International Journal of Geo-Information

MDPI

Article

Exploring the Latent Manifold of City Patterns

Amgad Agoub * and Martin Kada

Institute of Geodesy and Geoinformation Science (IGG), Technische Universität Berlin, Kaiserin-Augusta-Allee 104-106, 10553 Berlin, Germany; martin.kada@tu-berlin.de
* Correspondence: amgad.agoub@tu-berlin.de

Abstract: Understanding how cities evolve through time and how humans interact with their surroundings is a complex but essential task that is necessary for designing better urban environments. Recent developments in artificial intelligence can give researchers and city developers powerful tools, and through their usage, new insights can be gained on this issue. Discovering a high-level structure in a set of observations within a low-dimensional manifold is a common strategy used when applying machine learning techniques to tackle several problems while finding a projection from and onto the underlying data distribution. This so-called latent manifold can be used in many applications such as clustering, data visualization, sampling, density estimation, and unsupervised learning. Moreover, data of city patterns has some particularities, such as having superimposed or natural patterns that correspond to those of the depicted locations. In this research, multiple manifolds are explored and derived from city pattern images. A set of quantitative and qualitative tests are proposed to examine the quality of these manifolds. In addition, to demonstrate these tests, a novel specialized dataset of city patterns of multiple locations is created, with the dataset capturing a set of recognizable superimposed patterns.

Keywords: city patterns; dimensionality reduction; urban planning; deep learning

Citation: Agoub, A.; Kada, M. Exploring the Latent Manifold of City Patterns. *ISPRS Int. J. Geo-Inf.* **2021**, *10*, 683. https://doi.org/10.3390/ijgi10100683

Academic Editor: Wolfgang Kainz

Received: 30 July 2021
Accepted: 28 September 2021
Published: 11 October 2021

1. Introduction

Urban patterns within cities are complex and can have heterogeneous structures, and because of this, they are very challenging to simulate. Many factors directly or indirectly affect how a city is shaped and how it develops, and therefore, it can be very challenging to mimic or capture these factors correctly. These factors can be, for example, historical, political, economic, geographical, or a mixture [1]. However, finding a minimum number of parameters that explain the properties of an observed city might help to better tackle this issue and aid researchers in understanding such complex structures. For example, Whitehand et al. [2] argue that one can comprehend complex phenomena such as urban patterns by creating a picture of them using a minimum number of elements (parameters).

These parameters can be viewed as the intrinsic dimensions of an unknown process that generates a city. Scott and Storper [3] demonstrate a large spectrum of opinions discussing the previous statement. These opinions range from arguing that the attempts to define the main characteristics of cities is an impossible proposition since cities are too complex and large to be modeled in such methods. For example, Saunders has a view of cities as merely spatial vessels for many socio-economic activities [4]. In her book, "The Death and Life of Great American Cities", Jacobs [1] advocates for planning cities that have buildings with mixed primary uses and various age groups, small blocks, and high density. Jacobs initiated a discussion in the urban planning community by arguing to solve many of the cities' problems by assisting desirable interactions between agents and highlighting the flaws of a purely top-down approach in planning. These dynamics match the ones of complex adaptive systems, and by definition, the prediction of the outcome of such a system with high certainty is not possible. In other words, to successfully plan cities, the interactions and feedback loops that exist within such a system need to be considered [5].

Pumain [6] views several concepts that consider a city as a complex system, which self-organizes and coevolves in a larger system of cities, and theorizes over the factors that affect and shape cities on a global scale, and demonstrates the nonlinearity of the problem and the existence of autocorrelation in large city developments. Pumain [6] also argues that socio-spatial interactions are complex. Some cities are planned and constructed by several agents' architectures and developers to follow specific superimposed (artificial) patterns; for example, grids such as New York City [7], which facilitate vehicle transportation, or Garden Cities [8], which increase connectivity and create multiple downtown centers while having the benefit of a countryside feel. On the other hand, spontaneous cities (or natural cities) develop during a long period to adapt to the needs and interactions between agents (humans) and the surrounding environment [9]. Many cities are built with strategic advantages near hills, to leverage commercial connectivity or, as multiple numbers of major cities around the globe, to have a direct connection to harbors, are next to rivers, or as resting points on the post track. At the same time, other cities were designed with decoupled functions such as separated locations for shopping, business, and living. Additionally, to add to the complexity of the problem at hand, definitions of clusters or groups that depend on the semantics of phenomena are usually based on a subjective perception of features and the labels used to categorize them. Clustering, by definition, is simply the grouping of similar objects together in separate classes and Ackerman et al. [10] discuss a set of axioms that help to identify the quality of a clustering algorithm, whereas others argue that the problem of clustering itself is inherently subjective [11]. Kleinberg [11] argues that it is impossible to satisfy all of the following criteria simultaneously—scale-invariance, richness, and consistency—in what is called the impossibility theorem of clustering, which makes identifying city clusters inherently subjective.

Considering the previous opinions and limitations, simulating, generating, or clustering natural-looking city patterns can nevertheless be a useful tool for decision support and urban planning, and the mentioned challenges are fortunately not unique to the field of architecture and urban planning. Many solutions are proven to be very effective in understanding massive amounts of heterogeneous data, such as pattern recognition [12], data generation [13], and visualization [14], and in particular, Convolutional Neural Networks (CNNs) [15] are very effective in processing topologically structured data such as images. To leverage the use of these networks, many researchers represent their data in continuous field format (raster), where each of the raster's pixel locations is considered a random variable. Therefore, image data can be viewed as high-dimensional variables. A common technique used in machine learning is to project such high-dimensional data to a low-dimensional manifold with a compact representation. Since the information content does not change when discrete data are represented as a continuous field (vector-to-raster conversion), there must be an efficient encoding of the high-dimensional representation onto a low-dimensional representation. This is in accordance with information theory [16] since the number of bits contained in the data does not change when changing its representation.

In this paper, we explore this possible manifold for encoding city patterns in raster form with the contribution and research structured in the following way: the State of the Art section gives a theoretical background of the used algorithms and is coupled with the Related Work section that mentions similar work and research that use similar data or encoding methods. A novel dataset is then introduced in the Data Origin and Preparation section. This dataset includes multiple city patterns and is designed to contain a known primary semantic variation. This variation is subjectively observed by choosing cities that have a variety of superimposed patterns. The main methodology of the research examines this assumption by using a sequence of novel tests and metrics to determine if the encoding function is able to encode these clusters at separate locations at the manifold and to explore the quality of such encoding. The approach is introduced in the Methodology section. As a part of examining these tests, the dataset is encoded on low-dimension spaces by applying a number of methods that are suitable for linear subspaces and nonlinear manifold mappings. The original data labels are tracked on the low-dimensional manifold to measure

performance metrics and are introduced in the Experiments and Evaluation section. The results are then discussed along with the limitation in the Discussion, Limitation, and Conclusion sections.

2. State of the Art

There is a large number of dimensionality-reduction techniques, and they can be divided according to the objective function into convex and non-convex techniques. Convex techniques are those in which the objective function has no local minima as oppose to non-convex techniques where such minima exist. Full spectral approaches, such as principal component analysis (PCA) [17] and sparse spectral functions, such as Locally Linear Embeddings (LLE) and Laplacian Eigenmaps [18], are examples of convex techniques. Examples of non-convex approaches are Sammon Mapping and autoencoders. Van der Maaten et al. [18] give a complete description and comparative review of these techniques. Dimensionality-reduction techniques are frequently used to understand complex problems that involve high dimensional data, including phenomena that are within an urban context. This section gives a theoretical background of techniques that are utilized within the scope of this research.

Let $X \in \mathbb{R}^D$ be the city data matrix with the dimension (n, D), where n is the number of observations and D is the number of the dimensions or the number of random variables assumed to explain an observation. In this case, D represents the number of pixels in an image or the width of the image multiplied by its height. These variables are often correlated due to the fact that many buildings have a repetitive shape and thus appear on images with similar structures and numbers of pixels. The number of pixels in a high-resolution image is usually much higher compared to its equivalent boundary representation, which requires only a few point coordinates at the corners to encode footprint location and shape. Consequently, it is beneficial to find the smallest set of uncorrelated variables that describe the observations while preserving the maximum amount of information contained in X. The count of variables in this set will be referred to as d, the number of dimensions in the subspace. Or in other words, the objective is to find a mapping U between $X \in \mathbb{R}^D$ to $Y \in \mathbb{R}^d$.

2.1. Linear PCA

PCA is a very common technique that helps find a linear mapping between correlated variables in space D and a minimum set of uncorrelated variables in space d. The following subsection is based on the derivation explained in [19], where this reference can be viewed for a complete rigorous proof of this method. Let $Y \in \mathbb{R}^d$, where $d << D$ according to a transformation U, where:

$$Y = UX \tag{1}$$

The objective is to maximize the variance. If the data matrix X is standardized, i.e., has a mean of zero and a standard deviation of one, the variance–covariance matrix can be expressed as the result of the cross product between the data matrix and its transpose, as demonstrated in Formula (2).

$$max_{variance}\ XX^T \tag{2}$$

In practice, these values can be found by performing single-value decomposition SVD, resulting in the matrices U, Σ, and W:

$$X = U\Sigma W^T \tag{3}$$

Additionally, since these matrices are orthonormal and diagonal, they represent rotation stretching functions:

$$X^TX = U\hat{\Sigma}^2W^T \tag{4}$$

By truncating the transformation, or in other words, keeping a small number of principal components λ_i, one can perform dimensionality reduction. Additionally, since

the first components contain the largest amount of variance, only a small fraction of these components is included. The ratio of the remaining information can be expressed as follows:

$$\%_{Info} = \frac{\sum_{i=1}^{D} \lambda_i}{\sum_{i=1}^{N} \lambda_i} \quad (5)$$

From a geometric point of view, eigenvectors of a transformation matrix represent a set of vectors that preserve the orientation when a transformation is applied. When the variance–covariance matrix is viewed as a transformation, the eigenvectors represent the directions of the maximum variance or the direction where the data are stretched the most according to this specific transformation. This means that finding these vectors and projecting data onto the eigenvector space can highlight the main variance in the data. Visualizing data using Principal Component Analysis (PCA) leverages this observation to convert the data or observation to a set of linearly uncorrelated variables, where the main variance of high dimensional data can be explored using only a fraction of the original dimensions.

2.2. Kernel PCA

Full spectral techniques in general, and PCA in particular, are very practical for many scientific problems. However, linear PCA has some limitations when dealing with high dimensional data, and many solutions were developed to counter these problems. For example, the dimensions of the covariance matrix have the same dimensions as the input data D, and in the case where $D >> n$, the algorithm becomes inefficient [20]. Dual PCA addresses this with the following substitution to make the analysis invariant to the number of data samples n. The covariance matrix size is proportional to the data points' dimension. A second issue with linear PCA is that the analysis considers the similarities between data points without explicitly factoring in the distribution of neighboring points. Even with the success of linear methods, many data manifolds are complex and highly nonlinear. Mapping the data to a high dimensional space before applying PCA can help to make the data more separable with linear hyperplanes in the high dimensional space.

Kernels represent a measure of similarity between data points, and different kernels exist for different problems, with the condition that kernels need to be symmetric and positive semi-definite. Due to the large number of possible kernels to use for each problem, it becomes challenging to decide which kernel to use. A study that demonstrates the effect of using each type of kernel on multiple datasets is shown in the work of Alam and Fukumizu [21]. For example, the sigmoid kernel is useful to compare similarities between categorical data, and Logistic PCA assumes a Binary or Bernoulli distribution of input data. Therefore, it is suitable for binary data and is used to conduct the analysis presented in this paper.

