The Scientific Journal of Cihan University – Sulaimaniya **PP: 46-60**
Volume (7), Issue (1), June 2023
ISSN 2520-7377 (Online), ISSN 2520-5102 (Print)

Improve Search Performance for R+ tree Learned Spatial Index

Galawizh M. Najeeb[1, 2], Nzar A. Ali[3,4]

[1] Technical College of Informatics, Sulaimani Polytechnic University, Sulaimani, Iraq
[2] Technical College of Engineering, Sulaimani Polytechnic University, Sulaimani, Iraq
[3] School of Administration and Economic, Department of Statistics and Informatics, University of Sulaimani, Sulaimani, Iraq
[4] Department of Computer Science, Cihan University- Sulaimaniya, Sulaimaniya, Iraq

Emial: galawizh.najeeb@spu.edu.iq[1,2] , nzar.ali@univsul.edu.iq[3]

Abstract:

The R^+ tree algorithm is used to give index to each spatial data sets of (point, line polygon spatial objects). Our map datasets contain 1048575 records for point, as well as for line and polygon each. Machine learning approach used to learn the indices. Then nearest neighbor queries (NNQ) are carried out in purpose of evaluating system performance for proposed learned index. The evaluation of proposed indexing done based on query execution time. Execution times of k-Nearest neighbor query in learned methods for all kind of spatial objects (point, line, and polygon) are calculated, that 25 random sample points were took for each of them. In order to compare learnt and conventional indexing, we also developed the traditional indexing using the R+ tree technique. So, a significant finding that goes beyond the suggested approach (R^+ Learned Spatial Index (R^+ LSI)) is that we reached less execution times in learned index method for nearest neighbor queries for all three spatial objects. Getting benefits of Microsoft SQL Server database for handling spatial objects indices. We propose that machine learning (ML) allows for the independent creation of specific index structures by making it possible to design a model that reproduce the patterns in the data. We explore the extent to which learning models may replace traditional index structures like R^+ tree. Finally measuring the learned index (R^+ LSI) is done by getting benefits of both parameters which are, mean square error (MSE) and R-square (R^2). The estimator is linear regression which is used in machine learning approach to make a model. Enhancing spatial query performance is done through the minimum execution time of spatial NNQ in this research.

Key words: Learned index, R^+ tree, nearest neighbor spatial query.

الملخص:

تُستخدم خوارزمية R^+ الشجرة لإعطاء فهرس لكل مجموعات البيانات المكانية (نقط , مضلع, خطي , كائنات مكانية). تحتوي مجموعات بيانات الخرائط الخاصة بنا على 1048575 تسجيلًا للنقطة ، بالإضافة إلى الخط والمضلع, لكل منها. يشتخدم نهج التعلم الآلي لتعلم المؤشرات. ثم يتم تنفيذ الاستعلامات الاقرب المجاورة (NNQ) بغرض تقييم أداء النظام للفهرس المتعلم المقترح. تم تقييم الفهرسة المقترحة بناءً على وقت تنفيذ الاستعلام. تم حساب أوقات تنفيذ الاستعلام المجاور الأقرب k في الطرق التي تم تعلمها لجميع أنواع الكائنات المكانية (النقطة والخط والمضلع) ، حيث تم أخذ 25 نقطة عينة عشوائية لكل منها. من أجل مقارنة الفهرسة المتعلمة والتقليدية ، قمنا أيضًا بتطوير الفهرسة التقليدية باستخدام تقنية R^+ الشجرة. لذلك ، فإن النتيجة المهمة التي تتجاوز النهج المقترح (R^+ Learned Spatial Index (R^+ LSI)) هي أننا وصلنا إلى أوقات تنفيذ أقل في طريقة الفهرس المكتسبة لأقرب الاستعلامات المجاورة لجميع الكائنات المكانية الثلاثة. الحصول على فوائد قاعدة بيانات Microsoft SQL Server للتعامل مع فهارس الكائنات المكانية. نقترح أن يسمح التعلم الآلي (ML) بالإنشاء المستقل لهياكل الفهرس المحددة من خلال إتاحة تصميم نموذج

DOI: http://dx.doi.org/10.25098/7.1.3

The Scientific Journal of Cihan University – Sulaimaniya **PP: 46-60**
Volume (7), Issue (1), June 2023
ISSN 2520-7377 (Online), ISSN 2520-5102 (Print)

يعيد إنتاج الأنماط في البيانات. نستكشف المدى الذي يمكن أن تحل فيه نماذج التعلم محل هياكل الفهرس التقليدية مثل شجرة R^+. أخيرًا ، يتم قياس المؤشر المكتسب (R^+ LSI) من خلال الحصول على مزايا كل من المعلمات التي هي، متوسط الخطأ التربيعي (MSE) و (R^2) R-square. المقدر هو الانحدار الخطي الذي يستخدم في نهج التعلم الآلي لعمل نموذج. يتم تحسين أداء الاستعلام المكاني من خلال الحد الأدنى من وقت تنفيذ NNQ المكاني في هذا البحث.

