Augmentation Techniques for Balancing Spatial Datasets in Machine and Deep Learning Applications

Alberto Belussi
alberto.belussi@univr.it
Department of Computer Science
University of Verona
Verona, Italy

Diego Garofolo
diego.garofolo@studenti.univr.it
Department of Computer Science
University of Verona
Verona, Italy

Sara Migliorini
sara.migliorini@univr.it
Department of Computer Science
University of Verona
Verona, Italy

Abstract

Thanks to the availability of a huge amount of spatial data, many new machine and deep learning (ML/DL) applications have emerged that are able to deal with such kind of information. In particular, new cost models have been developed with the aim of predicting the cost of spatial operations carefully. For obtaining good ML/DL models, the training activity is usually performed with synthetically generated datasets that capture as many spatial distributions as possible and as many combinations of features as desired (e.g., cardinality, geometry complexity, etc), with the aim to improve the generalization capabilities of the trained models. However, when a model is used to estimate some properties of a spatial operation, like the range query selectivity, balancing the characteristics of the input datasets could be not enough to guarantee a balancing in the ground truth values of the target variable. Therefore, we need to develop a way to balance the final results without recomputing the operation from scratch. This paper formalizes the notion of dataset balancing in the context of spatial ML/DL, proposes a set of metrics for evaluating the degree of balancing of the input domains and the target values, and defines a set of augmentation techniques specifically tailored for spatial data. Finally, it tests the effects of such augmentations in the training of a generic ML cost model for estimating the selectivity of spatial range query.

CCS Concepts

• **Information systems** → **Database management system engines**; • **Computing methodologies** → **Machine learning approaches**.

Keywords

Spatial augmentation, machine learning, deep learning, training set balancing

ACM Reference Format:
Alberto Belussi, Diego Garofolo, and Sara Migliorini. 2024. Augmentation Techniques for Balancing Spatial Datasets in Machine and Deep Learning Applications. In *The 32nd ACM International Conference on Advances in Geographic Information Systems (SIGSPATIAL '24), October 29-November 1, 2024, Atlanta, GA, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3678717.3691230

SIGSPATIAL '24, October 29-November 1, 2024, Atlanta, GA, USA

ACM ISBN 979-8-4007-1107-7/24/10
https://doi.org/10.1145/3678717.3691230

1 Introduction

In recent years, the increased availability of spatial data has been accompanied by the development of many tools for efficiently dealing with such kind of information. Spatial big data systems, like SpatialHadoop [9], GeoSpark [23], Simba [21], LocationSpark [14], and Sphinx [7], have been developed with the aim to provide optimized implementations of spatial operations, like range queries, spatial join, and so on. Several different implementations of these operations are usually available, each one fitting well with the different characteristics of the input datasets. Therefore, they need to be accompanied by cost models that can help choose the best one and estimate the cost of the operations in advance. Many of these cost models are based on the application of machine learning (ML) or deep learning (DL) techniques that are able to capture better the intertwined characteristics of both the spatial datasets and the operations [16–18].

A recent development in such field is the attempt to provide generic ML/DL models that are able to deal with any kind of spatial datasets and can be tailored to specific spatial operations with a small effort [4, 5]. The training of such models is performed through synthetically generated datasets that are able to capture a large variety of spatial distributions and characteristics of geometries, like their cardinality, complexity, placement, and so on. For this purpose, specific spatial data generators have been developed, such as SpiderWeb [11], which are able to treat a wide set of relevant spatial parameters [20]. These tools are able to provide a huge training set that can be balanced with respect to the specific characteristics of interest. However, when ML/DL models are used to estimate some properties of spatial operations computed on such datasets, balancing the original dataset is not a sufficient requirement for obtaining balanced results. Let us consider, for instance, the ML model presented in [4] which estimates a set of range query parameters, such as selectivity, number of MBR tests, and execution time. In this case, even if the considered datasets are generated such that there are an equal number of them for each spatial distribution, and the range queries are properly distributed inside the reference space, the resulting parameters could be very skewed somehow. For instance, the range of resulting selectivity values could cover only a small portion of the possible spectrum from 0 to 1, with a very small number of candidates for values greater than 0.1.

Balancing the training set is a very important issue in ML/DL problems since the performances of the obtained models greatly depend on the grade of data balancing. Imbalanced data essentially

means having unequal examples for different things inside a training set. This can seriously affect how well your ML/DL model works, especially when it needs to handle the less common cases. The classical problems related to imbalanced data are biased learning and misleading accuracy. In classification tasks, the second problem has been overcome by considering other evaluation metrics, for instance, F1 score, precision, and recall, in place of accuracy. However, in regression tasks, like the estimation of the range query selectivity or execution time, this cannot be easily addressed in the same way. In these cases, there is a need for specific techniques dealing with the balancing of the input training set.

In the literature, there are essentially three adopted methodologies for dealing with unbalanced training sets: undersampling, oversampling, and augmentation. *Undersampling* is a resampling technique that essentially focuses on balancing the class distribution by reducing the number of instances in the majority class. This is typically achieved by randomly removing instances from the majority class until a more balanced training set is obtained. This technique is suitable if the number of available instances in the minority class is still enough. Conversely, *oversampling* essentially consists of increasing the number of instances in the minority classes. This is typically done by either duplicating existing instances or generating synthetic data points similar to the minority class. This last activity, also known as *augmentation*, represents a better choice with respect to the duplication of instances because it reduces the overfitting risk.

Data augmentation differs from the generation of synthetic data because it is driven by the original data with some minor changes. Many augmentation techniques have been defined in the literature for image datasets, which essentially consist of slightly changing an image to obtain a different one that includes the same information. For instance, they consist of making geometric and color space transformations, such as flipping, resizing, cropping, brightness, and contrast changes. These techniques are not suitable for spatial data in vector format, particularly for spatial operations. Let us consider the spatial range query; in this case, a slight modification on the input dataset or the query window can greatly change the obtained result. Since the recomputation of the operation result can be very costly, augmentation techniques for spatial data should be driven by the desired obtained results, preventing the need to recompute them on the modified input datasets.

The contribution of this paper is manifold: (a) it formalizes the notion of balancing for a spatial input collection from which both a training and test set will be extracted (Sect. 3). (b) It introduces a set of balancing metrics for both the input domains and the ground truth values, which allow one to measure the degree of balancing of an input collection correctly (Sect. 4). (c) It proposes a set of spatial augmentation techniques that allows one to increase the number of instances in the minority classes without re-executing costly operations (Sect. 5). These generic operations have been tailored to the specific problem of estimating spatial range query parameters, like selectivity, number of MBR tests, and execution time. Guided by the desired result, namely the minority class to be augmented, they are able to slightly modify the input datasets to obtain it without recomputing the range query from scratch. Finally, (d) it tests the proposed metrics and augmentation techniques to a set of spatial input collections with different characteristics (Sect. 6).

2 Related Work

Augmentation techniques have been widely studied and applied in the context of image classification. In [12], the authors study the impact of various data augmentation methods in image classification tasks made with Convolutional Neural Networks (CNN). The considered augmentation methods include GAN/WGAN, Flipping, Cropping, Shifting, PCA jittering, Color jittering, Noise, Rotation, and some combinations. Some of them, like PCA and color jittering, are specifically tailored for image processing and cannot be applied to spatial data in vector format. However, some others, like rotation and noise, have inspired some of the augmentation techniques proposed in this paper. In [13], the authors propose a comprehensive survey about image data augmentation techniques, subdividing them into two main classes: basic image manipulations and deep learning approaches. The first class includes geometric transformations, such as rotation and flipping, and photometric transformations. Conversely, the second class uses deep learning techniques, like adversarial training and GAN data augmentation, to produce new training data. Similarly, the work in [15] experimentally demonstrates that geometric augmentation methods outperform photometric methods when training on a coarse-grained dataset, showing that altering the geometry of the images is more important than just lighting and color. If some techniques of the geometric transformations can be considered as a starting point for the ones proposed in this paper, the main difference remains the difficulty in producing augmented data with the desired characteristics, which corresponds to preserving a particular behavior during the application of a spatial operation, like producing the same selectivity in a range query. Conversely, the photometric methods are not applicable here since they are based on the construction of a lower-dimensional representation of a raster image.