2.3. Graph-Based

Kernel PCA is not the only way to perform nonlinear mapping of data. Many methods utilize the assumed shape of the hidden manifold. Graph-based methods usually have an objective to minimize the distances between points that are located in close vicinity to each other while maximizing the distance to all others. For example, Isomap [22] uses similar principles to multidimensional scaling with the main difference of utilizing geodesic distance as a metric instead of Euclidian distance. For finding the geodesic distance, however, it is challenging to compute this distance practically. Therefore, Isomap calculates local Euclidian distances on a neighbor graph that preserves the curvature of the manifold and approximates the geodesic distance.

Isomap theoretically maps the manifold to a low dimensional space with a small number of distortions. However, practically this method suffers from multiple issues. Due to its time complexity, the algorithm is impractical when a large number of points is analyzed. Discontinuities or holes in the manifold, topological instabilities, and erroneous connections might cause the algorithm to fail or produce undesirable results. Locally

Linear Embedding (LLE) [23] addresses these issues by similarly constructing a K-Nearest Neighbor graph. For each local point, LLE only considers the weights of surrounding points. The objective function of LLE uses these weights to reconstruct the local point in the original space with a minimum error. Similarly, the spectral clustering algorithm builds an undirected graph between points based on a metric of similarity and performs clustering based on graph cuts with the minimum cases calculated based on a Laplacian Eigenmap. Due to their success and to the fact that the city pattern manifold is multidimensional and highly nonlinear, LLE is used in this experiment to visualize and explore the data.

2.4. Stochastic Approaches

The Stochastic Neighborhood Embedding method (SNE) proposes a mapping with an objective function that maintains a similar distribution of points in both original and projected spaces. This approach assumes that the probability p_{ij} of each point i choosing a certain neighbor j in both original and projected space q_{ij} [24]. The probability is based on a dissimilarity metric d and assumes a normal Gaussian distribution of points that are centered around each point at the projected space with the value y as follows:

$$p_{ij} = \frac{exp\left(-d_{ij}^2\right)}{\sum_{k \neq j} exp\left(-d_{ik}^2\right)} \tag{6}$$

$$q_{ij} = \frac{exp(\|y_i - y_j\|^2)}{\sum_{k \neq j} exp(-\|y_i - y_k\|^2)} \tag{7}$$

The distribution in the original spaces can be calculated using the dissimilarity measure "d" while the distribution in the projected space can be calculated using the objective of reducing Kullback–Leibler divergences KL between the original P_i and projected spaces Q_i.

$$C = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}} = \sum_i KL(P_i \;||\; Q_i) \tag{8}$$

This method has the advantage of penalizing the cost of nearby points being mapped on locations that are further apart, with a lower cost for more distant points in the original space being mapped to a nearby location in the projected space. This helps to minimize the risk of collapsing the manifold on itself when it has the characteristics of nearby folds. The SNE method is successful in visualizing many datasets. However, the KL-divergence loss is difficult to optimize, and data viewed in this method sometimes suffer from what is called the "crowding problem". T-distributed stochastic neighbor embedding (T-SNE), a variation of the SNE algorithm, addresses this problem by utilizing Student's or T-Distribution to minimize the effect of the issue [14].

3. Related Work

The techniques described above can be used to aid in urban planning and decision-making in numerous complex problems involving high dimensional data. The following researches demonstrate how dimensionality-reduction techniques can be applied to aid in solving problems that include spatial patterns. For example, Lu et al. [25] try to understand a driver's behavior. The daily activity of a driver can include many events such as stops, pickups, and deliveries, which means that an activity profile of a driver is of a high dimensional nature. Lu et al. [25] include the behavior profile of 243 drivers during 1099 days in Singapore and uses PCA and autoencoders to successfully reduce the dimensionality of the data to a fraction of the original data while keeping most of the original variance [25]. They show that the use of logistic PCA outperforms PCA in modeling events such as stops and drives. In this study, a variational autoencoder (VAE) is additionally implemented to generate novel driver profiles, demonstrating that data-reduction techniques can capture and manipulate a complex phenomenon in a spatial context such as the activity of a driver within a city. Jiang et al. [26] utilize data mining techniques and specifically PCA to examine

the activities of more than 30,000 individuals living in the Chicago metropolitan area and try to model their daily activities and how these activities vary over time. The advantage of using dimensionality-reduction techniques, in this case, is to identify a small number of variables compared to the original data dimension and, by performing this process automatically, avoid superimposing any biases or predefined social demographic classification on the data. Jiang et al. [26] cluster the research individuals into seven groups during the weekdays and eight groups during the weekends. These observation groups are selected by taking advantage of the fact that the variance of the coded data is maximized in these directions. Jiang et al. [26] propose new activity categorization based on the eigenvectors to span the activity space and cluster the projected data using the k-means clustering algorithm [27]. Shaw and Jebara [28] analyze a massive amount of spatio-temporal data of 250,000 daily taxi trips in New York City and model traffic flow within the 2000 busiest city blocks as a high-dimensional data vector. Shaw [28] uses Minimum Volume Embedding (MVE) [29] to project data on a space with a lower dimension and apply spectral clustering [30]. This approach can help visualize the similarities between locations and help understand the traffic flow within the city and derive meaningful conclusions from such data. This is due to the fact that similar instances or data points will be located in proximity to each other when data-reduction techniques are applied. Another example is explored by He et al. [31], who utilize Gaussian Mixture Models and dimensionality-reduction techniques in detecting anomalies in urban temporal networks.

Many researchers have also used data-reduction techniques to map the similarities between city patterns using dimensionality-reduction techniques. The following studies follow a similar approach to the one presented in this research. Therefore, they are described in detail in the following paragraphs to highlight the main differences in our contribution. Moosavi [32] argues that the availability of large datasets such as the one from OpenStreetMaps opens the possibility of using novel methods in analyzing urban patterns on a global scale. Furthermore, since these methods can be used to examine similarities between urban forms, they take into account many factors such as orientation, structural density, and partial deformations. Moosavi [32] additionally explores leveraging unsupervised machine learning techniques with the objective of creating a search engine for urban patterns and generating a research dataset by rasterizing road networks that surround city centers that are similar in scale for all samples regardless of the city's size. He implements a convolutional autoencoder to encode road network data and select the Euclidian distance at the projected space as a metric to measure the similarity between the samples and the k-nearest neighbors (k-NN) algorithm [33] to search and select the closest points in the projected space of over one million images of cities and villages all over the world. Kempinska and Murcio [34] follow similar footsteps and search for a quantitative method to describe urban patterns using generative deep learning techniques, namely, Variational Autoencoders (VAE) [35]. They encode 12,479 samples of binary images that represent road networks around the world using the VAE to project the data onto a latent space and use the T-SNE algorithm to reduce the data dimensions to only two, and apply the k-means clustering algorithm [27] to detect unique clusters. They also project the samples back to the geographical location and identify the extent of similar clusters around the world. However, the approach does not succeed in producing a high-quality interpolation between the road networks and generating novel samples.

Dong et al. [36] suggest examining and analyzing the similarity between urban fabrics using convolutional neural networks, namely, convolutional autoencoders (CAE). Dong et al. [36] choose to use images of residential neighborhoods in Nanjing, China, which contain new and old developed areas that are varied according to the urban morphology. They sample their data according to two criteria: administrative boundaries and scale. They argue that a data sample can be generated for each "plot" due to the fact that the administrative boundaries at the "plot" encourage different designs at the plot level. Therefore, they generate images of each of these plots, and since these plots have large discrepancies in the area they cover, the resulting images will thus have different scales.

Hence, they subsample the dataset according to a scale-based criterion where the patterns are visible. Then, they use a uniform geographical extent, and they center the images based on the plot's boundaries. They use the CAE to extract compressed feature vectors (CFV) that are a dense representation of the information that is available in the original data—however, with a lower dimension. In addition, they add statistical information to the CFV, such as the plot size, edge length, ground space index, and floor space index. They use these features in addition to the compressed vectors to measure the Euclidean distance at the encoded space and to cluster and explore the geometry and topology of these urban fabrics using hierarchical clustering to perform the analysis.

Law and Neira [37] leverage the fact that encoding data using the PCA technique produces latent vectors that are uncorrelated and ordered according to the variance contribution or explained variance. They use convolutional PCA to encode street networks on (Google Street View) images of the city of London and ortho plans of multiple cities streets with a unified geographical extent of 1.5 km $\times$ 1.5 km grid and rasterize them into 256 $\times$ 256 images from locations around the world with the data origin being from OpenStreetMap. They also demonstrate the use of main principal components that capture the street structures and argue that the embedded data at the latent space can be interpreted. The main components are explored by generating novel data using Convolutional PCA to infer the value of the new images by changing the values of a specific component while fixing all others. They also argue that the use of a convolutional PCA matches the accuracy of an autoencoder while maintaining independent and interpretable components. They integrate these components by using multiple visualizations of the samples located at the extreme and statistically interesting values of the principal components, such as the minimum, maximum, and median. Additionally, they demonstrate two-dimensional plots of the principal components and argue that they capture the density, global structure, and local structure of the road network in an independent way.

We set our research apart from previous related work with the following contributions: we propose and follow a distinct method in generating urban pattern datasets, namely, density-based clustering, and detect building clusters automatically and not based on administrative boundaries. Using this method, a novel dataset for all cities of Germany is generated and derived from the OpenStreetMap dataset, and then multiple data-reduction techniques such as linear, nonlinear, and graph-based projections are explored. This research proposes multiple tests to explore the manifold, which can be implemented as tools to aid researchers or urban planners. Our method is distinct from previous studies in the fact that it follows the nonlinearity of the latent space to select similar samples, and, to the best of our knowledge, this approach has not been implemented on building footprint patterns of urban fabrics. This research further suggests a few processes to query these patterns similar to a search engine and tests these queries on well-known city patterns. Additionally, it demonstrates that interpolating between two data samples on a path that is close to the manifold can create novel data or morph between these samples. In addition, the performance of these techniques is evaluated using a set of numerical metrics that indicates the quality and correctness of the predictions.

4. Data Origin and Preparation

It is common in machine learning to search for a dataset that represents the canonical state concerning the studied variables [38]. This approach is followed due to the fact that the major variation in the data is, in many cases, caused by hidden variables that are not of interest, as in the face-detection problem, for example. A successful algorithm aims to detect the face or the variation of expressions, and all other variations that are not of interest, such as the difference in lighting and face location, are considered to be problematic. A powerful algorithm would be able to detect the variation of interest nonetheless. However, at the stage of prototyping or research, it might be beneficial to focus on the analysis of variables of interest and aim to reduce data to a canonical form in order to help avoid the detection of secondary hidden variables as the main axes of variation.

For example, Zivkovic and Verbeek [39] try to capture the movement of a person through a series of images and suggests using a translation and rotation invariant PCA approach on binary data by assuming a Bernoulli distribution and adding hidden translation parameters that are used to rotate and translate the images automatically. Zivkovic and Verbeek [39] explain that when translation and rotation are present in an image, they can be detected as the main dominant deformations when applying data-reduction techniques. To focus back on the problem of city patterns, randomly selected orthoimages of cities usually contain translations and rotation of the objects of interest, such as buildings and road networks. Due to the strong discrepancy of pixel values, such effects will represent a major variation. To avoid this effect and focus our analysis on the semantic differences within city patterns, the data in this research are moved to a canonical form by using data samples that are represented as images with a center that matches that of the captured building cluster.

4.1. Study Locations and Data Source

Data are collected from OpenStreetMap [40]. Multiple study locations are selected, namely, the complete country of Germany and three locations where the superimposed pattern is prevalent: Paris in France, Barcelona in Spain, and New York City (N.Y.C.) in the United States of America. Building clusters in Paris have clear centers as squares with radial streets that pass through this center. Building clusters in Barcelona follow a block pattern that is almost identical in the complete area of interest (AOI), whereas the N.Y.C. AOI is built as a superimposed grid with identical street dimensions. Data samples and preprocessing code can be found under the following Digital Object Identifier (doi:10.5281/zenodo.5145665), with a collection of some of the statistics about the data presented in Table 1.

