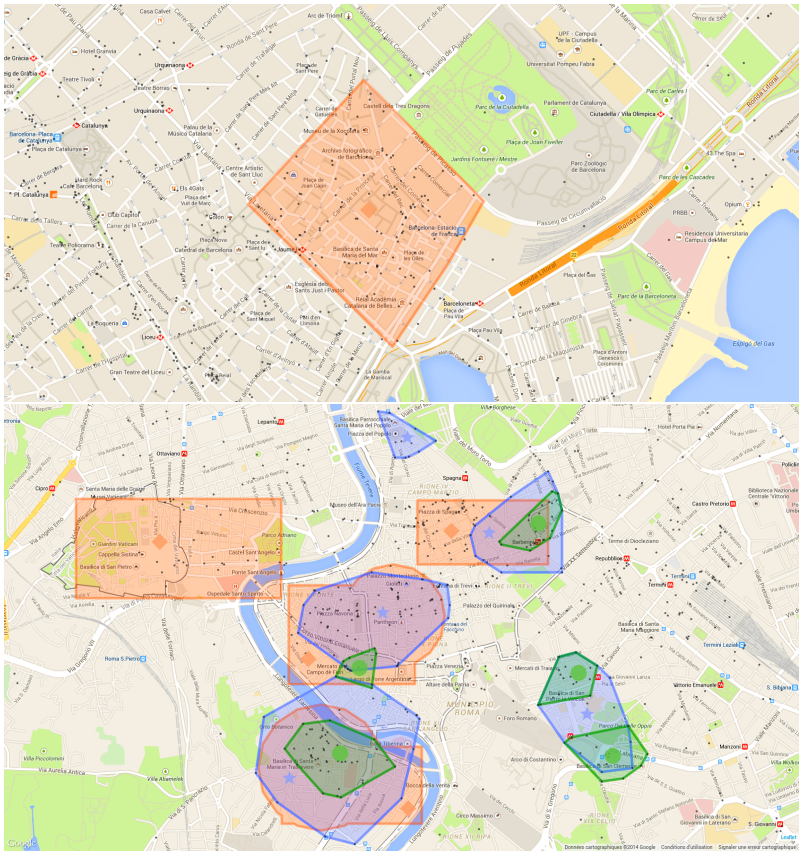


Figure 7: Top: the neighborhood query in Barcelona (La Ribera). Bottom: the results in Rome match the ground truth (in orange), both from Barcelona (green) and Paris (blue, the query is showed in blue in Figure 3).

Table 5: Percentile rank of the result obtained by our method compared with those obtained by chance. Rows refer to the neighborhood types described in target city, abbreviated as follow BR: Barcelona, NY: New York, PR: Paris, RO: Rome, SF: San Francisco and WS: Washington. For instance, the top-left number means that if we look in Barcelona for an area that is similar to the one chosen in New York by our expert to match the "luxurious shops" description, our method's result overlap more with the ground truth than 68.7% of the random samples. "—" denote queries not run due to lack of suitable ground truth. When our result has no overlap at all (in grey), the percentages indicate the fraction of random samples that fail likewise, even though our result is not better.

neighborhood	Barcelona					New York					Paris				
fashion shops	NY	PR	RO	SF	WS	BC	PR	RO	SF	WS	BC	NY	RO	SF	WS
& luxury	68.7	89.4	73.2	100.0	53.8	70.1	70.1	70.1	70.1	70.1	98.1	60.0	85.6	93.5	82.8
colleges	NY	PR	RO	SF	WS	BC	PR	RO	SF	WS	BC	NY	RO	SF	WS
& students	45.8	45.8	45.8	—	71.7	79.5	94.2	45.9	—	45.9	81.8	84.5	81.8	—	81.8
red light	NY	PR	RO	SF	WS	no ground truth for this query					BC	NY	RO	SF	WS
	—	91.6	—	85.6	69.0						93.7	—	—	93.5	78.0
touristic	NY	PR	RO	SF	WS	BC	PR	RO	SF	WS	BC	NY	RO	SF	WS
& artsy	32.4	91.1	13.9	75.1	91.0	90.3	87.9	49.2	49.2	49.2	81.2	81.2	81.2	81.2	81.2
government	NY	PR	RO	SF	WS	BC	PR	RO	SF	WS	BC	NY	RO	SF	WS
	49.5	99.7	78.8	51.9	99.6	87.2	87.2	87.2	95.0	87.2	74.1	74.1	74.1	77.5	74.1
LGBT	NY	PR	RO	SF	WS	BC	PR	RO	SF	WS	BC	NY	RO	SF	WS
	75.1	84.0	—	79.8	66.1	77.0	95.0	—	50.7	60.7	36.9	55.6	—	56.8	86.1
expensive	NY	PR	RO	SF	WS	BC	PR	RO	SF	WS	BC	NY	RO	SF	WS
residences	96.2	36.8	8.7	8.7	99.7	95.2	72.8	72.8	72.8	99.8	87.9	87.9	95.6	97.2	87.9
leisure	NY	PR	RO	SF	WS	BC	PR	RO	SF	WS	BC	NY	RO	SF	WS
	95.2	98.8	95.2	96.4	—	—	81.3	98.2	81.3	—	—	99.6	99.6	99.6	—