As highlighted in [10], the powerfulness of an augmentation technique depends on its ability to preserve the properties of the original data we are interested in. Therefore, augmentation techniques have to be tailored and specialized based on both the input data and the kind of properties under study. This suggests not only that the methods developed for raster images cannot be applied to vector spatial data but also that the applicable techniques can change based on the considered spatial operation and metrics.

In [22], the authors propose a set of GIS-based data augmentation techniques that can automatically generate labeled training map images from shapefiles using GIS operations. Even if these techniques are tailored for geographical data and try to preserve important spatial characteristics, they have been designed for raster images and include techniques like blurring and resizing, which are specific for raster data rather than vector formats.

The generation of synthetic spatial datasets with specific characteristics and distributions has been widely treated in [20] and implemented in tools like [11]. However, starting from a balanced bunch of spatial datasets is not enough to obtain a balanced training set for estimating the costs of spatial operations. For instance, for a cost model, like the one in [18], which estimates the selectivity of the range query, the notion of balancing is defined in terms of the number of query results rather than the properties of the input datasets. Obtaining a balanced set of ground truth values cannot be governed by simply changing the characteristics of the input.

In [5], the authors declare to balance the training set of the ML/DL models for spatial cost estimation by applying some undersampling and oversampling techniques without giving much details about them and without formalizing their definition. To the best of our knowledge, this paper is the first attempt to formalize the problem of balancing collections of spatial vector data by proposing a set of metrics and augmentation techniques that allow the increase of the minority classes without the need to recompute costly spatial operations.

3 Problem definition

As previously explained in Sect. 1, the training of a ML/DL model requires balanced training sets to be effectively performed. This means that the preparation of a training set, even synthetically generated or set up from real data, cannot be done without following some guidelines that lead to a balanced situation with reference to both the input domains and the ground truth values. Balancing the input domain means producing a training set with the proper level of variability and representability of the reality to be modeled, ensuring better generalization capabilities. Conversely, balancing the ground truth is essential to training the model correctly on all the proposed cases. In this paper, we concentrate on the proper generation of a synthetic training set that exposes good balancing characteristics in both dimensions. This goal is particularly achieved through the definition of a set of spatial augmentation techniques properly tailored for the scope.

In order to introduce the problem of balancing a training set with respect to both the input domains and the ground truth values, we need to formalize the notion of *input collection* from which the training and the test set will be produced. Essentially, the input collection identifies the set of information that needs to be produced to obtain a useful set of data points. Clearly, such a set of information and the structure of the resulting data points greatly depends on the purpose for which the training is performed. However, in the considered application context, we can identify two main kinds of input collections: the one that serves to estimate the characteristics of a spatial dataset, and the one used to estimate some properties of a spatial operation applied on a spatial dataset. In the first case, the input collection will consist only of a set of spatial datasets and a value associated with each of them, representing the property of interest. Conversely, in the second case, besides the set of spatial datasets and associated target values, we also need a set of parameters describing the operation to be performed. The first case can clearly be seen as a specialization of the second one, where the operation parameters are omitted. For this reason, without loss of generality, in the following formalization, we concentrate on the second case, while the notation for the first one can be obtained by removing all references to the operation parameters.

Definition 3.1 (Input collection). Given a set of target variables $\{v_1, \ldots, v_w\}$ that we want to estimate with an ML/DL model M, we define an input collection C from which the data points of a training and test set are generated, as :

$$C = \{\langle D_i, O_i, T_i \rangle\}_{i=1}^{k} \tag{1}$$

where in each tuple, D_i is the spatial dataset, O_i are the parameters of the operation to be performed on D_i, and T_i is the ground truth values for the variable $\{v_1, \ldots, v_w\}$ to be estimated. The ground truth values in T_i can be real values in regression tasks or labels in classification tasks.

In the following, the set of spatial datasets in a collection C will be denoted as $\mathcal{D}$, while the set of all operation parameters in C will be denoted as $\mathcal{O}$; finally, for each target variable v_i the set of ground truth values appearing in C is denoted as $\mathcal{V}_i$.

Starting from an input collection C, the training and test set of an ML/DL model can be easily obtained by extracting from $\langle D_i, O_i \rangle \in \mathcal{D} \times \mathcal{O}$ the set of features, and from T_i the corresponding desired target values. In particular, the set of features will properly represent $D_i \in \mathcal{D}$ and $O_i \in \mathcal{O}$ in the form of vectors. For instance, a spatial dataset D_i can be described by a histogram, like in [18], or a spatial embedding, like in [5]. Similar considerations can be made for O_i, which can be used to describe, for instance, the window of a range query operation. The most suitable representation is the one that correctly and meaningfully represents the characteristics of the problem. However, its finding is out of the scope of this paper since the balancing of the input domain is done with reference to the sets $D_i \in \mathcal{D}$ and $O_i \in \mathcal{O}$, not on their specific representations.

As regards the target variables, we can observe that several different target variables can be associated with the same pair $\langle D_i, O_i \rangle \in \mathcal{D} \times \mathcal{O}$. Given a variable v_i its value $t_i \in \mathcal{V}_i$ can be computed or obtained explicitly by running a procedure or by a surveyed labeling process. The combination of each pair $\langle D_i, O_i \rangle$ with the value of a single of these variables will lead to a different training and test set. Let us consider an ML/DL model like the one in [5], which tries to estimate several parameters of a range query operation, like the selectivity, the number of MBR tests, and the execution time. In this case, every $D_i \in \mathcal{D}$ represents a spatial dataset on which several range query operations are performed, each one described by a different set of parameters $O_i \in \mathcal{O}$. For each pair $\langle D_i, O_i \rangle$, T_i is composed of three values, one for each considered metric. Starting from this situation, three training and test sets will be extracted and fed into a specific ML/DL model.

Given an input collection C, the general assumption made in this paper is that generating a new tuple $\langle D, O, T \rangle$ is a very costly operation, particularly if we want to produce a tuple such that one or more variables in T belongs to specific values. For instance, generating a new dataset and range query pair from scratch that produces a given selectivity value is difficult and time-consuming.

Starting from an input collection C, we can say that balancing the input domains of a training set means balancing the spatial characteristics of the datasets in D_i and the parameters in O_i from which such a representation has been extracted.

Definition 3.2 (Balancing of the input domains). Given a collection $C = \{\langle D_i, O_i, T_i \rangle\}_{i=1}^{k}$, C is well balanced with respect to the input domain if and only if the datasets in $\mathcal{D}$ and/or the operation parameters in $\mathcal{O}$ are well distributed in the reference space.

In the same way, we can define the notion of balance for the ground truth values by considering the set of values in $\mathcal{V}_i$ for each target variable v_i.

Definition 3.3 (Balancing of the ground truth value). Given a collection $C = \{\langle D_i, O_i, T_i \rangle\}_{i=1}^{k}$, we say that it is well balanced with respect to the ground truth values of a target variable $v_i \in \{v_1, \ldots v_w\}$

if and only if the corresponding values in $\mathcal{V}_i$ are uniformly distributed in their specific domain.

Given such preliminary definitions, it is necessary to measure and improve the balancing level of an input collection with respect to both the input domains and the ground truth values. For this purpose, Sect. 4 introduces some balancing metrics based on the well-known notion of fractal dimension, while Sect. 5 describes a set of spatial augmentation techniques that allow one to improve the balancing of an input collection.