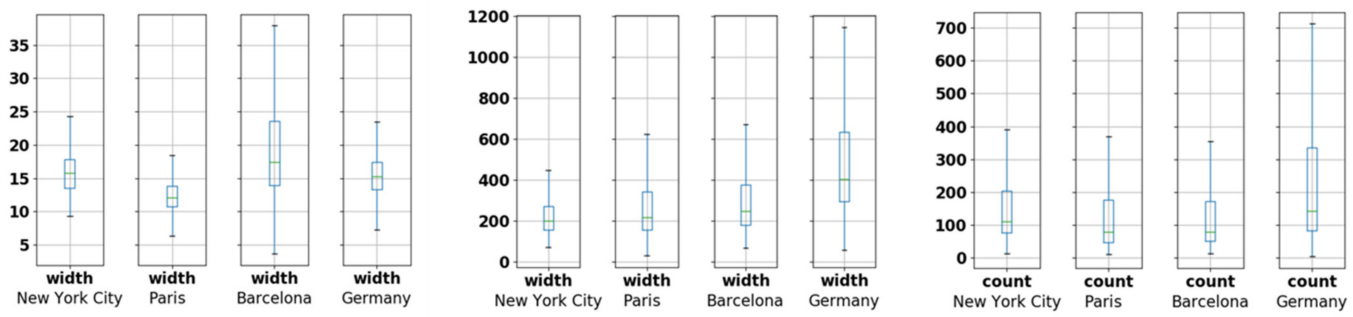
Table 1. Experiment data statistics.

Cluster	**# Used Building Footprints**		**# Images**
Germany	30,823,846		49,069
N.Y.C.	2,231,164		2379
Paris	3,269,579		8555
Barcelona	2,821,748		5226
Training Autoencoders			
Data division	Training: **70%**	Validation: **20%**	Testing: **10%**
# images used in cluster and path visualization			
Germany: **85**	N.Y.C.: **85**	Paris: **85**	Barcelona: **85**

Due to the unified data generation process that is followed in this research, the four classes of patterns in this experiment have similar statistical characteristics. However, the variation of the data depends highly on the scale. For example, if data are captured on a small scale, then many superimposed patterns will not be captured on such a resolution, and the variation will correspond to the general shape of the city. On the other hand, if a large scale is used, then the main variation of the data will be affected greatly by the translation and rotation of individual building footprints. This happens since building footprints occupy a large portion of the image in this case.

Figure 1 shows the count of buildings in each data sample and the maximum width of each cluster belonging to the same range. The complete dataset of Germany has a wider range compared to all other samples because of the large sample count, because the outliers in count and width are very large compared to the rest of the dataset and, if included in the images, will cause the remainders of the samples to not be captured correctly. Therefore, outliers are not displayed, and it is noticeable that for this dataset, most of the data samples contain less than 400 building footprints, and the data sample's geographical span is less than 800 m, while most building footprints' bounding box width is less than 24 m. With these statistics in mind, the image resolution of 64 pixels is chosen to enable the rasterization algorithm to capture individual building footprints when the

distance between them is more than 16 m, which corresponds to a distance less than the width of one building for most building footprints in our dataset. The resolution, however, is adaptive, and changes from sample to sample according to the sample's extent. The data are divided into training, validation, and testing in 70, 20, and 10% ratios, respectively, which is a standard machine learning approach. The results are presented using the testing dataset throughout the experiment.



(**a**) (**b**) (**c**)

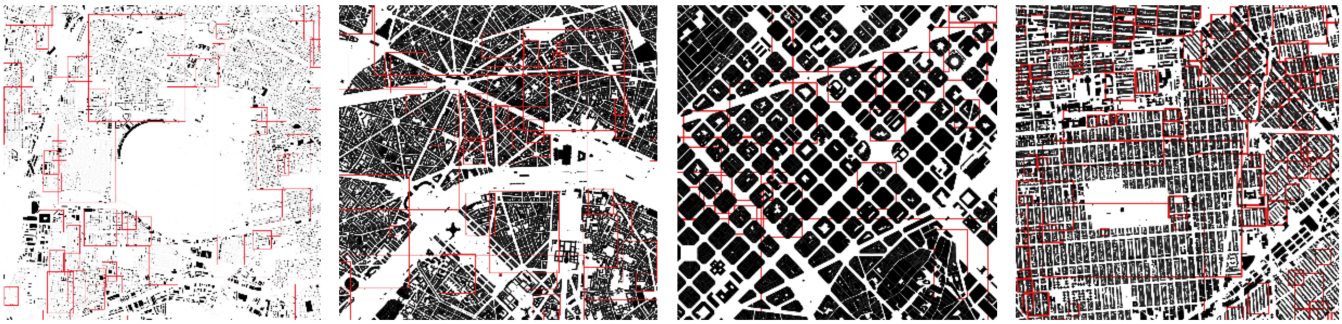
Figure 1. (**a**) A boxplot of average building footprint width in each data sample for each of the experiment classes in the units of meters. (**b**) A box plot of the width of data samples for each of the experiment classes in meters. (**c**) A boxplot of the building count in each data sample grouped by experiment classes.

4.2. Image Sampling Method

It is natural to think about cities, in a city pattern clustering problem, as individual instances of multiple classes. However, there are a few challenges that are related to this approach. The number of cities around the world is comparably small to the number of images that are required to train models that perform classification tasks when applying many state-of-the-art machine learning techniques.

Moreover, building patterns within a city's boundary do not necessarily follow a specific superimposed pattern in all the cities, and many cities have multiple superimposed patterns due to, among others, historic or political reasons. Additionally, a city centroid is not necessarily surrounded by building footprints; many cities are distributed as a crescent, for example, around a geographical or natural feature such as mountains or rough terrains. To counter the mentioned problems, a density-based clustering algorithm is used to detect building clusters in our dataset, namely, Density-Based Spatial Clustering of Applications with Noise (DBSCAN) [41].

The algorithm yields a large number of clusters on a variety of geographical scales, as shown in Figure 2. The DBSCAN algorithm detects clusters of buildings regardless of administrative boundaries and, if the cluster is continuous over a large geographical extent, then this is considered to be a unique variation in the data sample and should be preserved in the image.



(**a**) (**b**) (**c**) (**d**)

Figure 2. Image datasets of building patterns. (**a**) An example of Germany's dataset in Berlin, (**b**) Paris's building pattern, (**c**) Barcelona's building pattern in the area of interest, and (**d**) the superimposed grid pattern in N.Y.C. in the area of Brooklyn. The red squares represent the clusters' extents around building footprints.

4.3. Preparing Data Images

City pattern data are converted to images. First, polygons representing building footprint features are converted to binary masks, where pixel values that correspond to geographical locations where buildings exits are given the value of "one", and all other pixel values are given the value "zero". Formulas (9)–(15) are used to calculate the corresponding coordinates on the local binary masks in the following way:

$$P_{image} = P * T \tag{9}$$

$$P = [x,\ y,\ 1] \tag{10}$$

$$s_x = \frac{img_x}{(max_x - min_x)} \tag{11}$$

$$s_y = \frac{img_y}{(max_y - min_y)} \tag{12}$$

$$d_x = s_x * min_x \tag{13}$$

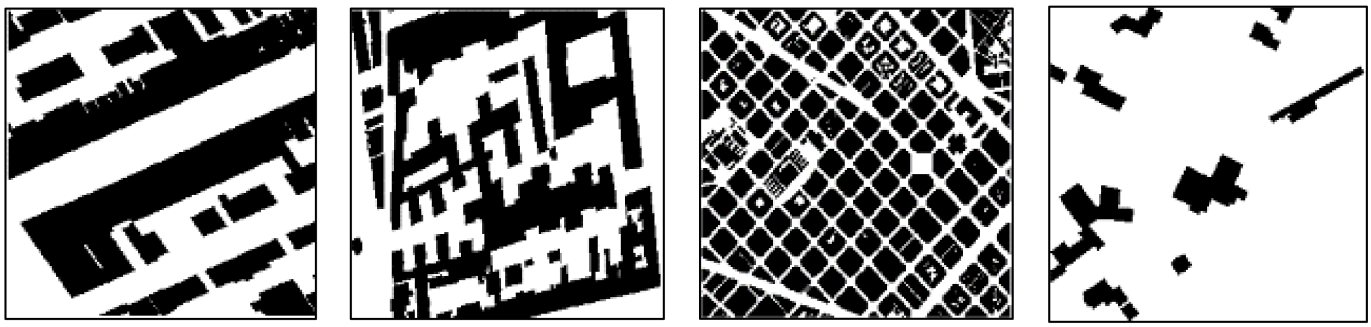
$$d_y = s_y * min_y \tag{14}$$

$$T = \begin{vmatrix} s_x & 0 & -d_x \\ 0 & s_y & -d_y \\ 0 & 0 & 1 \end{vmatrix} \tag{15}$$

The ratio of the images is consistent regardless of the shape of the cluster, as a square is calculated around the centroid of the cluster. The coordinates min_x, min_y, max_x, and max_y are computed as the maximum extent with respect to x and y, and half of the extent is added or subtracted from the coordinates of the centroid accordingly. Building footprints are projected to the image space using a translation matrix T, for which the scaling parameters s_x and s_y are calculated using the geographical extent, and the image pixel count img_x and img_y for the axes' direction of x and y separately. Figure 3 displays some examples of this process on the used dataset for the used building pattern classes.

Each of the locational pixels is a variable in the dimensionality reduction analysis, and it is a common step to subtract the mean before performing the analysis. Figure 4 shows the average of each group. The values that are the highest are located at the sample center, with lower values at the edges. This is expected since the samples were created using the DBSCAN algorithm with the condition that each of the samples needs to be centered around a building footprint, and for each of these centers, a minimum of the number of other building instances within a threshold distance must exist. Due to the small number of samples in the first three groups, a deviation from a centered average can be noticed. We argue that this is caused by the particular data selection for each specific pattern and the

small sample count in each of these groups. The German dataset, on the other hand, shows that this method of data production produces a high average and non-zero-pixel count at the center with a decreasing value toward the edges.



(**a**) (**b**) (**c**) (**d**)

Figure 3. Random samples from the four groups (**a**–**d**) depict clusters from datasets of N.Y.C., Paris, Barcelona, and Germany, respectively. Black pixels represent the location of footprints, and white pixels represent background.

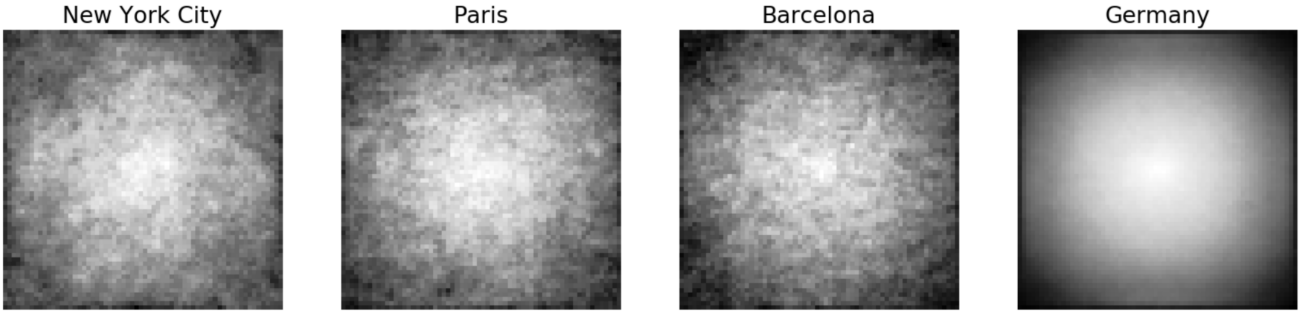


Figure 4. The average image pixels over the four classes used in the experiment for each locational pixel. Brighter pixels indicate higher values.