الكلمات المفتاحية: الفهرسة المتعلمة، شجرة R^+، الجوار الاقرب للاستعلام المكانى.

پوختە:

ئەلگۆریتمیR^+ درەختی بەکاردێت بۆ پێدانی ئیندێکس بە هەر کۆمەڵە داتایەکی فەزایی لە (خاڵ، هێڵ, و فرەگۆشە). کۆمەڵە داتاکانی نەخشەکەمان 1048575 تۆمار لەخۆدەگرێت بۆ خاڵ، هەروەها بۆ هێڵ و فرەگۆشە بۆ هەر هەریەکەیان . ڕێبازی فێربوونی ئامێر کە بۆ فێربوونی پێوەرەکان بەکاردێت. پاشان پرسیارەکانی نزیکترین دراوسێ (NNQ) بە مەبەستی هەڵسەنگاندنی ئەدای سیستەم بۆ پێوەرە فێربووە پێشنیارکراوەکان ئەنجام دەدرێن. هەڵسەنگاندنی ئیندێکسی پێشنیار کراو کە بە پشتبەستن بە کاتی جێبەجێکردنی پرسیار ئەنجام دەدرێت. کاتی جێبەجێکردنی پرسیاری دراوسێی نزیکترین k-لە شێوازە فێربووەکاندا بۆ هەموو جۆرە شتێکی فەزایی (خاڵ، هێڵ و فرەگۆشە) لەبەرچاو بگیرێت، کە 25 خاڵ نمونەی هەڕەمەکی بۆ هەریەکەیان وەرگیراوە. بۆ ئەوەی ئیندێکسکردنی فێربوو و ئاسایی بەراورد بکەین، ئێمە ئیندێکسکردنی تەقلیدیشمان پەرەپێدا بە بەکارهێنانی تەکنیکەکانی R^+ tree. کەواتە، دۆزینەوەیەکی گرنگ کە لە ڕێبازی پێشنیار کراو تێدەپەڕێت (R^+ Learned Spatial Index)(R^+ LSI) ئەوەیە کە ئێمە گەیشتینە کاتەکانی جێبەجێکردنی کەمتر لە شێوازی فێربووی ئیندێکس بۆ نزیکترین پرسیارەکانی دراوسێ بۆ هەر سێ شتە فەزاییەکە. بەدەستهێنانی سوودەکانی بنکەدراوەی Microsoft SQL Server بۆ مامەڵەکردن لەگەڵ پێوەرەکانی شتە فەزاییەکان. ئێمە پێشنیار دەکەین کە فێربوونی ئامێر (ML) ڕێگە بە دروستکردنی سەربەخۆی پێکهاتەی ئیندێکسی تایبەت دەدات بە ڕێگەدان بە دیزاینکردنی مۆدێلێک کە نەخشەکانی ناو داتاکان بەرهەمبهێنێتەوە. ئێمە لێکۆڵینەوە لەوە دەکەین کە تا چەند مۆدێلی فێربوون ڕەنگە جێگەی پێکهاتە ئیندێکسی تەقلیدی وەک R^+ درەختی بگرێتەوە. لە کۆتاییدا پێوانەکردنی پێوەرە فێربووەکان (R^+ LSI) بە بەدەستهێنانی سوودی هەردوو پارامێترەکە ئەنجام دەدرێت کە بریتین لە، هەڵەی چوارگۆشەی ماماوەند (MSE) و پێوانەی R دووجا, (R^2) . خەملێنەرەکە بریتییە لە پاشەکشەی هێڵی کە لە ڕێبازی فێربوونی ئامێردا بەکاردێت بۆ دروستکردنی مۆدێلێک. بەرزکردنەوەی ئەدای پرسیاری فەزایی لە ڕێگەی کەمترین کاتی جێبەجێکردنی NNQ فەزایی لەم لێکۆڵینەوەیەدا ئەنجام دەدرێت.

کلیلە وشە: ئیندێکسی فێربوون، درەختی R^+، نزیکترین دراوسێی شوێنی.