4 Balancing Metrics

As highlighted in the previous section, balancing a training set requires, as a first step, the identification of some metrics that can quantify the distribution of a set of values in a domain.

Regarding balancing the input domains, we can analyze the degree of balancing of the spatial datasets in $\mathcal{D}$. This phase will involve the analysis of all spatial characteristics of a dataset, namely the location (spatial distribution), the extension, and the complexity of the geometries contained in it, as detailed in Sect. 4.1-4.3. In a similar way, the analysis of the level of balancing of the ground truth values can be performed as done in Sect. 4.4

4.1 Metrics for the input spatial distributions

The distribution of the geometry locations inside a dataset can be described by the well-known concept of fractal dimension, as done in [3]. The fractal dimension of a uniformly distributed spatial dataset is close to the dimension of its embedding space; thus, it is close to two if we consider datasets in 2D, while it is close to zero if all geometries of the considered dataset are concentrated around a point of the reference space. In theory, the fractal dimension can be computed only on an infinite set of points with the self-similarity property. However, an estimation of the fractal dimension of a finite set of points can be obtained by applying the box-counting method [1], and in [2, 3] its computation has been extended to a finite set of geometries. In particular, given a dataset D, a family of histograms covering its reference space and with an increasing cell size r are computed. From each histogram, a single number is obtained by summing up all the values contained in its cells; usually, the number of geometries falling in the cell is considered. This sum is called *box-counting*, and the trend of this quantity, by varying the size r of the grid cells, provides information about the dataset distribution. More than one box-counting function can be defined by considering different values for the exponent q in the following definition, producing different fractal dimensions (E_0, E_2, ...) as theoretically defined in fractal theory.

Definition 4.1 (Box-counting plot). Given a dataset D, containing a set of geometries, the *Box-counting plot* is the plot of $BC_D^q(r)$ versus r in logarithmic scale:

$$BC_D^q(r) = \sum_i (hs_D^r(i))^q \qquad with\ q \neq 1 \tag{2}$$

where $hs_D^r(i)$ is the value contained in the i-th cell of the histogram built on D with cells of size r.

Given the box-counting plot of a spatial dataset D, we can exploit the observation in [1] to determine its fractal dimension, as explained below.

Definition 4.2 (Fractal dimension). For real datasets, the box-counting plot reveals a trend of the box-counting function that, in a large interval of scale values r, behaves as a power law:

$$BC_D^q(r) \simeq \alpha \cdot r^{E_q} \tag{3}$$

where α is a constant of proportionality and E_q is a fixed exponent that characterizes the power law. E_q is an estimate of the q-fractal dimension of D, denoted as D_q, which is equal to:

$$D_q = \frac{1}{(q-1)} \cdot \frac{\partial\, log(\sum_i (hs_D^r(i))^q)}{\partial\, log(r)} \tag{4}$$

for the range of r where it is a constant value.

Given this estimate E_q of the fractal dimension of D, we can concentrate on the simplest cases: $q = 0$ and $q = 2$. Considering for simplicity only E_2, we will use the notation $E_2(D_i)$ for denoting the E_2 value computed on the dataset $D_i \in \mathcal{D}$. A measure of balancing of the spatial distributions of all datasets $D_i \in \mathcal{D}$ can be defined starting from the set of values $E_2(D_i)$ as follows.

Definition 4.3 (Measure of balancing of the spatial distributions). Given the collection of datasets $\mathcal{D} = \{D_1, \ldots, D_k\}$ contained in the input collection C, a measure of balancing of their spatial distributions can be computed with the following two steps. (a) The individual fractal dimension of each dataset $D_i \in \mathcal{D}$ is computed through the box-counting method in Def. 4.2, obtaining the list: $E_2(\mathcal{D}) = (E_2(D_1), \ldots, E_2(D_k))$ representing for each dataset $D_i \in \mathcal{D}$ the estimation of the fractal dimensions E_2.(b) The measure of balancing of the input spatial distributions can then be obtained by computing again the fractal dimension on $E_2(\mathcal{D})$:

$$B_q(\mathcal{D}, E_2()) = E_q(E_2(\mathcal{D})) \tag{5}$$

The more $B_q(\mathcal{D}, E_2())$ is close to 1, the more $\mathcal{D}$ is balanced with respect to the distribution of its spatial datasets. Regarding the choice of q, we can use $q = 0$ or $q = 2$: B_0 is close to 1 if at least one sample per class is present, while a value of B_2 near to 1 means that the samples are uniformly distributed among all classes.

4.2 Metrics for the input spatial extensions

Considering other features that characterize the content of a spatial dataset, we can concentrate on those that describe the spatial extensions of the geometries contained in the datasets of $\mathcal{D}$ and apply the same measure of balancing to them. With reference to what has been done in [18], the following measures can be considered:

- The average area of the geometries of a dataset $D \in \mathcal{D}$

$$f_{area}(D) = \frac{\sum_{g_i \in D} area(g_i)}{|D|} \tag{6}$$

  where $g_i \in D$ is a geometry in the dataset D, $area(g_i)$ is a function returning the area of g_i, and $|D|$ is the number of geometries in D.
- The average length of the x-side of the Minimum Bounding Rectangle (MBR) of the geometries in $D \in \mathcal{D}$:

$$f_{len_x}(D) = \frac{\sum_{g_i \in D} MBR(g_i).x_{max} - MBR(g_i).x_{min}}{|D|}$$

  Where $MBR(g_i)$ returns the MBR of the geometry g_i, while $MBR(g_i).x_{max}$ and $MBR(g_i).x_{min}$ are the maximum and minimum x coordinates of the MBR of g_i, respectively.

- The average length of the y-side of the MBR of the geometries in $D \in \mathcal{D}$: $f_{len_y}(D)$ can be obtained as the previous measure by simply using y_{max} and y_{min} in the equation above.

For each of these measures, we can apply a procedure analog to the one described in Def. 4.3 for obtaining a measure of balancing based on the notion of fractal dimension. In the remainder of this paper, we will consider only the balancing of the average area of the geometries in the datasets $D_i \in \mathcal{D}$. The extension to the other two measures, and eventually other measures of interest for the problem at hand, is straightforward.

Definition 4.4 (Measure of balancing of the spatial extension). Given the collection of datasets $\mathcal{D} = \{D_1, \ldots, D_k\}$ contained in the input collection C, a measure of balancing of their spatial extensions can be computed with the following two steps. (a) For each dataset $D \in \mathcal{D}$, the average area of the geometries in D is computed as in Eq. 6, obtaining the list: $f_{area}(\mathcal{D}) = (f_{area}(D_1), \ldots, f_{area}(D_k))$. (b) The measure of balancing of the input spatial extensions can then be obtained by computing the fractal dimension on $f_{area}(\mathcal{D})$:

$$B_q(\mathcal{D}, f_{area}()) = E_q(f_{area}(\mathcal{D})) \tag{7}$$

The more $B_q(\mathcal{D}, f_{area}())$ is close to 1, the more $\mathcal{D}$ contains datasets whose average geometry area covers the entire range of values. In other words, if our input collection wants to contain geometries with an average area from a value a_{min} to a value a_{max}, B_1 is close to 1 if there is at least a dataset for each meaningful discrete interval between a_{min} and a_{max}. Conversely, a value of B_2 near 1 requires that the discrete intervals are also uniformly covered by the datasets in $\mathcal{D}$.

4.3 Metrics for the input geometry complexities

The final spatial characteristic of the input domains we could want to balance is the complexity of the geometries in the datasets of $\mathcal{D}$. The complexity of the geometries can be represented by both the kind of geometries (i.e., points, linestrings, boxes, and polygons) and the average number of vertices in each geometry. Given these, and eventually other analogous measures of the geometry complexity, their balancing level can be determined through the computation of the fractal dimension of the values as done in Eq. 7.