5. Methodology

The datasets are encoded into latent manifold Y using mapping U following Formula (13). U is represented as linear (PCA), nonlinear mappings Kernel PCA, T-SNE, or an autoencoder neural network. Several quantitative and qualitative tests are conducted to evaluate the quality and correctness of the resulting clusters. In addition to the amount of information that is extracted using reconstruction techniques such as PCA and autoencoders, the quality of the resulting manifold is examined using the following tests. See Figure 5.

$$U(X) \rightarrow Y \tag{16}$$

A visual indicator of the projection method quality is examined by visualizing the projection weights of the PCA and autoencoders. These weights are reshaped into a two-dimensional array and examined for any superimposed patterns that match that of some of the data samples. The amount of information encoded using PCA is examined by plotting the variation captured on each of the principal components in a Cattell–Nelson–Gorsuch scree test (or CNG scree test) [42]. At the same time, the information captured by the autoencoder is examined by performing a reconstruction of the samples and observing the main characteristics of the reconstructed sample compared to the original one.

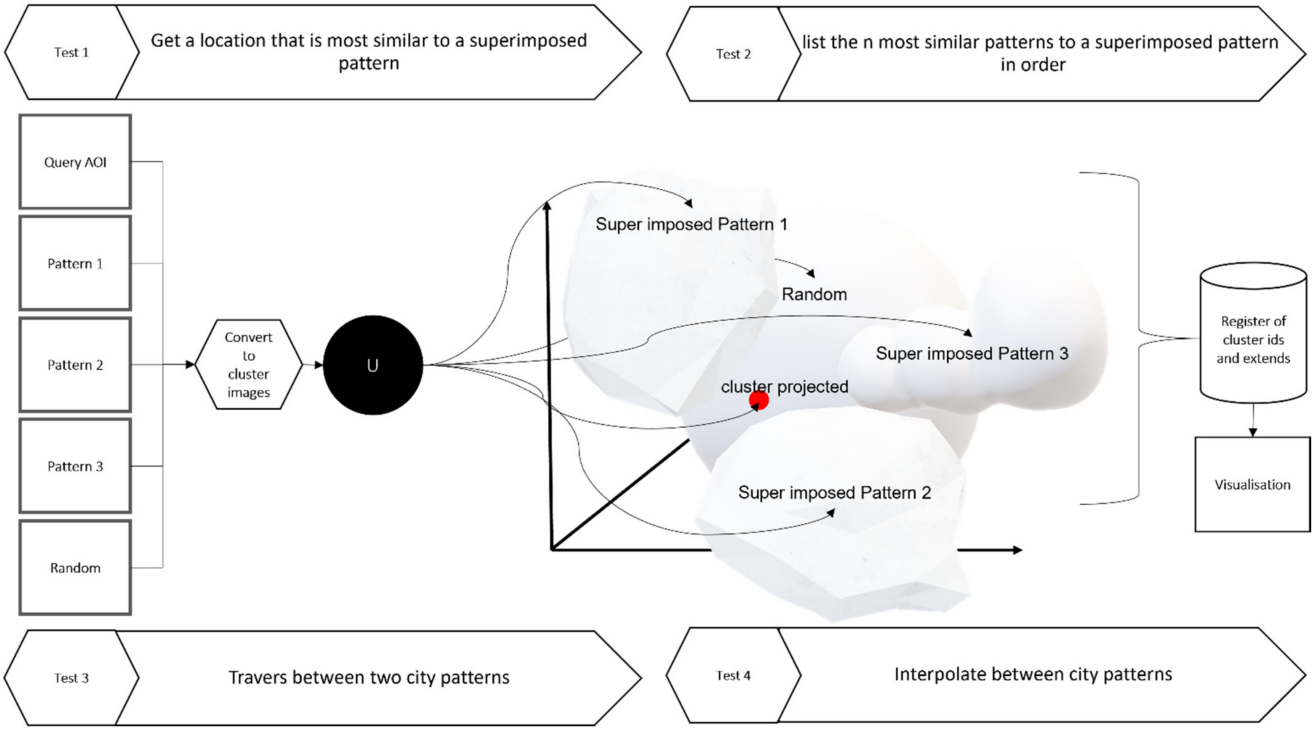


Figure 5. Mapping to a hidden manifold. The operation sequences demonstrate multiple operations that can be used to better understand and analyze the build-up patterns. U represents a function or mapping from a high to a low-dimensional manifold.

To visually examine the quality of the projection, it is expected that similar samples are located near each other in the latent manifold, with input data X and a projection method U that projects data to a low dimensional manifold as $Y = U(X)$. The algorithm assumes that the point y that represents the pattern the most is located at the geometric centroid of the pattern at the latent space. It is unlikely that any points will be located exactly at the coordinates of point C. Therefore, point y is computed by selecting a point with the minimum Euclidian distance from the pattern centroid's coordinate. Point y is identified and paired with a registry/spatial database entry that contains the geographical extent of building clusters as follows:

- Compute pattern centroid C for a subset Y_I;
- From Y_I , compute the nearest point y to C;
- From Y, compute the nearest point $y_{nearest}$ to y;
- From X, get the matching $x_{nearest}$ sample and the cluster's geographical extent.

This test can be extended to find the n most similar patterns to a given pattern I in order of similarity. For example, when one would like to inquire about other cities or parts of cities around the world that have similar designs or structures to a given example. The algorithm continues by building a K-Nearest-Neighbor Search tree for efficiency and then querying this tree for points that are closest according to the Euclidean distance as a metric of closeness. Test 2 is defined as:

- Build a K-D tree $\mathbb{T}$ to perform KNN search;
- Calculate pattern centroid C for the subset Y_I;
- Set point C: query $\mathbb{T}$ for KNN search for n neighbors $\rightarrow P_{nearest}$.

Furthermore, the quality of the manifold is examined by traversing points between two city patterns Y_I and Y_L. However, tracing a direct vector connecting two points on the manifold can yield points/samples that are not located on the manifold [22]. This is especially true if the manifold is strongly curved or nonlinear, and this is a characteristic that is assumed here for city patterns manifolds. To have a smooth transition between patterns, the algorithm must be able to sample or select points in a transition that follows

this curvature. The algorithm is inspired by the logic used by Isomap [22] and utilizes the approximation of the geodesic distance as our metric and builds the shortest path between the centroids of the patterns that one wants to traverse. The algorithm starts by building a K-Nearest Neighbors points graph between the mapped points in the encoded data. Then, it calculates the shortest distance path between these points using Dijkstra's algorithm [43]. The number of points on this path depends on the distance between the centroids and the density of the mapped points, and the density of projected sample points at these locations. Test 3 is defined as:

- Build a K-Nearest-Neighbour graph G;
- Calculate pattern centroid C_I for the subset Y_I;
- Calculate pattern centroid C_L for the subset Y_L;
- Calculate the shortest path $\mathbb{P}$ from C_I To C_L;
- For index i in $\mathbb{P}$:;
  - Get cluster p_i Image;
- End.

A further test is added to examine the manifold further outside of the locations where data samples are encoded. In many cases, the number of samples in the dataset is not sufficient to cover the manifold without any interruptions or holes. The sampling methods when collecting the data could have some biases or simply do not cover all possible cases. To counter this issue, for some projection functions, it is possible to create a novel data sample using U^{-1}, an inverse of the transformation U. When the parameters of projection U are calculated in a way that can capture the main characteristics of the sample X, if the projection U has a mathematical inverse. Two points x_i and x_j from the original dataset are selected with the corresponding projection y_i and y_j, respectively. Additionally, according to the number of desired subsamples n, calculate the value of step s $= \frac{y_j - y_i}{n}$. According to this value, calculate the value of the novel sample x_l. Test 4 is defined as:

- Project x_i according to $U(x_i) \rightarrow y_i$;
- Project x_j according to $U(x_j) \rightarrow y_j$;
- Compute direction vector d as $d = y_j - y_i$;
- Compute step s as s $= \frac{y_j - y_i}{n}$;
- Compute intermediate points at the projected space for each step;

$$l \text{ as } y_l = y_i + l * s$$

- Project intermediate points back to space $\mathbb{R}^D$ as:

$$x_l = U^{-1}(y_l)$$

Data sampling in the experiment assumes several unique classes based on subjective observations of superimposed patterns of some cities of interest. This assumption is evaluated by automatically dividing the projected data Y to a set of clusters with the help of K-means clustering [27] using the number of suggested clusters as a hyper-parameter (for example: {2, 3, 4, 5, 6, 7, 8, 9, 10}). The resulting cluster quality is evaluated using the Fowlkes–Mallows [44], average Silhouette Coefficient [45], Adjusted Rand Index [46], and Adjusted Mutual Info [47] depending on this hyper-parameter. The Silhouette Coefficient measure will correspond to a high value if the cluster count is correct, meaning there is a high similarity between an element and its cluster (cohesion) and dissimilarity between an element and other clusters (separation). In other words, the clustering results when the hyperparameter "number of clusters" of the K-means clustering algorithm corresponds to the correct value, in which case the Silhouette Coefficient will probably yield a higher silhouette score compared to the resulting clusters with a far-off value.

Fowlkes–Mallows Index, Adjusted Rand Index, and Adjusted Mutual Info determine the similarity between two sets of clusters, where one of them is being considered as ground truth. Both Adjusted Mutual Info and Adjusted Rand index takes into account the fact that some samples will be assigned the correct class by random chance and normalize the score accordingly. Therefore, the values of these metrics will normally be lower than

the Rand Index or Mutual Info Score. These tests yield numeric values that can be used to objectively compare the performance of different encoding algorithms. Higher scores are interpreted as good performance of these algorithms.

Using K-means clustering on Y will only identify unique clusters that might not match the labels of the ground truth. To perform this matching, the centroid of each of the resulting clusters is computed, then matched with the closest ground truth cluster centroids. All members of this cluster are given the corresponding label. The true positives, false positives, and false negatives are computed based on this matching. Samples that are correctly identified are considered true positives, samples that are falsely labeled as a member of the sample are considered false positives, and samples that are not matched to a different cluster are considered false negatives.

6. Experiments and Evaluation

To enable the reproducibility of the experiment, the main parameters of the experiment are listed in Table 2. The experiment is implemented using the Python 3.7 programing language [48]. The main library that is used to implement the autoencoder deep learning models in Keras 2.4.0 [46]. The choice of the model parameters is supported by default by the library. The models are implemented with the help of the scikit-learn 0.24 library [49]. Only the main parameters of the models are highlighted in the table, while the remainder are chosen to be kept as the default by the scikit-learn library. A repository of the experiment, the parameters, and example data can be found by resolving the following DOI:10.5281/zenodo.5145665. The repository additionally contains the weights of a trained version of Autoenconder_6 and Autoencoder_200 and a Python Notebook that performs the tests 1–4.

Table 2. Parameter values that are used with the encodings and tests.