DOI: http://dx.doi.org/10.25098/7.1.3

The Scientific Journal of Cihan University – Sulaimaniya **PP: 46-60**
Volume (7), Issue (1), June 2023
ISSN 2520-7377 (Online), ISSN 2520-5102 (Print)

1. INTRODUCTION

Geometric types of data or spatial data types offer a basic concept for simulating the interactions, attributes, and operations of objects located in space, along with their geometrical form. Objects in space provided from point, line, polygon, and data having higher dimensions are called spatial data. The aim of R^+ tree is to retain spatial query performance improving; it could deal with less execution time Nearest Neighbor Inquiries [1]. Sellis et al. and Faloutsos et al. (1987) [2] introduced the R^+ tree, a variation of the R-tree. Hierarchical data structures include R^+ trees. They are used to dynamically organize a collection of d-dimensional geometric objects, which are represented by minimal bounding d-dimensional rectangles (abbreviated MBRs in the sequel) [3]. Modern systems utilize index structures that are extensively specialized towards improving query efficiency to keep up with the volume of data that is created daily [4]. A sensible spatial index structure may significantly increase the storage efficiency and query performance of the geographic database. A book's catalogue, for instance, might be thought of as an index. This catalog makes it easier to comprehend the book's overall structure and makes it simple for readers to find the pages they wish to read. If there is no appropriate and effective index structure, the search will take a long time, especially when working with a lot of spatial data. As a result, creating an effective index structure is crucial for retrieving spatial data. Grid index structure and R tree index structure are used mostly in the current spatial databases for data query processing[5]. The goal of the learnt index is to learn a function that associates a search key with a data object's storage location[6]. The fact that we do not argue that learned indexes should completely replace traditional index structures must be emphasized. Instead, this study's main contribution which complements earlier work is to explain and evaluate the potential of a special approach for creating indexes. Multi-dimensional objects can be handled in several ways. Finding the minimal bounding rectangle (MBR) of the given item is how the techniques handle handling more complicated objects, such as circles, polygons, etc. examining techniques for dealing with multidimensional points, as these raise numerous insightful points that also apply to rectangles. There are some methods for dealing with objects, first, methods for multi-dimensional Points Partitioning of a region, known as a split. By inserting one (or more) hyper planes that also divide a region into distinct sub-regions, the split is accomplished [7]. These techniques are done depending on position, dimensionality and locality attributes as shown in table (1).

Table (1): Illustration N-Dimensional Methods [6]

Attribute	Position	Dimensionality	Locality
Methods	grid file	quad-trees	grid file
	K-D-B-trees	oct-trees	hierarchical decomposition
	k-d trees	k-d trees	

DOI: http://dx.doi.org/10.25098/7.1.3

The Scientific Journal of Cihan University – Sulaimaniya **PP: 46-60**
Volume (7), Issue (1), June 2023
ISSN 2520-7377 (Online), ISSN 2520-5102 (Print)

A second method is methods for rectangles, a- methods that transform the rectangles into points. Therefore, one of the previously mentioned methods for storing points can be chosen. k-d trees, grid file, after a rotation of the axes. In order to prevent an asymmetric distribution of points that would compromise the efficiency of the grid file, rotation is required. b- techniques for mapping a k-d space onto a 1-d region that make use of space filling curves. Such as z-transform, transform k-dimensional objects to line segments [6]. And Quadtree that scan pixels in a k-dimensional space. c- Methods that Divide the Original Space into Appropriate Sub-regions (overlapping or disjoint). Any of the previously described methods for points can be used to deconstruct the space if the regions are not connected. The possibility that a rectangle may cross a splitting hyper plane presents the only challenge. One option is to divide the problematic rectangle into two sections, then tag each piece to show that it is a part of the same rectangle. R-trees are suitable for components which are symmetric and either points or regions. We used R^+ tree, it is a particular kind of R-tree that prevents the performance decrease brought on by overlapping sections. A height-balanced tree with intermediate and leaf nodes makes up the information representation [8].

Mean Square Error (MSE), is a measure of the quality of an estimator. It assesses the average squared difference between the observed and predicted values. When a model has no error, the MSE equals zero. The percentage of the variation in the dependent variable that can be predicted from the independent variable is a function of coefficient of determination in statistics and is symbolized by the R^2, its range is 0 to 1. As the R^2 values are close to 1, it is the optimal situation, our estimated R^2 values provide us a confidence of our suggested solution.