In the experiments of this paper, we will consider only homogenous datasets containing geometric objects of the same type and the same number of vertices. Therefore, without loss of generality, we will not consider such dimension in the following.

4.4 Metrics for the ground truth values

The previous metrics concentrate on measuring the level of balancing of the input domains. However, the balancing of the ground truth values is not only the most important factor for improving the prediction capabilities of a model, but also the most challenging task to obtain.

With reference to Eq. 1, each element of an input collection can contain one or more ground truth values: $T_i = \{t_1, \ldots, t_w\}$, one for each target variable $\{v_1, \ldots, v_n\}$ to be estimated. Therefore, the balancing analysis has to be performed for every single variable v_i by comparing the corresponding set of values $\mathcal{V}_i$ appearing in C to its set of possible values. In other words, for each target variable v_i, we can apply the box-counting method to the elements in $\mathcal{V}_i$. The more the fractal dimension is close to one, the more the balancing is good.

Definition 4.5 (Measure of balancing of a target variable). Given an input collection C and a target variable v for which the set of values appearing in C is denoted as $\mathcal{V}$, the measure of balancing of v can be estimated by computing the fractal dimension on the set $\mathcal{V}$: $B_q(\mathcal{V}) = E_q(\mathcal{V})$.

Notice that this definition also requires the subdivision of the spectrum of possible values into discrete intervals of interest. This can be done in several ways that depend on the desired target to be achieved. However, as will be clear in Sect. 6, a user with the proper information can surely formulate an educated guess.

4.5 Balancing Metrics of an Input Collection

Given all the metrics introduced above for evaluating the balancing of the input domains and the ground truth values, the overall balancing of an input collection C with respect to a target variable v_i can be described by the following tuple:

$$\begin{aligned} B(C) = \langle & B_0(\mathcal{D}, E_2()), B_2(\mathcal{D}, E_2()), \\ & B_0(\mathcal{D}, f_{area}()), B_2(\mathcal{D}, f_{area}()), B_0(\mathcal{V}_i), B_2(\mathcal{V}_i) \rangle \end{aligned} \tag{8}$$

To not clutter the notation, in the following, we refer to the tuple $B(C)$ as $\langle b_1, b_2, b_3, b_4, b_5, b_6 \rangle$, where $b_1 = B_0(\mathcal{D}, E_2())$, $b2 = B_2(\mathcal{D}, E_2())$, $b3 = B_0(\mathcal{D}, f_{area}())$, $b_4 = B_2(\mathcal{D}, f_{area}())$, $b5 = B_0(\mathcal{V}_i)$, and $b6 = B_2(\mathcal{V}_i)$.

In the next section, we propose some techniques for balancing an input collection C also through the addition of new spatial datasets, synthetically generated from the available ones, and with known ground truth values for a target variable v_i.

5 Balancing techniques

Given an input collection $C = \{\langle D_i, O_i, T_i \rangle\}_{i=1}^{k}$, a balancing technique allows one to add new tuples or remove existing ones already in C in order to obtain a new input collection C' that is more balanced with respect to input domains and/or the ground truth values. A balancing technique can be formally defined as follows.

Definition 5.1 (Balancing technique). A balancing technique is a function $\mathcal{F}()$ that can be applied to an input collection $C = \{\langle D_i, O_i, T_i \rangle\}_{i=1}^{k}$ to generate a new collection $C_b = \{\langle D'_i, O'_i, T'_i \rangle\}_{i=1}^{h}$ such that: $B(C_b) \succ B(C)$. Given two balancing tuples $B(C_b) = \langle b'_1, b'_2, b'_3, b'_4, b'_5, b'_6 \rangle$ and $B(C) = \langle b_1, b_2, b_3, b_4, b_5, b_6 \rangle$, we say that:

$$\begin{aligned} & B(C_b) \succ B(C) \iff \\ & \quad b'_i \geq b_i \text{ for } 1 \leq i \leq 4 \ \wedge \ b'_j > b_j \text{ for } 5 \leq j \leq 6 \end{aligned} \tag{9}$$

Given this general definition of a balancing technique, we can identify the following three important characteristics:

- It should be able to increase the number of instances of an under-represented class without forcing the re-computation of the corresponding ground truth value. This is the most complex part of a balancing technique since it requires defining and implementing proper augmentation strategies.
- It should be able to decrease the number of instances of an over-represented class without reducing the balancing of

the input characteristics of the datasets of the class. This characteristic corresponds to having some under-sampling mechanisms that are guided by the balancing metrics.
- It should be applicable to a variable number of subdivisions of the ground truth domain. As described at the end of Sect. 4.4, balancing the domain of a target variable requires the subdivision of its domain into a set of discrete intervals in order to properly identify which cases should be artificially produced (i.e., augmented) or removed (i.e., sampled).

From this set of desiderata, it follows that a good augmentation technique can be structured in the following two steps: (a) the elements in the input collection C are firstly subdivided into a given number of classes according to the domain of the ground truth values of the considered target value, (b) a new input collection is obtained by producing new elements in the minority classes and decreasing the elements in the majority ones. The first step involves the definition of a proper partitioning technique, which essentially takes the domain $\mathcal{T}$ of a target variable and subdivides it into a set of intervals (or classes). Given them, the elements in C are associated with the corresponding class based on its own value T_i.

Definition 5.2 (Partitioning technique). Given a collection of input datasets $C = \{\langle D_i, O_i, T_i\rangle\}_{i=1}^{k}$, the finite domain $\mathcal{T} = [t_{start}, t_{end}]$ of the ground truth values of the considered target variable, and a desired number n of classes to generate, the partitioning of C with respect to $\mathcal{T}$ is produced as follows:

$$\mathcal{P}(C, \mathcal{T}) = \{C_j\}_{j=1}^{n}$$

where $\forall j = \{1, \ldots, n\}\, C_j \subseteq C$ contains the elements $\langle D_i, O_i, T_i\rangle$ such that $t_{start} + ((i-1)\cdot\delta) \leq T_i < t_{start} + (i\cdot\delta)$, and $\delta = (t_{end} - t_{start})/nc$.

This discretization of the ground truth domain allows one to identify the under-populated and the over-populated classes, as well as to select the correct augmentation or sampling technique to apply. In this regard, the second step of an augmentation technique consists of applying the correct balancing action to each class.

Definition 5.3 (Balancing action). Given a partitioning $P = \mathcal{P}(C, \mathcal{T})$, the balancing of C produces a new input collection C' by unioning the elements in the classes obtained by applying a balancing action $\mathcal{A}$ to the classes of the partition P:

$$C' = \bigcup_{C_j \in P} \mathcal{A}(C_j) \tag{10}$$

The balancing action $\mathcal{A}$ can decide to add or remove elements from a given class $C_j \in P$ according to the cardinality of that class. In particular, given the threshold $\theta = |\mathcal{D}|/nc$, where nc is the desired number of classes:

- If $|C_j| \geq \theta$, then $\mathcal{A}$ performs an under-sampling to remove some elements from the over-populated classes so that a total amount of about θ elements are present in the class. The selection can be done randomly or by trying to preserve the balancing level of the domains of the input features characterizing the class.
- If $|C_j| < \theta$, then $\mathcal{A}$ adds new elements to the under-populated class by selecting another element of C and applying an augmentation transformation that preservers the ground truth value already in C_j, or that generates a desired ground truth value not present before in C_j. In this way, a new element $\langle D, O, T\rangle$ can be added to C_j without requiring to recompute the ground truth from scratch.