neighborhood	Rome					San Francisco					Washington				
fashion shops	BC	NY	PR	SF	WS	BC	NY	PR	RO	WS	BC	NY	PR	RO	SF
& luxury	100.0	97.2	99.1	99.2	89.6	78.2	40.5	91.6	58.0	48.7	91.8	95.1	91.8	91.8	91.8
colleges	BC	NY	PR	SF	WS	BC	NY	PR	RO	WS	BC	NY	PR	RO	SF
& students	28.0	28.0	28.0	—	28.0	95.6	95.6	95.6	95.6	95.6	39.0	—	95.5	50.9	—
red light	no ground truth for this query					BC	NY	PR	RO	WS	BC	NY	PR	RO	SF
						92.2	—	95.0	—	36.9	51.2	—	51.2	—	51.2
touristic	BC	NY	PR	SF	WS	BC	NY	PR	RO	WS	BC	NY	PR	RO	SF
	83.3	0.4	90.0	70.5	87.6	8.9	9.9	8.9	8.9	74.8	58.9	95.3	89.3	99.5	47.9
government	BC	NY	PR	SF	WS	BC	NY	PR	RO	WS	BC	NY	PR	RO	SF
	1.1	4.5	11.5	5.9	11.7	73.2	—	95.0	73.2	92.5	83.2	—	97.0	62.6	46.1
LGBT	BC	NY	PR	SF	WS	BC	NY	PR	RO	WS	BC	NY	PR	RO	SF
	90.5	90.5	90.5	90.5	90.5	96.9	93.4	76.5	—	76.5	88.3	78.6	86.5	—	57.6
expensive	BC	NY	PR	SF	WS	BC	NY	PR	RO	WS	BC	NY	PR	RO	SF
residences	82.7	77.3	77.3	77.3	84.2	68.2	68.2	68.2	68.2	73.7	67.3	66.4	73.9	67.0	66.4
leisure	BC	NY	PR	SF	WS	BC	NY	PR	RO	WS	BC	NY	PR	RO	SF
	—	50.9	50.9	50.9	—	—	86.5	88.9	86.5	—	—	98.2	98.2	98.2	98.2

8 Related work

Our work lies within the scope of urban computing, an area of increasing interest. In one line of work, the concentration of social online activity is used to determine interesting geographic regions of cities. For instance, Deng, Chuang, and Lemmens (2009) use DBSCAN to cluster `Flickr` photos, and they exploit tag co-occurrence to characterize the discovered clusters. Rattenbury and Naaman (2009) also employ spatial methods to discover hotspot regions in San Francisco where certain photo tags appear in burst. Moreover, Wakamiya, Lee, and Sumiya (2012) used geolocated data from `Twitter` to detect sentiment and movement and draw a socio-cognitive map of the Kinki region in Japan.