Method	Parameters			
PCA	# of components: **6**	Kernel: **None**		
Kernel PCA	# of components: **6**	Kernel: **Sigmoid**		
LLE	# of components: **2**	# of neighbors: **5**	Regularization constant: $\mathbf{10^{-3}}$	Max iteration: **100**
Isomap	# of components: **2**	# of neighbors: **5**	Metric: **Minkowski Distance**	P (Minkowski Distance): **2**
MDS	# of components: **2**	Relative tolerance: $\mathbf{10^{-3}}$	Dissimilarity: **Euclidean**	
t-SNE	# of components: **2** Min grad norm: $\mathbf{10^{-7}}$	Perplexity: **30** Metric: **Euclidean distance**	Early exaggeration: **12** Angle: **0.5**	Learning rate: **200** Early exaggeration: **12**
Autoencoder_6	Input shape: **(4096.1)** Output activation: **Sigmoid**	Bottleneck layer's shape: **(6.1)** Loss function: Binary Cross-Entropy	Activation: **Relu** Optimizer: **Adam**	Output layer's shape: **(4096.1)**
Autoencoder_200	Input layer's shape: **(4096.1)** Output activation: **Sigmoid**	Bottleneck layer's shape: **(200.1)** Loss function: Binary **Cross-Entropy**	Activation: **Relu** Optimizer: **Adam**	Output layer's shape: **(4096.1)**

6.1. Data Encoding Using PCA vs. an Autoencoder

Only a small percentage of variation is captured using techniques such as PCA in the first components on this dataset, as seen in Figure 6. It is difficult to estimate the minimum number of components that are required in ordered to capture the variation between building patterns classes. Usually, a Cattell–Nelson–Gorsuch scree test or CNG scree test [42] is performed with the objective of selecting the number of parameters. This number specifies a cutting point where adding more principle components will not have a significant contribution to the overall variation. However, in this experiment, the first component has the highest contribution to the variation, which only represents 10% of the variation, and the contribution drops to only 2% for the second component and less for all of the others. This means that each of the samples is adding a small amount of variation, and the technique fails to capture the most amount of variation by only using a small subset of dimensions. The experiment, however, focuses on designing an exploratory technique that captures the difference between building patterns and does not focus on the reconstruction of detailed city images. Therefore, the desired variation for exploration purposes may exist in the first components, as the rest of this experiment confirms.

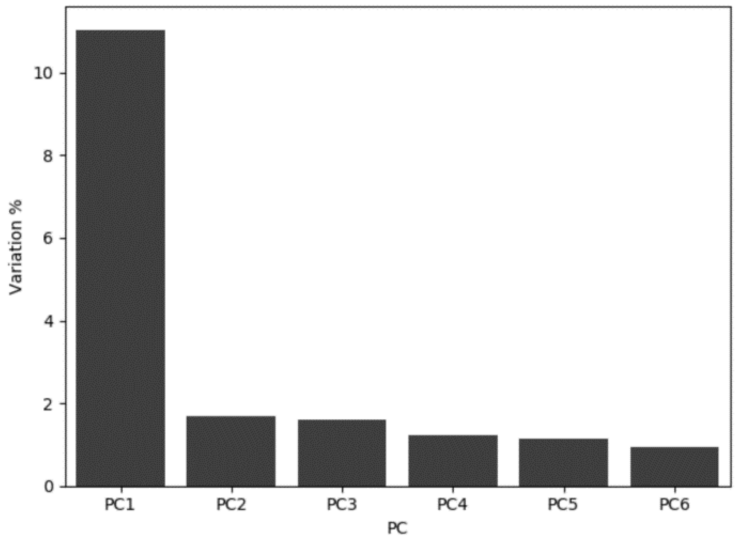


Figure 6. The amount of variation captured by the first six components using a PCA decomposition.

Figure 7 shows that when running PCA on the experiment dataset, the first components capture some features of the dataset, such as a block in the center of the image as in PC3, or a repetitive pattern such as in PC2, PC4, PC5, and PC6, while PC1, which captures 10% of the data is not easily interruptible. These components are multiplied in a dot product with the data to produce the projected data. Therefore, these values might indicate a combination of the main characteristics of the depicted images by PCA. For example, multiplying PC3 with the image might have a very low activation for Barcelona patterns at the center, whereas PC6 might have a strong activation for patterns that have a building density at the center of the image.

Similarly, an autoencoder network is trained on the dataset, and the encoder weights are displayed in Figure 8. These weights are taken from the encoder branch. The autoencoder has a simple structure of an input and output layer that matches the number of pixels in the city pattern image and a bottleneck of a fully connected layer that is composed of six neurons. This choice is based on the experiment conducted using PCA analysis to determine the minimum number of components needed to encode the data efficiently. The connections of the input layer carry the encoder weights. These weights are reshaped into a 2D image, as shown in Figure 8. The autoencoder uses the compensation of these weights to encode building pattern images into a tensor of six values, which is a very low number of dimensions compared to the original dimensions of the images, which are 4096.

A binary cross-entropy loss function is used to guide the optimization during training the autoencoder, and this means that a low loss corresponds to the number of correct pixels reconstructed by the decoder using only the six values that are passed through the bottleneck. The results shown in the figure are predictions based on the test dataset.

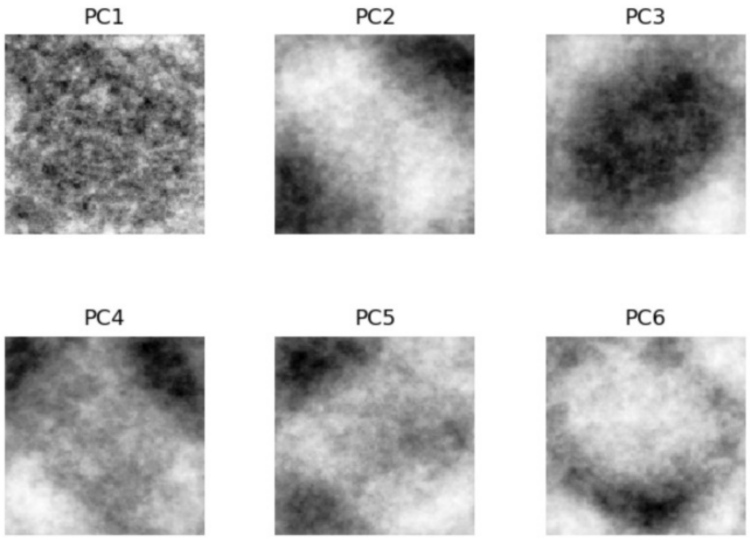


Figure 7. PCA components. This figure shows each of the first six principal components from the PCA analysis. This analysis is trained in the experiment dataset with a reconstruction loss function. Brighter pixels indicate higher numeric values in contrast to dimmer values that indicate low numeric values at that location.

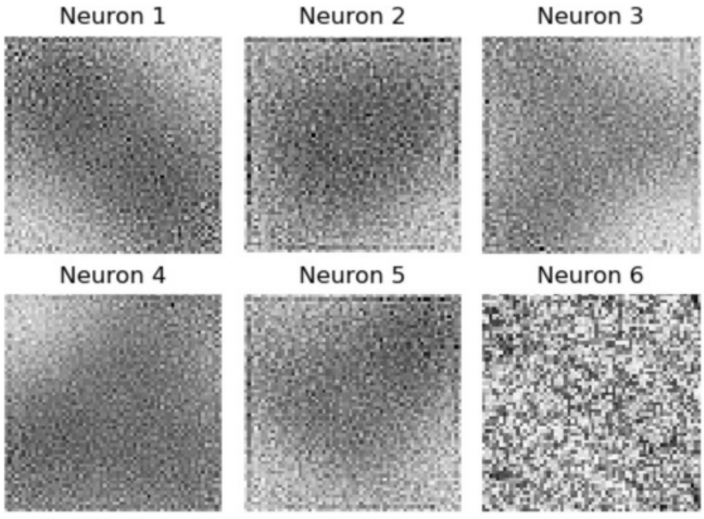
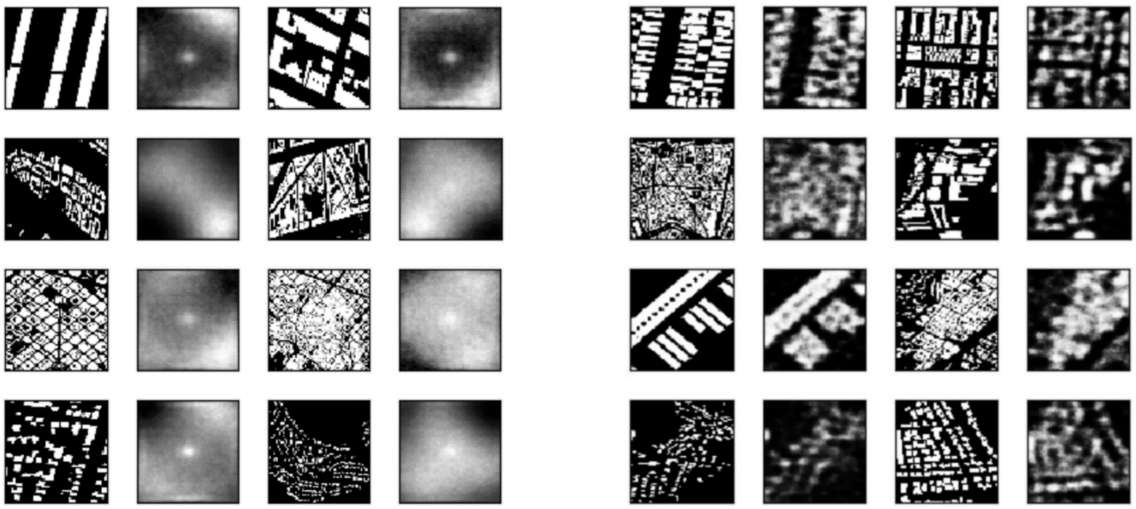


Figure 8. The weights of trained Autoencoder_6 neurons. Bright values indicate high numeric values, while dim values indicate a low numeric value.

The extreme compression of information into only six dimensions forces the weights to correspond to the main visual characteristics of the images. This conclusion is based on the reconstruction results displayed in Figure 9, where the constructed images depict a rough estimate of the main direction of the original pattern and the distribution of its density. Subfigure b in Figure 9 demonstrates that the usage of a large number of neurons enables Autoencoder_200, which has 200 neurons at the bottleneck layer, to reconstruct higher-quality images with a higher number of details retained compared to using only six neurons.

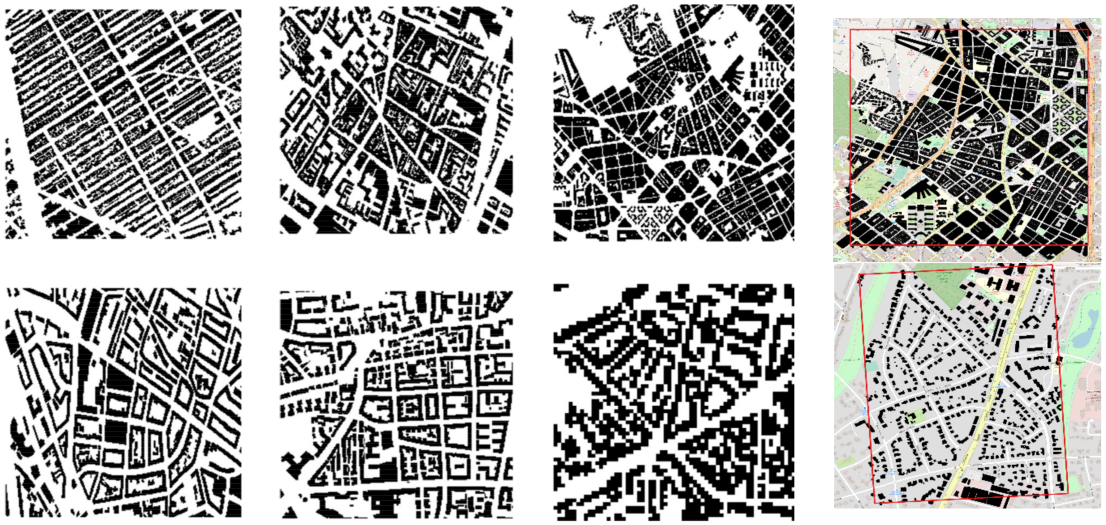


(**a**) (**b**)

Figure 9. City images and their corresponding reconstructions. (**a**,**b**) An alternation of columns from the left-hand side to the right-hand side show original and reconstructed images. Each row is selected randomly from a pattern class, and they are from top to bottom New York City, Paris, Barcelona, and Germany, respectively. The figure images are reconstructed using Autoencoder_6 with six neurons at the bottleneck section, whereas figure b shows images that are reconstructed using Autoencoder_200 with 200 neurons in the bottleneck section.