Starting from this general approach, the problem is to identify a set of augmentation transformations that change the characteristics of the input datasets while preserving the ground truth value or generating new ground truth values in a controlled manner without recomputing it. In general, given a vector spatial dataset, an augmentation transformation is any transformation that moves the geometries in the reference space; thus, it can be any affine transformation, like rotation, translation, or scaling. However, with the aim to preserve the ground truth value or change it in the proper way, the specific set of augmentation techniques also depends on the target value and, eventually, the operation performed on the dataset to obtain it. In this paper, we propose the following set of augmentation operations: rotation, noise, and merge that are deeply presented in the following subsections.

5.1 Dataset Rotation

The first augmentation technique we propose is the rotation of the dataset D by an angle α around the centroid of the MBR of its reference space. In general, this affine transformation should leave the ground truth unvaried or allow one to recompute the new value of ground truth very easily.

Let us consider, for example, the case in which the input collection contains a set of datasets and the ground truth for each of them is another geometry representing the position of a point of interest. In this case, the rotation of the input dataset D also produces the rotation of the desired point p, so the new ground truth for the new dataset D' can be easily obtained by applying the same transformation on p. In a similar way, in case the input collection contains a set of elements, each one composed of a dataset D, the parameters of range query W (i.e., query window), and a target variable representing the selectivity, rotating both the dataset D and the range query W does not substantially modify the original selectivity. The selectivity of a range query is defined as follows:

Definition 5.4 (Selectivity of range query). Given a spatial dataset D and a query window $W = (x_{min}, y_{min}, x_{max}, y_{max})$, the range query operation $RQ(D, W)$ produces the following result:

$$RQ(D, W) = \{g \mid g \in D \wedge g \cap W \neq \emptyset\}$$

where $g \cap W$ returns the intersection between the geometry of the spatial object g and the query window W. The selectivity of a range query is defined as:

$$sel(RQ(D, W)) = \frac{|RQ(D, W)|}{|D|}$$

namely, it is the ratio between the cardinality of the result and the cardinality of the dataset.

From this definition, it is clear that a rotation of both the dataset D and the query window W does not substantially impact the previously computed selectivity value. More specifically, this operation can discard some original geometries because they go outside the reference space after the rotation. However, if the query window remains inside the reference space, the new selectivity value can be

easily computed starting from the result of the original query and the updated cardinality of the dataset. Fig. 1 illustrates an example of this operation applied to a dataset and its range query window.

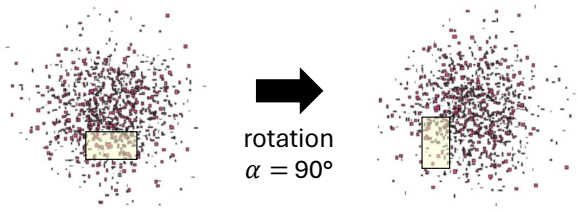

Figure 1: Example of application of a rotation operation applied to a dataset D and a range query window W.

Similar considerations can be made for other target values of range query operation, like the execution time or the number of MBR tests. Since the result of the range query operation does not change, the time required or the number of MBR tests performed to compute it are not affected by this operation.

5.2 Noise addition or removal

Inspired by the classical addition or removal of noise in image augmentation, this technique aims to add new geometries or remove existing ones inside the dataset without altering the ground truth value. The effective implementation of this technique will depend on the specific input collection and, in particular, on the considered operation and target variable.

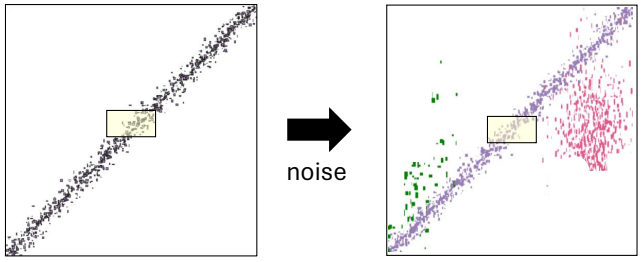

Figure 2: Example of application of a noise augmentation applied to a dataset D and a range query window W.

Let us consider again the case of an input collection generated to estimate the selectivity of the range query. In this case, the addition or removal of noise corresponds to the addition of new geometries, or the removal of existing geometries, outside or inside the query window W. This augmentation operation can be used to generate from $\langle D, O, T\rangle$ a new tuple $\langle D', O', T'\rangle$ for a specific value of T' without recomputing the range query operation from scratch. In particular, if we want to produce a new tuple for another selectivity value $T' > T$, we can alternatively add some geometries inside the range query window W or remove some geometries outside it. Fig. 2 illustrates an example of this operation applied to a dataset and its range query window.

The addition or removal of geometries outside a query window can also be used to produce a new tuple for the other two range query target variables: the execution time and the number of MBR tests. Indeed, if the range query operation is performed by exploiting the use of a spatial index [6], the addition of geometries outside the region of interest does not influence the computational cost or the number of performed comparisons.

5.3 Dataset merge

This technique essentially tries to exploit some existing datasets in $\mathcal{D}$ by combining them properly so that a new element can be produced with minimal effort. The way the combination is performed greatly depends on the target variable to be estimated. Considering the case of the range query operation again, the merging two datasets $D_1, D_2 \in \mathcal{D}$ such that we know the result of the application of a range query W on D_1 and $W \cap D_2 = \emptyset$, allows one to obtain another tuple for which the new selectivity value can be easily computed, while the required time and number of comparisons remains the same. Fig. 3 illustrates an example of this augmentation applied to two existing datasets and a range query window W,

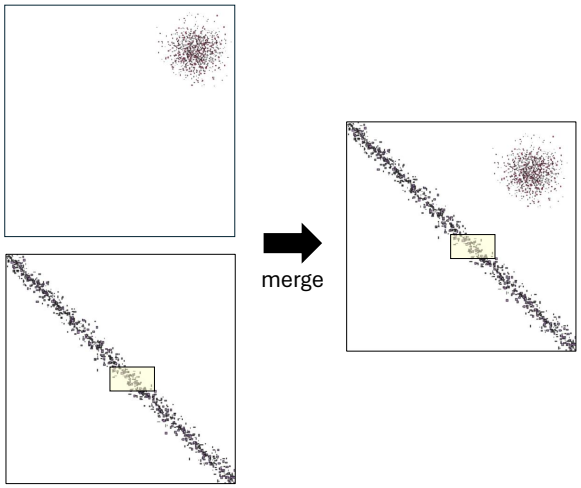

Figure 3: Example of application of a merge augmentation applied to two datasets D_1 and D_2 and a range query window W previously computed on D_1.

The following section illustrates the results of a set of experiments performed on a set of input collections synthetically generated to estimate the range query selectivity.

6 Experiments

In this section, we present a set of experiments performed with the aim to (a) explore the expressiveness of the balancing metrics proposed in Sect. 4, (b) verify the impact of the level of balancing of a training set in both the input domains and the ground truth values on the accuracies of an ML/DL model, (c) test the effectiveness of the augmentation techniques introduced in Sect. 5 in improving the balancing of a collection of spatial datasets and thus increasing the accuracy of the trained models[1].

[1]Source code available at `https://github.com/smigliorini/spatial-augmentation`.

For the purposes of these experiments, we start by synthetically generating, through the SpiderWeb tool [11], 2,000 datasets in the reference space $RS = (0, 0, 10, 10)$. These datasets are characterized by different cardinalities, contain geometries with different average areas, and have different global MBRs that are contained in RS but can be smaller than RS. Moreover, they cover all available distributions in [20]: uniform, diagonal, Gaussian, parcel, bit, and Sierpinski, some of which are illustrated in Fig. 4.