Closer to our work, Cranshaw et al. (2012) analyze 18 million `Foursquare` check-ins to find so-called *livehoods*.[6] They build a spatial neighbor graph between venues, where edges are weighted by the cosine similarity of the user distribution of two venues, and then perform spectral clustering. Faced with the same difficulty as ours to evaluate their results, they interview residents of Pittsburgh who validate the resulting subdivisions. Another approach was proposed by Zhang et al. (2013), also based on `Foursquare` check-ins. Each venue is assigned a category, a peak activity time, and a binary label: touristic or not. The venues are then clustered along these features, and the city is divided into a regular grid, where grid cells are described by their feature density. Similar cells are then clustered into neighborhoods. With respect to the last two papers, our work makes a number of advances: different notions of similarity between neighborhoods are defined and evaluated, the problem of finding similar neighborhoods in other cities is considered, and the scalability of the search problem is addressed.

In addition to the analysis of static data like check-ins, another line of work takes advantage of dynamic data, such as trajectories. For instance, Giannotti et al. (2011) analyze GPS data in Italian cities to find temporal patterns, which can then be used for detect events or regulate traffic jams. Similarly, Cao, Cong, and Jensen (2010) extract "stay points" from car GPS data and assess the significance of each point by the number of visitors, time taken to reach there, and duration of stay. Uddin, Ravishankar, and Tsotras (2011) develop efficient methods to address closely-related tasks. Once the semantics of locations is known, it is still challenging to find frequent patterns efficiently (Zhang et al., 2014).

The problem of identifying and characterizing neighborhoods has also been addressed by companies. For instance, research in `Flickr` has shown that by computing the α-shape of a set of tagged photos, it is possible to recover neighborhood boundaries;[7] we remind that the α-shape is a generalization of the convex hull (Edelsbrunner, Kirkpatrick, and Seidel, 1983). Likewise, `Airbnb`, a social lodging renting website, has recently produced a ranking of cities by hospitality[8], as well as data-driven characterization of neighborhoods.[9] However, the methods and details for these commercial

[6] *http://livehoods.org/*
[7] *http://code.flickr.net/2008/10/30/the-shape-of-alpha*
[8] *http://nerds.airbnb.com/most-hospitable-cities*
[9] *http://airbnb.com/locations*

systems are not publicly available.

Finally we note that our setting is tangentially related to work on spatio-temporal topic modeling. In this line of work, Yin et al. (2011) address the problem of finding k localized topics and n Gaussian spatial regions. They develop and apply a *latent geographical topic analysis* framework: each region has a topic distribution and each topic is a multinomial distribution over all possible photos tags. By taking into account user information, such an approach can be used to provide recommendations (Kurashima et al., 2013). Similar ideas can be applied to finding localized tweets with a hierarchical topic model (Ahmed, Hong, and Smola, 2013). The main difference between these works and ours is that they focus on discovering salient regions rather than comparing them with each other. Going beyond words, Doersch et al. (2012) discover frequent and discriminative patches from Google Street images and use them, among other applications, to retrieve similar buildings across cities.

9 Conclusion

In this paper, we studied the problem of matching neighborhoods across cities using `Foursquare` data and considered various measures for their comparison. Evaluating against ground truth data, we found that EMD performs best and presented a method to enhance its running time with minimal loss of accuracy. Finally we illustrated the quality of results obtained by anecdotal evidence.

Although these results are encouraging, they could be extended in various directions:

- Use more data sources to derive more features (for instance photos sharing website, transportation, weather, air quality, energy consumption, demographics, etc).
- Match several (or all) neighborhoods at the same time (but without excessive overlapping) and thus obtain a similarity measure between whole cities.
- Provide a justification for the neighborhoods selected by the algorithms. For instance, design algorithms that provide explanations of the type "These two regions match because people take a lot of photos, go there on weekday from 4pm to 8pm and there are a lot cultural buildings." Another source of information is the resulting flow of EMD: looking at what kind of venues are close to what, and obtain insights from those matches.

References

Ahmed, A.; Hong, L.; and Smola, A. 2013. Hierarchical geographical modeling of user locations from social media posts. In *WWW*.

Bellet, A.; Habrard, A.; and Sebban, M. 2013. A survey on metric learning for feature vectors and structured data. Technical report, Université de Saint-Etienne.

Cao, X.; Cong, G.; and Jensen, C. 2010. Mining significant semantic locations from GPS data. *VLDB Endowment* 3(1-2):1009–1020.