6.2. Visual Tests

Test 1 is executed to answer the following query: select from the patterns of Germany the most similar pattern to the following target clusters: New York City, Paris, and Barcelona. Both images of the New York City centroid and its paired image from the Germany dataset according to Autoencoder_200 are depicted in Figure 10. This paired image shows the same orientation of patterns and large streets that are in New York City. Both images of the Paris pattern and its paired image from the Germany dataset show a similar block sectioning with an empty square at the lower right side of the image, whereas both Barcelona centroid and its paired images contain a large square on the right-hand side with similarly curved streets.



(**a**) (**b**) (**c**) (**d**)

Figure 10. Results of Test 1. The top row shows the centroid of each of the clusters, which are (**a**) New York City, (**b**) Paris, and (**c**) Barcelona. The top row depicts the image that is most similar to the centroid. The last column (**d**) is a repetition of the Barcelona cluster; however, it is shown in the geographical context.

Figure 11 shows the results of Test 2 execution. It depicts a query image and the five most similar cities according to the Euclidean distance in the space that is encoded using Autoencoder_200. It is noticeable that the selected patterns have the same orientation of streets in the query image; the density of the building footprints and street curvature has a similar visual distribution to that of the query image.

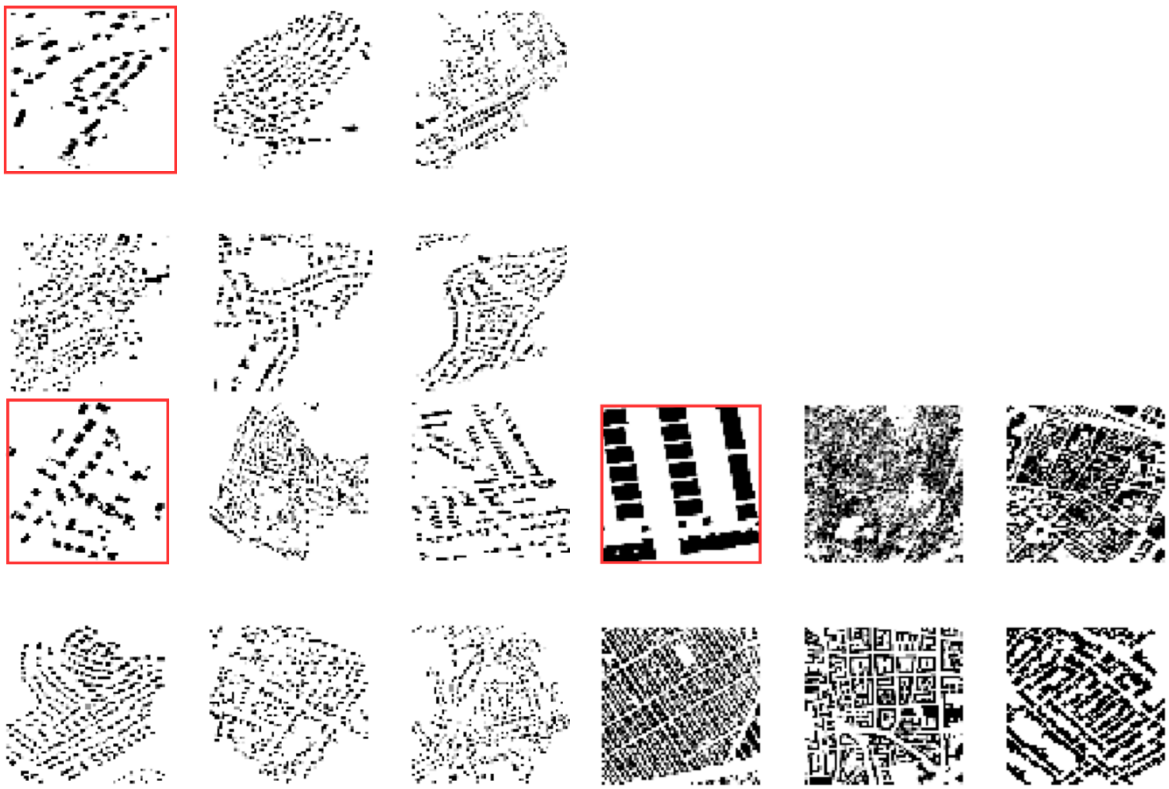


Figure 11. Results of Test 2. The data sample surrounded with a red square is the query image. The remaining 5 images are images that have the shortest Euclidean distance at the manifold from the image encoded by Autoencoder_200.

Test 3 is performed to traverse points between each of the four centroids on the shortest path connecting these centroid points, and results are shown in Figure 12. The desired results are samples whose characteristics change gradually from the start sample depicted in Figure 12 at the top left to the end sample of the centroid of the US dataset depicted on the bottom right. However, the algorithm is limited to the data samples that are given as input. Notably, for both examples, there is a gradual increase in the density with the selected images moving toward the US centroid. The superimposed pattern direction of the traversed sample alternates suddenly at some steps, as shown in subfigures (b,d). It is worth mentioning that Figure 12 shows the shortest path between samples in only three dimensions. However, the cost distances are calculated over 200 dimensions, and due to this fact, it is noticeable that the complex and not the straight-forward path is shown in yellow.

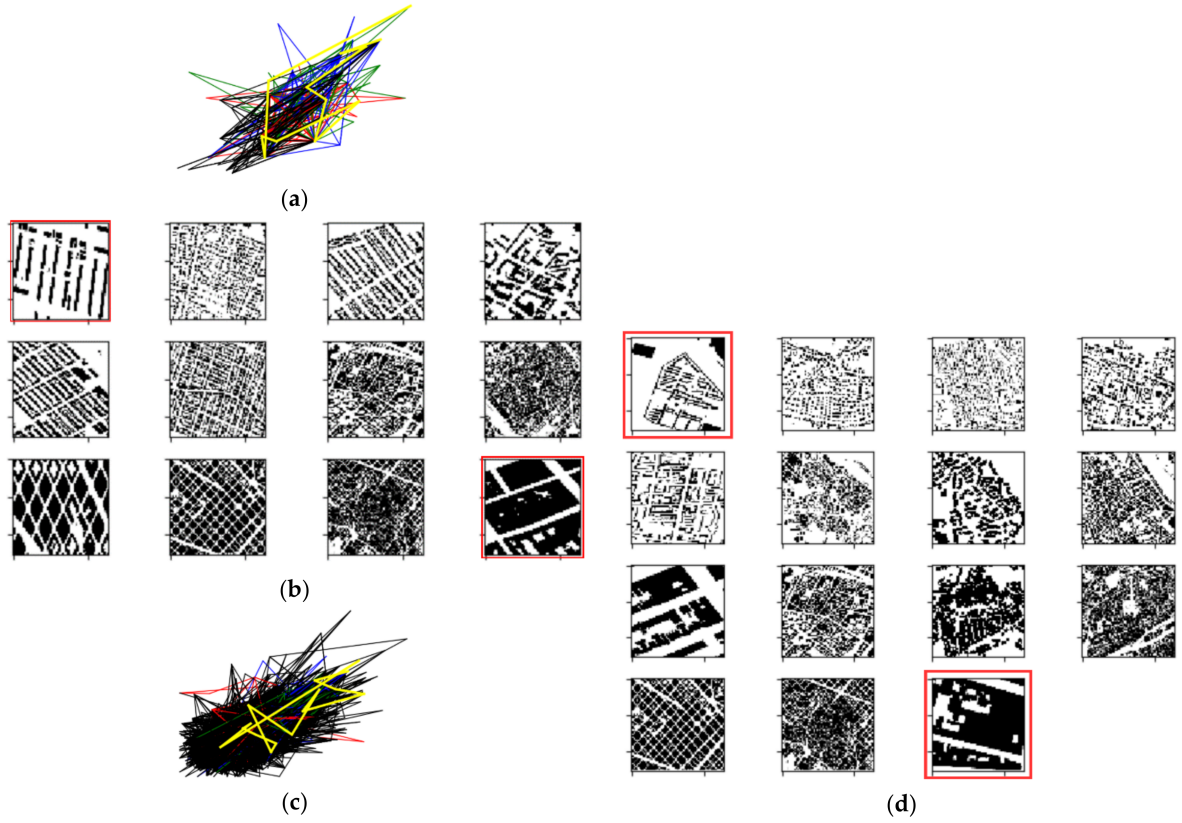


Figure 12. Results of Test 3. Traversing the graph between two data samples. The starting and ending data samples are marked in red in both subfigures (**b**,**d**). The data samples show the data points that match the ones on the graph traversal shown in (**a**,**b**). The subfigures (**a**,**c**) show the shortest path depicted in yellow in a three-dimensional scene.

Arguably, the number of samples for each class in the experiment is not sufficient to cover the completed and nonlinear manifold without any gaps. To test this assumption, an interpolation between samples on the graph was performed, and the parameters of a high dimensional vector v were calculated with the step toward the next sample on the graph, as explained in Test 4. The results are shown in Figure 13 using the decoder branch U^{-1} as an approximation of the inverse transformation of the encoder branch U. Note that the orientation and density of each location on the image change gradually between subsamples. These images are out of the sample and predicted by the decoder. The results are based on the values of the 200-dimensional vector that are incremented according to a small threshold starting from the first and ending at the second image. It indicates that the autoencoder was able to encode some of the main characteristics of city patterns, such as orientation and density.

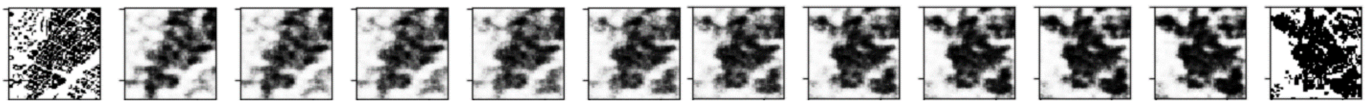


Figure 13. Results of Test 4. Interpolation between two data samples using decoder_200. The most left and right samples belong to the original dataset X. The rest of the images are out of the sample calculated based on the decoder prediction with an increment left to right, top to bottom.

6.3. Clustering Performance and Model Evaluation

Following the steps of testing presented in the Methodology section, the data samples are projected on a low dimension space using the following methods: PCA, Kernel PCA, LLE, Isomap, MDS, t-SNE, and Autoencoder_6 with six neurons and Autoencoder_200 with 200 neurons at the bottleneck. The experiments and metrics were collected using these methods, and tests were performed on a subset of the data (85 samples per class). These results are shown in Figure 14 and Table 3.

The visualization in subfigure (a) in Figure 14 keeps track of the original classes for each projected image of the building pattern classes and is shown as ground truth data with the label GT. Additionally, the prediction of these methods is indicated with the label P. The predicted data are clustered using the K-means clustering algorithm with a hyperparameter of cluster count set to four. The predicted clusters are compared visually with the ground truth, and it is observed that Linear PCA, referred to as PCA and Kernel PCA, projected the clusters and of Barcelona and Germany to be near the ground-truth clusters, while Paris and N.Y.C. were located in proximity to each other. Other methods such as Autoencoder_6 located a large density of the data samples on one of the axes of variation, while samples encoded using LLE followed a linear pattern with the sample cluster of ground truth and predictions following the same sequence. MDS and t-SNE produced what looks like a uniform distribution of samples over the manifold. At the same time, both Isomap and Autoencoder_200 predicted all clusters with locations that resample the ground-truth data with Autoencoder_200, giving nearly identical locations for the corresponding centroids. The visualization in subfigure (b) in Figure 14 depicts a plot of the Silhouette Coefficient for each of the methods. It is noticeable that the value of cluster count of "two" produced a high Silhouette Coefficient average score for PCA, Kernel PCA, LLE, MDS, Autoenocder_6, and Autoencoder_200. Moreover, a high spike of value was noted at the value of 8 for the PCA, Kernel PCA, t-SNE, and Autoenconder_200 methods. On the other hand, a maximum value was observed when setting this number to three for the Isomap method, compared to a local maximum setting of four for the methods of PCA, Kernel PCA, and MDS.