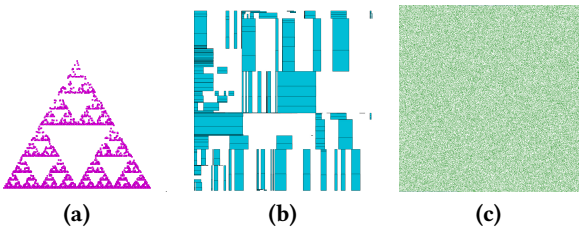

Figure 4: Examples of the considered datasets and their distributions: (a) Sierpinski, (b) parcel, (c) uniform.

Given such datasets, we build an initial input collection by performing a total of 100,000 range queries over them, varying their placement and size, and collecting three different target variables: the selectivity, the number of MBR tests, and the execution time. Range query operations are performed through the open source Beast library (Scalable Exploratory Analytics on Spatio-temporal Data) [8] and exploit the use of a very efficient spatial index, called R*-Grove [19], also provided by the same library. Relatively to the considered ML/DL model, we use the ones proposed in [5] for estimating the range query selectivity. This model is based on (i) a first layer that computes the histogram of the given dataset, (ii) a second layer that extracts the spatial embedding of the histograms, (iii) a third layer that receives the spatial embedding and the four coordinates of the query rectangle and produces the estimate of the range query parameters.

Starting from this input collection of 100,000 data points, that is called *global collection* C_0, we have first extracted a collection, called C_1^{test}, from which we generate the test set used in all the experiments. With the aim of obtaining a balanced test set, we have applied on C_1^{test} a set of balancing actions for trying to balance the input domains and the ground truth values, taking the selectivity as the target variable. This leads to a new balanced test collection denoted as $CBAL_1^{test}$.

With the aim to study the effectiveness of both the balancing metrics and the augmentation techniques, starting from the global input collection C_0, we also generate the set of collections in Tab. 1 from which different training sets are then produced.

- C_1^{train}: it contains only randomly chosen datasets with the most frequent distributions: uniform and parcel. The distribution of the ground truth values is left as is in the original collection without any balancing action being applied. The purpose of this collection is to demonstrate the effect of an imbalanced input domain on the training process.
- C_2^{train}: it contains only randomly chosen datasets with the most frequent distributions: diagonal, bit, Gaussian, and Sierpinski. The distribution of the ground truth is left as in the original collection. As in the previous case, this collection aims to show the effect of having an imbalanced input domain in the training set.
- C_3^{train}: it contains randomly chosen datasets of all distributions and with a high frequency of small values of ground truth (close to zero). In this case, we want to study the effect of having an imbalanced set of ground truth values.
- C_4^{train}: it contains randomly chosen datasets of all distributions but with a high frequency of big values of ground truth (close to 0.3). This is similar to C_3^{train} but with a strong imbalance on the opposite side of the ground truth domain.

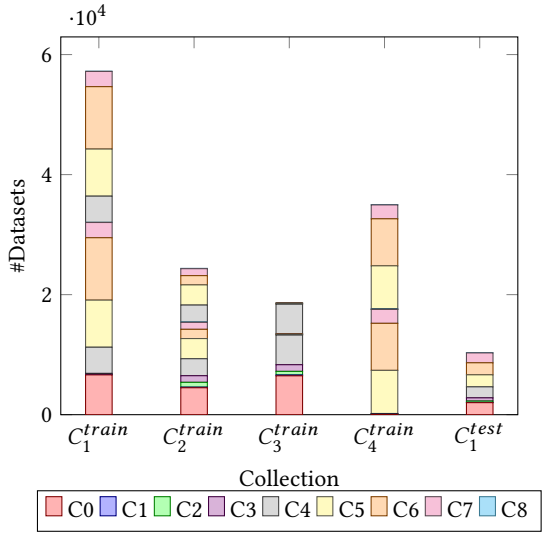

Figure 5: The number of datasets for each selectivity class with a partitioning of the input domains done with a logarithmic scale as follows: Class 0 = [0, 5*e*-9], Class 1 = [5*e*-9, 5*e*-8], Class 2 = [5*e*-8, 5*e*-7], Class 3 = [5*e*-7, 5*e*-6], Class 4 = [5*e*-6, 5*e*-5], Class 5 = [5*e*-5, 5*e*-4], Class 6 = [5*e*-4, 5*e*-3], Class 7 = [5*e*-3, 5*e*-2], Class 8 = [5*e*-2, 3*e*-1].

For each of these collections, Tab. 1 contains in the first columns the number of datasets they contain and their subdivision among all possible distributions, where "uni" means uniform, "diag" stays for diagonal, "gau" is the Gaussian, "par" means parcel and "sie" is the Sierpinski one. The table then reports the measure of balancing of each collection by showing the fractal dimension computed on the input domains, the distribution and the average area of geometries, and the ground truth values considering three target variables, i.e., selectivity, number of MBR tests and execution time.

Fig. 5 illustrates for each collection the result of applying a partitioning $\mathcal{P}$ which subdivides its datasets with respect to the ground truth of the selectivity and considering the following 9 classes or intervals {[0, 5*e*-9], [5*e*-9, 5*e*-8], [5*e*-8, 5*e*-7], [5*e*-7, 5*e*-6], [5*e*-6, 5*e*-5], [5*e*-5, 5*e*-4], [5*e*-4, 5*e*-3], [5*e*-3, 5*e*-2], [5*e*-2, 3*e*-1]}. Notice that the adopted partitioning is not linear, but we use a logarithmic scale because we are more interested in the order of magnitude of the selectivity value, rather than in its linear increment.

Table 1: Input collections for the training and test set extraction. The first section reports unbalanced collections, while the second section is related to balanced collections. For each collection, the B_0 and B_2 values for the input domain (i.e., spatial distribution and average geometry area) and the ground truth values are reported. In bold, the minimum values of B_2 (corresponding to unbalanced cases) and underlined the maximum values of B_2 (corresponding to more balanced cases).

Name	#datasets	% of distribution (uni,dia,bit,gau,par,sie)	Measure of balancing $B(C)$ (see Eq. 8)				
			input		ground truth		
			distribution (B_0, B_2)	area (B_0, B_2)	selectivity (B_0, B_2)	#MBRTests (B_0, B_2)	execution time (B_0, B_2)
Unbalanced collection							
C_1^{train}	32,036	(66,4,3,4,21,2)	(0.78,**0.61**)	(0.95,0.79)	(0.93,0.48)	(0.87,0.49)	(0.68,**0.59**)
C_2^{train}	15,433	(3,32,18,34,3,9)	(0.79,0.65)	(0.95,0.86)	(0.88,0.33)	(0.84,0.61)	(0.67,0.61)
C_3^{train}	13,460	(22,22,12,22,14,8)	(0.80,0.62)	(0.96,0.69)	(0.85,**0.15**)	(0.77,**0.41**)	(0.69,0.61)
C_4^{train}	17,577	(34,17,7,12,28,2)	(0.80,0.64)	(0.95,**0.66**)	(0.87,0.87)	(0.87,0.64)	(0.64,**0.59**)
C_1^{test}	10,316	(16,18,17,18,16,15)	(0.84,0.75)	(0.95,0.84)	(0.92,0.49)	(0.86,0.62)	(0.63,0.57)
Balanced collection							
$CBAL_2^{train}$	30,813	(3,33,19,35,3,6)	(0.78,0.65)	(0.91,0.74)	(0.89,0.64)	(0.81,0.72)	(0.63,0.61)
$CBAL_3^{train}$	40,460	(14,25,18,25,9,9)	(0.79,0.67)	(0.82,0.53)	(0.85,0.57)	(0.73,0.70)	(0.65,0.56)
$CBAL_1^{test}$	11,055	(13,21,19,21,12,13)	(0.80,0.53)	(0.89,0.71)	(0.90,0.61)	(0.74,0.63)	(0.57,0.58)