Cheng, Z.; Caverlee, J.; Lee, K.; and Sui, D. 2011. Exploring millions of footprints in location sharing services. In *ICWSM*.

Cranshaw, J.; Toch, E.; Hong, J.; Kittur, A.; and Sadeh, N. 2010. Bridging the gap between physical location and online social networks. In *UbiComp*.

Cranshaw, J.; Schwartz, R.; Hong, J.; and Sadeh, N. 2012. The livehoods project: Utilizing social media to understand the dynamics of a city. In *ICWSM*.

Cuturi, M. 2013. Sinkhorn Distances: Lightspeed Computation of Optimal Transport. In Burges, C. J. C.; Bottou, L.; Welling, M.; Ghahramani, Z.; and Weinberger, K. Q., eds., *Advances in Neural Information Processing Systems 26*. Curran Associates, Inc. 2292–2300.

Davis, J.; Kulis, B.; Jain, P.; Sra, S.; and Dhillon, I. 2007. Information-theoretic metric learning. In *ICML*.

Deng, D.-P.; Chuang, T.-R.; and Lemmens, R. 2009. Conceptualization of place via spatial clustering and co-occurrence analysis. In *Workshop on Location Based Social Networks*.

Doersch, C.; Singh, S.; Gupta, A.; Sivic, J.; and Efros, A. A. 2012. What makes paris look like paris? *SIGGRAPH* 31(4).

Edelsbrunner, H.; Kirkpatrick, D.; and Seidel, R. 1983. On the shape of a set of points in the plane. *IEEE Transactions on Information Theory* 29(4):551–559.

Giannotti, F.; Nanni, M.; Pedreschi, D.; Pinelli, F.; Renso, C.; Rinzivillo, S.; and Trasarti, R. 2011. Unveiling the complexity of human mobility by querying and mining massive trajectory data. *The VLDB Journal* 20(5):695–719.

Kedem, D.; Tyree, S.; Sha, F.; Lanckriet, G.; and Weinberger, K. 2012. Non-linear metric learning. In *NIPS*.

Kurashima, T.; Iwata, T.; Hoshide, T.; Takaya, N.; and Fujimura, K. 2013. Geo topic model: Joint modeling of user's activity area and interests for location recommendation. In *WSDM*.

Maaten, L., and Hinton, G. 2008. Visualizing data using t-SNE. *JMLR* 9:2579–2605.

Munkres, J. 1957. Algorithms for the Assignment and Transportation Problems. *Journal of the Society for Industrial and Applied Mathematics* 5(1):32–38.

Pele, O., and Werman, M. 2009. Fast and robust Earth Mover's Distances. In *2009 IEEE 12th International Conference on Computer Vision*, 460–467. IEEE.

Rattenbury, T., and Naaman, M. 2009. Methods for extracting place semantics from Flickr tags. *TWEB* 3(1):1–30.

Rubner, Y.; Tomasi, C.; and Guibas, L. 1998. A metric for distributions with applications to image databases. In *ICCV*.

Sakai, T. 2007. On the reliability of information retrieval metrics based on graded relevance. *IPM* 43(2):531–548.

Tang, Y.; Leong Hou, U.; Cai, Y.; Mamoulis, N.; and Cheng, R. 2013. Earth mover's distance based similarity search at scale. *Proceedings of the VLDB Endowment* 7(4):313–324.

Uddin, M.-R.; Ravishankar, C.; and Tsotras, V. 2011. Finding regions of interest from trajectory data. In *MDM*.

Wakamiya, S.; Lee, R.; and Sumiya, K. 2012. Crowd-sourced cartography: Measuring socio-cognitive distance for urban areas based on crowd's movement. In *UbiComp*.

Yin, Z.; Cao, L.; Han, J.; Zhai, C.; and Huang, T. 2011. Geographical topic discovery and comparison. In *WWW*.

Zhang, A.; Noulas, A.; Scellato, S.; and Mascolo, C. 2013. Hoodsquare: Modeling and recommending neighborhoods in location-based social networks. In *SocialComp*.

Zhang, C.; Han, J.; Shou, L.; Lu, J.; and Porta, T. 2014. Splitter: Mining fine-grained sequential patterns in semantic trajectories. *VLDB Endowment* 7(9):769–780.