Furthermore, the experiment continued to explore setting the value of cluster count to four and examine the assumption enforced by the data design. Results are shown in Table 3. The LLE method had the highest score of the Silhouette Coefficient for the N.Y.C. cluster of 0.58, while the Isomap method performed the best for the clusters of Paris, Germany, Barcelona, and the average Silhouette Coefficient Score, with values of 0.49, 0.69, 0.5, and 0.57, respectively. On the other hand, Kernel PCA had the heights score for the metrics of Adjusted Rand Index, Adjusted Mutual Info, homogeneity, completeness, and Fowlkes–Mallows Index score, with values of 0.33, 0.39, 0.37, 0.42, and 0.52, respectively. Other methods had overall high scores for both the cluster quality and prediction correctness compared to the ground truth, such as linear PCA and LLE. Autoencoder_200 had low-value cluster quality compared to high correctness values. T-SNE had consistently lower performance when compared to other methods regarding the quality of the resulting clusters and the correctness of the resulting labels.

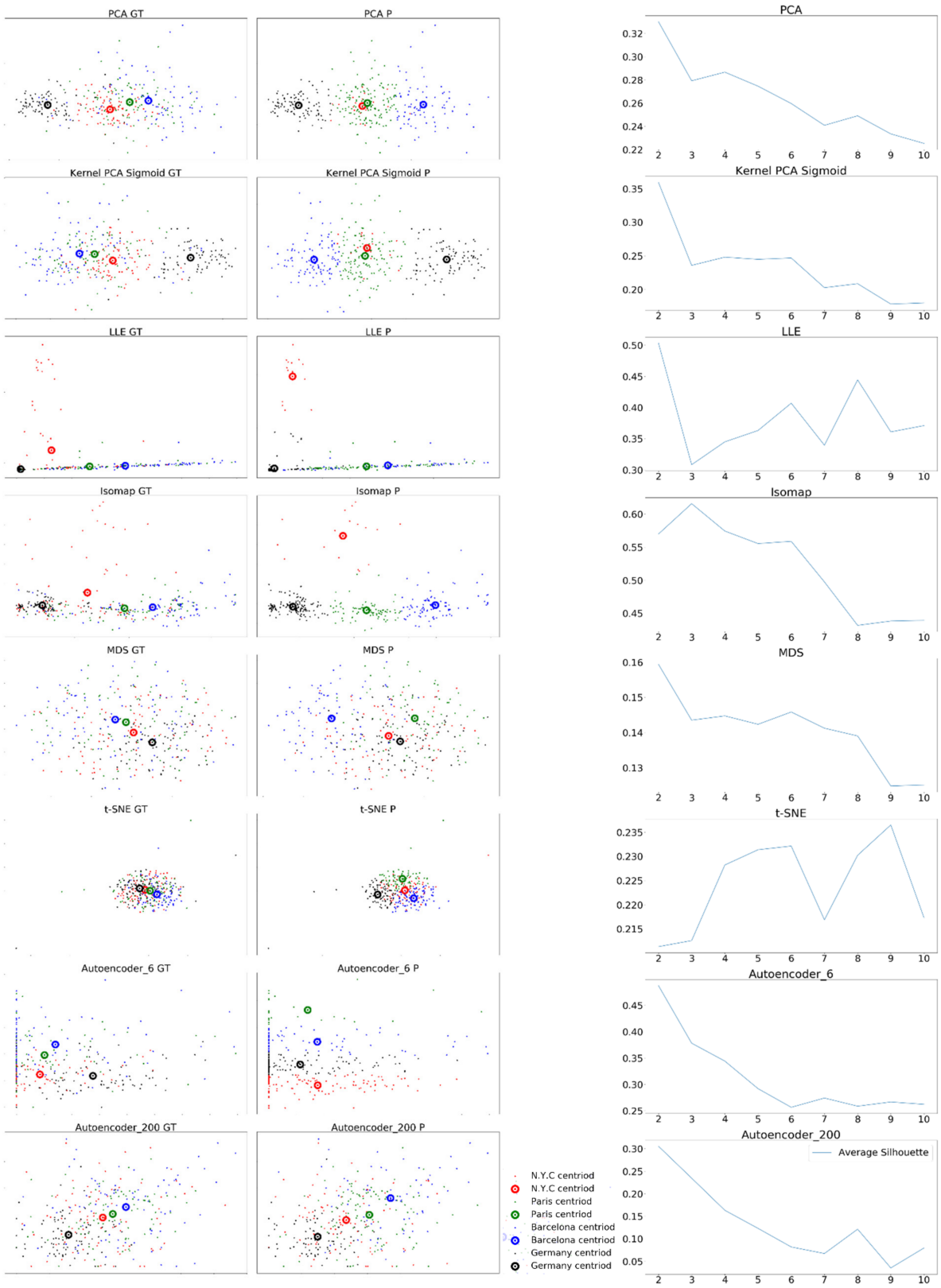


Figure 14. Clustering performance. In (**a**), each colored dot represents a data sample projected using the corresponding method. Ground-truth data are labeled with GT and prediction P. Each unique color indicates distinct clusters. Subfigure (**b**) demonstrates plots of Silhouette Coefficient value for each method when changing the number of clusters, which is depicted on the x-axis.

Table 3. Resulting metrics of used methods when fixing the cluster count value to four.

Score	PCA	Kernel PCA Sigmoid	LLE	Isomap	MDS	t-SNE	Autoencoder_6	Autoencoder_200
N.Y.C. Silhouette Average	0.49	0.35	**0.58**	0.42	0.09	0.22	0.37	0.06
Paris Silhouette Average	0.19	0.15	0.12	**0.49**	0.08	0.22	0.37	0.07
Germany Silhouette Average	0.53	0.46	0.52	**0.69**	0.27	0.21	0.36	0.43
Barcelona Silhouette Average	0.16	0.18	0.37	**0.50**	0.09	0.26	0.27	0.10
Average Silhouette Coefficient	0.29	0.25	0.35	**0.57**	0.15	0.23	0.34	0.16
Adjusted Rand Score	0.31	**0.33**	0.20	0.20	0.23	0.07	0.10	0.28
Adjusted Mutual Info	0.36	**0.39**	0.28	0.24	0.22	0.08	0.14	0.31
Homogeneity	0.34	**0.37**	0.26	0.24	0.23	0.09	0.15	0.32
Completeness	0.40	**0.42**	0.33	0.27	0.23	0.09	0.16	0.32
Fowlkes–Mallows Index	0.51	**0.52**	0.45	0.43	0.42	0.30	0.35	0.46

7. Discussion, Limitations, and Conclusions

7.1. Conclusions

In this paper, a few algorithms are presented to explore city patterns with the help of machine learning techniques. To test these algorithms, a specialized dataset that has a variety of city patterns was created. In addition, the complete dataset of OSM of Germany's over 30,823,846 million building footprints was processed. The paper also demonstrates that using encoding techniques is an effective way of exploring these patterns and presented several example algorithms or tests that can be used in exploring and understanding urban patterns.

Initially, the visual comparison was performed on a trained linear PCA of six components and an equivalent autoencoder of six neurons. The visual demonstration showed that the first six components of PCA captured a large amount of variation and information about the city patterns (see Figure 6). Additionally, superimposed patterns were observed on the visualized weights, including values that corresponded to central or repetitive linear patterns activation in a variety of directions (see Figure 7). This is an indicator that these weights will produce significantly different values for samples that belong to distinct clusters that themselves have certain superimposed patterns. Similar behavior is observed when using Autoeoncder_6 to force data to be encoded and reconstructed using only six neurons (see Figure 9). The weights of Autoencoder_6 reflected noisy images without any detectable superimposed patterns. However, the model was able to reconstruct images that arguably contain the main superimposed visual characteristics of the original images, such as density and orientation. Not surprisingly, increasing the number of neurons at the bottleneck of the autoencoder to 200 produced highly detailed images. We argue that the reason for visually detectable patterns on the weights of a trained PCA is due to the fact that PCA enforces uncorrelated components by design. This means that the variance encoded on one of the component axes should not correlate with the following axis, causing the model to learn to correspond to the main visual characteristics of the image dataset independently.

The experiment focuses on the model Autoencoder_200's ability to perform test sequences 1–4 introduced in the Methodology section. Test 1 returns city patterns that are most similar to a certain cluster centroid. The result of the test is not decisive due to the fact that the similarity between the images is not prominent (see Figure 10). However, the similarity between the samples was noticeable when performing Test 2 that returns an n number of similar samples (see Figure 11). Likewise, traversing between two clusters in Test 3 is expected to yield images that have a visual appearance that transitions between the cluster centroids. However, this transition was not easily detectable (see Figure 12). This is due to the fact that the amount of samples used in the experiment does not cover the complete manifold. Test 4 proves the previous assumption in addition to demonstrating the generalization capabilities of the autoencoder. "Out of sample" images were generated by

interpolating between two points on the manifold. The resulting successful interpolation images form a smooth transition between the data samples concerning the direction and density of the sample and are an indicator of a high-quality manifold (see Figure 13).

The previous tests are subjective due to the fact that they depend on a human annotator to determine the quality of the results. However, they are a strong indicator of the successful encoding of the main visual characteristics of the city patterns. To support this conclusion from these tests further, a number of metrics are computed by clustering the data at each of the encoded manifolds, and results are presented in Figure 14 and Table 3. The metrics can be considered to be representative of two categories: the quality of clusters measured by the Silhouette Coefficient, and the correctness of the labeling measured by the Adjusted Rand Score, Adjusted Mutual Info, homogeneity, completeness, and the Fowlkes–Mallows Index. Isomap had the best performance regarding the quality of the separated clusters, while PCA with a Sigmoid kernel performed the best in matching the clusters to the ground-truth labels. The high scores on these metrics for Autoencoder_200 match the observations that were collected in the previous tests and confirm that a simple encoder with a low number of neurons can perform very well on a complicated task such as the unsupervised clustering of city patterns.

7.2. Limitation and Future Work

Many decisions and thresholds hat to be taken and selected to execute this experiment, including the images' spatial resolution, size, and the way samples were selected. For example, the DBSCAN algorithm is used to select the data sample centers. Choosing a different clustering algorithm can produce a dataset that has dissimilar statistical characteristics to the one used in the experiment. Additionally, multiple thresholds are used in the data-encoding algorithms, such as the number of components and the number of neighbors allowed in the graph (see Test 3: Methodology). Similarly, changing the threshold of the K-mean clustering will yield a different set of clusters for each method. Changing these thresholds might impact the path traversed by the algorithm.

The main hypothesis of this research is that similar patterns are located in the near vicinity of each other in the projection space. However, this assumption has two main caveats. First of all, one has to choose the metric of distance in the projected space. The Euclidian distance is chosen to test the algorithms. However, a different measure of proximity or distance might have a better correlation to the similarity between patterns in that space. The second caveat is that the dataset clusters used in this research were selected based on a subjective observation that New York, Paris, and Barcelona have visually different city patterns. The encoding algorithms used in this research were able to differentiate between these patterns and confirmed this assumption. Future work can include an independent assessment of these similarities, evaluate the results objectively, and give more rigorous proof. In addition, a benchmark database of different city patterns can be created and annotated by professional city developers and researchers in order to have an objective ground truth of the cluster labels.