6.1 Balancing analysis

Tab. 1 allows us to analyze, for each generated collection, the level of balancing in both the input domains and the ground truth values. The first considered input domain is the spatial distribution of the geometries contained in the datasets. In this case, all collections have a $B_0(\mathcal{D}, E_2())$ and $B_2(\mathcal{D}, E_2())$ value below 1, with $B_2(\mathcal{D}, E_2())$ always lower than the corresponding value of B_0. This situation derives from the aim to study the effect of unbalanced collections and the need for at least one representative for each distribution to exploit the augmentation techniques in Sect. 5 during the balancing activity. Indeed, having a value of $B_0(\mathcal{D}, E_2())$ near 1 means that at least one representative for each spatial distribution is present in the collection, while $B_2(\mathcal{D}, E_2())$ is near 1 only when the number of representatives for each class is uniform. Conversely, as regards the second considered input domain, namely the average area of the geometries, the collections are quite balanced in this case both with respect to $B_0(\mathcal{D}, f_{area})$ and $B_2(\mathcal{D}, f_{area})$ in three cases and only with respect to $B_0(\mathcal{D}, f_{area})$ in the other two. Comparing the maximum value of B_0 for the two input domains, we can notice that even when a collection contains all the distributions in [20], it is more difficult to cover the entire domain of the spatial distributions rather than the entire range of spatial extensions, indeed $B_0(\mathcal{D}, E_2()) < B_0(\mathcal{D}, f_{area})$ for all collections.

The most interesting part is related to the balancing of the ground truth values. In this case, we can observe that a higher balancing value on the input domains does not always correspond to a higher balancing value on the ground truth values. As mentioned for the input domains, the obtained balancing measures are usually quite good for the B_0 parameter, since we tried to build them in order to cover the entire range of values, but they are usually poor in the B_2. Having complete coverage of the possible range of values will allow us to apply the augmentation techniques to improve also the values of B_2 for the considered target variable.

In Tab. 1, we can observe that even the collection with the most balanced input domains (i.e., C_2^{train}) can be imbalanced in the ground truth values. Indeed, for the selectivity, the balancing measure of C_1^{train} is better than the one of C_2^{train}, even if the first is lower balanced in both input domains. The best balancing value for the selectivity is obtained for C_4^{train}, which is among the worst with respect to both the distribution and the area.

Moreover, when the balancing of different target variables is considered, each collection can behave in an inconsistent way. For instance, C_2^{train} is quite balanced for the number of MBR tests and the execution time, but is very unbalanced for the selectivity. Conversely, C_4^{train} is quite good for the selectivity, but is less good for the other two parameters. This observation reveals that different balancing actions have to be performed on the same collection based on the selected target variable.

6.2 Augumentation and testing

Given the input collections in Tab. 1, we extract the corresponding training set from each of them, and we apply the ML model M proposed in [5] for the prediction of the range query parameters. The extraction essentially consists of computing the dataset histogram, representing the spatial dataset, and concatenating this representation with the coordinates of the range query, to form the input features of the model. A trained model is obtained from each collection in Tab. 1, all these models are tested with respect to the same balanced test set $CBAL_1^{test}$.

Tab. 2 reports the results of the test performed on each model with the same training set. For the accuracy evaluation, we use the standard *WMAPE* metric, namely the Weighted Mean Absolute

Table 2: Results of the training and testing of M for estimating the selectivity of the range query. Only the best results are reported. Hyperparameters of column Hyperpar are H1 = 64,32,32,16,16, H2 = 128,64,64,32,32, H3 = 256,128,128,64,64, H4 = 512,256,256,128,128, H5 = 1024,512,512,256,256. Column TT reports the training time in seconds and the number of epochs inside brackets.

Collection	Hyperparam	WMAPE	TT	
Unbalanced collections				
C_1^{train}	H5	0.756	594	(54)
C_2^{train}	H3	0.994	149	(17)
C_3^{train}	H4	0.996	67	(10)
C_4^{train}	H4	0.871	401	(42)
Balanced collections				
$CBAL_2^{train}$	H1	0.409	594	(80)
$CBAL_3^{train}$	H2	0.456	613	(80)

Percentage Error, since it allows us to correctly treat zeros in the set of actual and predicted values. It is a variant of MAPE in which the mean absolute percentage error is treated as a weighted arithmetic mean. Most commonly, the absolute percent errors are weighted by the actual values, which leads to the following formula: $WMAPE = \sum_{i=1}^{n} |A_i - P_i| / \sum_{i=1}^{n} |A_i|$, where A_i are the actual value, while P_i are the predicted value. The table also reports the required training time, which clearly increases as the cardinality of the training set increases, as well as the number of used epochs. In this regard, we can notice that the number of epochs can be less than the set value (i.e., 80) if the training is prematurely stopped since no improvements of the loss function are observed for more than six epochs. This is another indicator that the training set is not sufficiently good to perform profitable learning.

The obtained results confirm that building a training set from an unbalanced collection does not produce good results. In particular, the worst results are obtained for C_2^{train}, C_3^{train} and C_4^{train}, which are the collections with the lower values of B_2 with respect to the selectivity. This confirms that the proposed measures are effective in identifying the goodness of a training set.

Given the values in Tab. 1 and Tab. 2, we decide to concentrate on the two worst cases: C_2^{train} and C_3^{train}, and apply to them the augmentation techniques proposed in Sect. 5. The new balancing metrics for the corresponding balanced datasets are reported in the second part of Tab 1 with the name $CBAL_2^{train}$ and $CBAL_3^{train}$, respectively. After the augmentation actions, in both cases, the value of B_2 for the selectivity is much better than the original ones. In particular, for $CBAL_2^{train}$, it reaches the value of 0.64 starting from an initial value of 0.33, while for $CBAL_2^{train}$, it becomes 0.57 in place of 0.15. As already explained, the value of B_0, in this case, does not change so much since the original datasets have been built in a way to ensure that at least one representative for each class is present, while the number of elements in each class is very unbalanced. We can also notice that this augmentation performed with reference to the selectivity also has a positive effect on another target variable, namely the number of MBR tests, since these two are highly related to each other. Conversely, no substantial improvement has been made in the balancing of the execution time.

These new datasets are used to extract two new training sets for the same model M, and the obtained results are also reported in Tab. 2. The obtained WMAPE error confirms that using a more balanced training set allows one to obtain more accurate models. Moreover, we can see that also the maximum number of epochs is reached in these cases, suggesting that the learning process is completely performed.

7 Conclusion

This paper presents an extensive discussion about the balancing problem in training and test sets with reference to ML/DL models dealing with spatial datasets and spatial operations. First of all, a general formalization of the problem is provided, and a measure of balancing for the input domains and the ground truth values is proposed, which is based on the notion of fractal dimension and its box-counting estimation. Secondly, a set of augmentation techniques for spatial input collections is provided by focusing on a specific spatial operation that is the well-known range query. Finally, experiments are reported, which illustrate the applicability of the proposed balancing measures, the effects of having unbalanced training sets on the accuracy of ML/DL models, and demonstrate the benefits of the proposed spatial augmentation techniques. The performed experiments reveal that producing a balanced training set is not a simple task, even when only the balancing of the input domains is considered and particularly when the balancing of the ground truth values is desired. Moreover, since producing new data points in a training set for specific ground truth values is often a costly operation, tailored balancing techniques need to be proposed that allow the generation of new data points without recomputing the ground truth from scratch.

The obtained results confirm the goodness of the approach and encourage further investigation in this direction, in particular in the identification of other spatial augmentation techniques that can be generally applied or are tailored for other kinds of spatial operations. Another future work consists of increasing the efficiency of the proposed augmentation techniques in terms of space and time requirements for their execution. In this regard, massive exploitation of the index structures, as well as the use of big data paradigms like MapReduce, can help in achieving such results.