From a statistical point of view, it is assumed in this research that the observed images are capturing an equilibrium of the same statistical process that governs the development of cities (ergodicity and stationarity). This assumption allows the sampling of different data samples in multiple locations. Without this assumption, one needs to observe a repeated development of each city a large number of times, and unfortunately, such a dataset is not realistic, and therefore each image is considered as an independent observation. These aspects could be further investigated by projecting cities that are created using a variety of processes onto the subspace and examining the results.

From a practical point of view, Test 3 has a high time complexity due to the fact of building a nearest neighbor graph between all projected points. Future research could investigate alternatives to build this graph and navigate the shortest path between points with low time complexity. Additionally, only a fully connected neural network is considered in the tests. However, Convolutional Neural Networks are suitable for topologically

structured data. Convolutional autoencoders might have the advantage of detecting additional features on multiple scales. Further research needs to be conducted to test if this observation is also true for city pattern data.

Projecting city patterns onto a lower-dimensional space can help better understand cities, and tools that are implemented to explore this pattern can be a powerful aid to support urban designers and city planners. This fits the direction of current research where more and more technologies are used in cities with the goal of improving the lives of people around the world in future cities [50–52].

Author Contributions: For this research, the following contributions are made. Conceptualization, Amgad Agoub; Data curation, Amgad Agoub; Formal analysis, Amgad Agoub; Investigation, Amgad Agoub; Methodology, Amgad Agoub; Software, Amgad Agoub; Supervision, Martin Kada; Visualization, Amgad Agoub; Writing—original draft, Amgad Agoub; Writing—review & editing, Martin Kada. All authors have read and agreed to the published version of the manuscript.

Funding: We acknowledge support by the German Research Foundation and the Open Access Publication Fund of TU Berlin.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Experiment code and trained model are available by resolving the following doi:10.5281/zenodo.5145665.

Acknowledgments: Map data copyrighted by OpenStreetMap contributors are available from https://www.openstreetmap.org (accessed on 11 April 2020).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Jacobs, J. *The Life and Death of Great American Cities*; Random House: New York, NY, USA, 1961.
2. Whitehand, J.; Batty, M.; Longley, P. *Fractal Cities: A Geometry of Form and Function*; Academic Press: Cambridge, MA, USA, 1996.
3. Scott, A.J.; Storper, M. The Nature of Cities: The Scope and Limits of Urban Theory. *Int. J. Urban Reg. Res.* **2015**, *39*, 1–15. [CrossRef]
4. Saunders, P. *Social Theory and the Urban Question*; Hutchison: London, UK, 1981.
5. Holland, J.H. Complex Adaptive Systems. *Daedalus* **1992**, *121*, 17–30.
6. Pumain, D. An evolutionary theory of urban systems. In *International and Transnational Perspectives on Urban Systems*; Springer: Berlin/Heidelberg, Germany, 2018; pp. 3–18.
7. Gerard, K. *City on a Grid: How New York Became New York*; Da Capo Press: Boston, MA, USA, 2015.
8. Howard, E. *Garden Cities of Tomorrow*; Faber: London, UK, 1946.
9. Larice, M.; Macdonald, E. *The Urban Design Reader*, 2nd ed.; Taylor and Francis: Hoboken, NJ, USA, 2013; ISBN 9781136205668.
10. Ackerman, M.; Shai, B.-D. Measures of clustering quality: A working set of axioms for clustering. *Adv. Neural Inf. Process. Syst.* **2008**, *21*, 121–128.
11. Kleinberg, J. An Impossibility Theorem for Clustering. In *Advances in Neural Information Processing Systems*; The MIT Press: Cambridge, MA, USA, 2003; pp. 463–470.
12. Qassim, H.; Verma, A.; Feinzimer, D. Compressed residual-VGG16 CNN model for big data places image recognition. In Proceedings of the IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC), Las Vegas, NV, USA, 8–10 January 2018; IEEE: Piscataway, NJ, USA, 2018.
13. Goodfellow, I.J.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative Adversarial Networks. *arXiv* **2014**, arXiv:1406.2661. [CrossRef]
14. van der Maaten, L.; Hinton, G. Visualizing data using t-SNE. *J. Mach. Learn. Res.* **2008**, *9*, 2579–2605.
15. LeCun, Y.; Haffner, P.; Bottou, L.; Bengio, Y. Object Recognition with Gradient-Based Learning. In *Shape, Contour and Grouping in Computer Vision*; Springer: Berlin/Heidelberg, Germany, 1999; pp. 319–345.
16. Shannon, C.E. A Mathematical Theory of Communication. *Bell Syst. Tech. J.* **1948**, *27*, 623–656. [CrossRef]
17. Pearson, K.L., III. On lines and planes of closest fit to systems of points in space. *Lond. Edinb. Dublin Philos. Mag. J. Sci.* **1901**, *2*, 559–572. [CrossRef]
18. Van Der Maaten, L.; Postma, E.; van den Herik, J. Dimensionality reduction: A Comparative Review. *J. Mach. Learn. Res.* **2009**, *10*, 66–71.
19. Jolliffe, I.T.; Cadima, J. Principal component analysis: A review and recent developments. *Philos. Trans. A Math. Phys. Eng. Sci.* **2016**, *374*, 20150202. [CrossRef]
20. Schölkopf, B.; Smola, A.; Müller, K.-R. Nonlinear Component Analysis as a Kernel Eigenvalue Problem. *Neural Comput.* **1998**, *10*, 1299–1319. [CrossRef]

21. Alam, M.A.; Fukumizu, K. Hyperparameter selection in kernel principal component analysis. *J. Comput. Sci.* **2014**, *10*, 1139–1150. [CrossRef]
22. Tenenbaum, J.B. Mapping a manifold of perceptual observations. *Adv. Neural Inf. Process. Syst.* **1998**, *10*, 682–688.
23. Roweis, S.T.; Saul, L.K. Nonlinear dimensionality reduction by locally linear embedding. *Science* **2000**, *290*, 2323–2326. [CrossRef]
24. Hinton, G.; Roweis, S.T. Stochastic neighbor embedding. *NIPS* **2002**, *15*, 833–840.
25. Lu, F.; Zhao, F.; Cheah, L. Dimensionality Reduction to Reveal Urban Truck Driver Activity Patterns. *Transp. Res. Rec.* **2018**, *2672*, 81–92. [CrossRef]
26. Jiang, S.; Ferreira, J.; González, M.C. Clustering daily patterns of human activities in the city. *Data Min. Knowl. Disc.* **2012**, *25*, 478–510. [CrossRef]
27. Le Cam, L.M.; Neyman, J. *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability: Weather Modification*; University of California Press: Los Angeles, CA, USA, 1967.
28. Shaw, B.; Jebara, T. Dimensionality Reduction, Clustering, and PlaceRank Applied to Spatiotemporal Flow Data. In Proceedings of the Machine Learning Symposium, Montreal, QC, Canada, 14–18 June 2009.
29. Shaw, B.; Jebara, T. Minimum Volume Embedding. *PMLR* **2007**, *2*, 460–467.
30. Ng, A.Y.; Jordan, M.I.; Weiss, Y. On Spectral Clustering: Analysis and an algorithm. In Proceedings of the 14th International Conference on Neural Information Processing Systems: Natural and Synthetic (NIPS'01), Vancouver, BC, Canada, 3–8 December 2001; pp. 849–856.
31. He, M.; Pathak, S.; Muaz, U.; Zhou, J.; Saini, S.; Malinchik, S.; Sobolevsky, S. Pattern and Anomaly Detection in Urban Temporal Networks. *arXiv* **2019**, arXiv:1912.01960.
32. Moosavi, V. Urban morphology meets deep learning: Exploring urban forms in one million cities, town and villages across the planet. *arXiv* **2017**, arXiv:1709.02939.
33. Fix, E.; Hodges, J.L. *Nonparametric Discrimination: Consistency Properties*; Project; Randolph Field, USAF School of Aviation Medicine: Randolph Field, TX, USA, 1951; pp. 21–49.
34. Kempinska, K.; Murcio, R. Modelling urban networks using Variational Autoencoders. *Appl. Netw. Sci.* **2019**, *4*, 114. [CrossRef]
35. Kingma, D.P.; Welling, M. Auto-Encoding Variational Bayes. *arXiv* **2014**, arXiv:1312.6114 2013.
36. Dong, J.; Li, L.; Han, D. New Quantitative Approach for the Morphological Similarity Analysis of Urban Fabrics Based on a Convolutional Autoencoder. *IEEE Access* **2019**, *7*, 138162–138174. [CrossRef]
37. Law, S.; Neira, M. An unsupervised approach to geographical knowledge discovery using street level and street network images. In *Proceedings of the 3rd ACM SIGSPATIAL International Workshop on AI for Geographic Knowledge Discovery, Chicago, IL, USA, 5 November 2019*; Gao, S., Ed.; Association for Computing Machinery: Chicago, IL, USA, 2019; ISBN 9781450369572.
38. Jaderberg, M.; Simonyan, K.; Zisserman, A.; Kavukcuoglu, K. Spatial Transformer Networks. *Adv. Neural Inf. Process. Syst.* **2015**, *28*, 2017–2025.
39. Zivkovic, Z.; Verbeek, J. Transformation invariant component analysis for binary images. In *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2006), New York, NY, USA, 17–22 June 2006*; Fitzgibbon, A., Taylor, C.J., LeCun, Y., Eds.; IEEE Computer Society: Los Alamitos, CA, USA, 2006; ISBN 0769525970.
40. OpenStreetMap. Planet Dump. Available online: https://planet.osm.org (accessed on 11 April 2020).
41. Ester, M.; Kriegel, H.-P.; Sander, J.; Xu, X. A density-based algorithm for discovering clusters in large spatial databases with noise. *Kdd* **1996**, *96*, 226–231.
42. Cattell, R.B.; Gorsuch, R.L.; Nelson, J. Cng scree test: An objective procedure for determining the number of factors. In *Proceedings of the Annual Meeting of the Society for Multivariate Experimental Psychology*; APS: Washington, DC, USA, 1981.
43. Dijkstra, E.W. A note on two problems in connection with graphs. *Numer. Math.* **1959**, *1*, 269–271. [CrossRef]
44. Fowlkes, E.B.; Mallows, C.L. A method for comparing two hierarchical clusterings. *J. Am. Stat. Assoc.* **1983**, *78*, 553–569. [CrossRef]
45. Eisner, J. Silhouette Coefficient. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL), Prague, Czech Republic, 28–30 June 2007*; Association for Computational Linguistics: Stroudsburg, PA, USA, 2007.
46. Steinley, D. Properties of the Hubert-Arable Adjusted Rand Index. *Psychol. Methods* **2004**, *9*, 386. [CrossRef]
47. Vinh, N.X.; Epps, J.; Bailey, J. Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance. *J. Mach. Learn. Res.* **2010**, *11*, 2837–2854.
48. van Rossum, G.; Drake, F.L. *Python 3 Reference Manual*; CreateSpace: Scotts Valley, CA, USA, 2009; ISBN 1441412697.
49. Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; et al. Scikit-learn: Machine Learning in Python. *J. Mach. Learn. Res.* **2011**, *12*, 2825–2830.
50. Shirowzhan, S.; Tan, W.; Sepasgozar, S.M.E. Digital Twin and CyberGIS for Improving Connectivity and Measuring the Impact of Infrastructure Construction Planning in Smart Cities. *Int. J. Geo-Inf.* **2020**, *9*, 240. [CrossRef]
51. Camero, A.; Alba, E. Smart City and information technology: A review. *Cities* **2019**, *93*, 84–94. [CrossRef]
52. Sun, M.; Fan, H. Detecting and Analyzing Urban Centers Based on the Localized Contour Tree Method Using Taxi Trajectory Data: A Case Study of Shanghai. *Int. J. Geo-Inf.* **2021**, *10*, 220. [CrossRef]