Acknowledgments

This study was carried out within the Interconnected Nord-Est Innovation Ecosystem (iNEST) and received funding from the European Union Next-GenerationEU (Piano Nazionale di Ripresa e Resilienza (PNRR) – Missione 4 Componente 2, Investimento 1.5 – D.D. 1058 23/06/2022, ECS00000043). This manuscript reflects only the authors' views and opinions, neither the European Union nor the European Commission can be considered responsible for them.

References

[1] Alberto Belussi and Christos Faloutsos. 1998. Self-spacial Join Selectivity Estimation Using Fractal Concepts. *ACM Trans. Inf. Syst.* 16, 2 (1998), 161–201. https://doi.org/10.1145/279339.279342

[2] Alberto Belussi, Sara Migliorini, and Ahmed Eldawy. 2018. Detecting skewness of big spatial data in SpatialHadoop. In *Proceedings of the 26th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (SIGSPATIAL 2018)*. ACM, Seattle, WA, USA, 432–435. https://doi.org/10.1145/3274895.3274923

[3] Alberto Belussi, Sara Migliorini, and Ahmed Eldawy. 2020. Skewness-Based Partitioning in SpatialHadoop. *ISPRS International Journal of Geo-Information* 9, 4 (2020), 201. https://doi.org/10.3390/ijgi9040201

[4] Alberto Belussi, Sara Migliorini, and Ahmed Eldawy. 2022. Spatial embedding: a generic machine learning model for spatial query optimization. In *Proceedings of the 30th International Conference on Advances in Geographic Information Systems* (Seattle, Washington, USA) *(SIGSPATIAL '22)*. Association for Computing Machinery, New York, NY, USA, Article 26, 4 pages. https://doi.org/10.1145/3557915.3560960

[5] Alberto Belussi, Sara Migliorini, and Ahmed Eldawy. 2024. A Generic Machine Learning Model for Spatial Query Optimization based on Spatial Embeddings. *ACM Trans. Spatial Algorithms Syst.* (apr 2024). https://doi.org/10.1145/3657633 Just Accepted.

[6] Ahmed Eldawy, Louai Alarabi, and Mohamed F. Mokbel. 2015. Spatial partitioning techniques in SpatialHadoop. *Proc. VLDB Endow.* 8, 12 (2015), 1602–1605. https://doi.org/10.14778/2824032.2824057

[7] Ahmed Eldawy, Mostafa Elganainy, Ammar Bakeer, Ahmed Abdelmotaleb, and Mohamed Mokbel. 2015. Sphinx: distributed execution of interactive SQL queries on big spatial data. In *Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems (SIGSPATIAL '15)*. Article 78, 4 pages. https://doi.org/10.1145/2820783.2820869

[8] Ahmed Eldawy, Vagelis Hristidis, Saheli Ghosh, Majid Saeedan, Akil Sevim, A.B. Siddique, Samriddhi Singla, Ganesh Sivaram, Tin Vu, and Yaming Zhang. 2021. Beast: Scalable Exploratory Analytics on Spatio-temporal Data. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management* (Virtual Event, Queensland, Australia) *(CIKM '21)*. Association for Computing Machinery, New York, NY, USA, 3796–3807. https://doi.org/10.1145/3459637.3481897

[9] Ahmed Eldawy and Mohamed F Mokbel. 2015. Spatialhadoop: A MapReduce framework for spatial data. In *31st IEEE International Conference on Data Engineering (ICDE)*. 1352–1363. https://doi.org/10.1109/ICDE.2015.7113382

[10] Zeshan Hussain, Francisco Gimenez, Darvin Yi, and Daniel Rubin. 2018. Differential Data Augmentation Techniques for Medical Imaging Classification Tasks. *AMIA Annu Symp Proc* 2017 (apr 2018), 979–984.

[11] Puloma Katiyar, Tin Vu, Ahmed Eldawy, Sara Migliorini, and Alberto Belussi. 2020. SpiderWeb: A Spatial Data Generator on the Web. In *Proceedings of the 28th International Conference on Advances in Geographic Information Systems* (Seattle, WA, USA) *(SIGSPATIAL '20)*. Association for Computing Machinery, New York, NY, USA, 465–468. https://doi.org/10.1145/3397536.3422351

[12] Jia Shijie, Wang Ping, Jia Peiyi, and Hu Siping. 2017. Research on data augmentation for image classification based on convolution neural networks. In *2017 Chinese Automation Congress (CAC)*. 4165–4170. https://doi.org/10.1109/CAC.2017.8243510

[13] Connor Shorten and Taghi M. Khoshgoftaar. 2019. A survey on Image Data Augmentation for Deep Learning. *Journal of Big Data* 6, 1 (2019), 60. https://doi.org/10.1186/s40537-019-0197-0

[14] MingJie Tang, Yongyang Yu, Qutaibah M. Malluhi, Mourad Ouzzani, and Walid G. Aref. 2016. LocationSpark: A Distributed In-Memory Data Management System for Big Spatial Data. *Proc. VLDB Endow.* 9, 13 (2016), 1565–1568.

[15] Luke Taylor and Geoff Nitschke. 2018. Improving Deep Learning with Generic Data Augmentation. In *IEEE Symposium Series on Computational Intelligence ((SSCI))*. 1542–1547. https://doi.org/10.1109/SSCI.2018.8628742

[16] Tin Vu, Alberto Belussi, Sara Migliorini, and Ahmed Eldawy. 2021. A Learned Query Optimizer for Spatial Join. In *Proceedings of the 29th International Conference on Advances in Geographic Information Systems* (Beijing, China) *(SIGSPATIAL '21)*. Association for Computing Machinery, New York, NY, USA, 458–467. https://doi.org/10.1145/3474717.3484217

[17] Tin Vu, Alberto Belussi, Sara Migliorini, and Ahmed Eldawy. 2022. Towards a Learned Cost Model for Distributed Spatial Join: Data, Code & Models. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management* (Atlanta, GA, USA) *(CIKM '22)*. Association for Computing Machinery, New York, NY, USA, 4550–4554. https://doi.org/10.1145/3511808.3557712

[18] Tin Vu, Alberto Belussi, Sara Migliorini, and Ahmed Eldawy. 2024. A learning-based framework for spatial join processing: estimation, optimization and tuning. *The VLDB Journal* (13 Feb 2024). https://doi.org/10.1007/s00778-024-00836-1

[19] Tin Vu and Ahmed Eldawy. 2020. R*-Grove: Balanced Spatial Partitioning for Large-Scale Datasets. *Frontiers in Big Data* 3 (2020). https://doi.org/10.3389/fdata.2020.00028

[20] Tin Vu, Sara Migliorini, Ahmed Eldawy, and Alberto Belussi. 2022. *Spatial Data Generators* (1 ed.). Association for Computing Machinery, New York, NY, USA, 13–24. https://doi.org/10.1145/3548732.3548736

[21] Dong Xie, Feifei Li, Bin Yao, Gefei Li, Liang Zhou, and Minyi Guo. 2016. Simba: Efficient In-Memory Spatial Analytics. In *Proceedings of the 2016 International Conference on Management of Data (SIGMOD '16)*. 1071–1085. https://doi.org/10.1145/2882903.2915237

[22] Jimin Wang Yingjie Hu, Zhipeng Gui and Muxian Li. 2022. Enriching the metadata of map images: a deep learning approach with GIS-based data augmentation. *International Journal of Geographical Information Science* 36, 4 (2022), 799–821. https://doi.org/10.1080/13658816.2021.1968407

[23] Jia Yu, Jinxuan Wu, and Mohamed Sarwat. 2016. A demonstration of GeoSpark: A cluster computing framework for processing big spatial data. In *32nd IEEE International Conference on Data Engineering (ICDE)*. 1410–1413.