2. RELATED WORK

In research [9] a variation to R trees, R^+ trees, is introduced. All algorithms needed to search, update and pack the structure are thoroughly examined, and the analytical findings show that R^+ While exploring files with numbers of rectangles, it can reduce the number of disk requests by up to 50% compared to an R tree. Lauther [10] and Rosenberg [11] employed kd-trees. Nievergelt and Hinrichs [12] Following rotating the axes, utilizing the grid file was advised. [13] Proposed new R-tree packing techniques for static databases. Considering a set of rectangles, they sort items while creating a bottom-up R tree. The innovation of this work is the use of fractals, and specifically the Hilbert curve, to achieve better ordering of the rectangles and eventually better packing. It was suggested to use a hybrid index structure made of a 3D multi-level adaptive grid as well as an R^+ tree. [14]Proposed a nearest neighbor query algorithm based on space-filling curve grid division.

Using the dimensionality reduction and data clustering properties of the space-fulfilling curve, this technique ordered the points in the grid sequentially. By traveling to the query point's location and the points in the grid's immediate vicinity, one may determine their nearest neighbor. [15], The R^+ tree and multi-level grid made up the majority of the index architecture. The extent, wide, and depth of the grid were determined after the set of data was analyzed using the multi-level automated grid technique based on normal distribution. Furthermore, the information area was rapidly and effectively partitioned using a multi-level adaptive grid structure, and optimal indexing was accomplished by getting the benefit of the no overlapping of the intermediate nodes of the R^+ tree. [16] In order to expedite the precise nearest neighbor approach in high-dimensional Euclidean space, this work investigates lower bound-based methods. Utilizing block vectors and the Cauchy-Schwartz

DOI: http://dx.doi.org/10.25098/7.1.3

The Scientific Journal of Cihan University – Sulaimaniya **PP: 46-60**
Volume (7), Issue (1), June 2023
ISSN 2520-7377 (Online), ISSN 2520-5102 (Print)

inequality, they calculate the lower bound of Euclidean Distance. In order to allow effective similarity query processing in metric spaces, we offer a unique indexing strategy termed LIMS that makes use of data clustering, pivot-based data transformation methods, and learning indexes. The underlying data is divided into clusters in LIMS such that each cluster has a comparatively homogeneous distribution of data[17]. In [18], the authors propose a brand-new search structure that is GPU-friendly and is predicated on nearest neighbor graphs and information spreading on graphs. Their approach is developed to take benefit of GPU structures to speed up the hierarchical development of the index structure and query execution. [19] Presents Flood, a multi-dimensional in-memory read-optimized index that automatically adjusts to a specific dataset and workload by coordinating index structure and data storage layout optimization.

3. Machine Learning Approach

Machine learning is a method of data analysis that automates the analytical model building process. It is a subset of artificial intelligence that is predicated on the notion that computers are capable of learning from data, recognizing human behavior, and making decisions with little to no human involvement. Despite the fact that many machine learning algorithms have been around for a while, it is still difficult to automatically conduct intricate mathematical operations on massive volumes of data often and quickly. Is an updated development. As long as it uses a consistent method for learning from data, machine learning can undoubtedly automate learning. The data is used to train the Learned Index models. Their most important discovery is that performance may be greatly enhanced by utilizing models that can adjust to the data distribution in order to generate a reasonable prediction as to where a key actually lies[20]. Regression models, instance-based algorithms, decision trees, Bayesian techniques, and artificial neural networks are just a few of the machine learning algorithms proposed in this field that are pertinent to the learning task [21].

4. Spatial Data Structure R^+ tree

Techniques for clustering and spatial indexing must both be considered in a spatial access strategy. Without a spatial index, it is necessary to do a "full table scan" in a relational database to determine whether each item in the database satisfies the spatial selection requirement. In fact, this is inappropriate for interactive usage and the majority of other applications since geographic data sets are frequently quite vast. A spatial index is therefore necessary in order to effectively locate the relevant object addresses without having to look at every item. Index structures are the solution whenever efficient data access is required, and a broad range of options are available to meet the varying demands of distinct access patterns [22]. An R^+ tree is a method for looking up data using a location, often (x, y) and often for locations on the. It is both a height-balanced tree as well as a variation of the utilized for. R^+ tree handle the cases “forking” easily, because they split these large data objects into smaller ones, and also the R^+ tree, can be used in a database system in order to index any kind of geometric data [23]. An example of R tree data structure is represent in Fig (1).

DOI: http://dx.doi.org/10.25098/7.1.3

The Scientific Journal of Cihan University – Sulaimaniya **PP: 46-60**
Volume (7), Issue (1), June 2023
ISSN 2520-7377 (Online), ISSN 2520-5102 (Print)

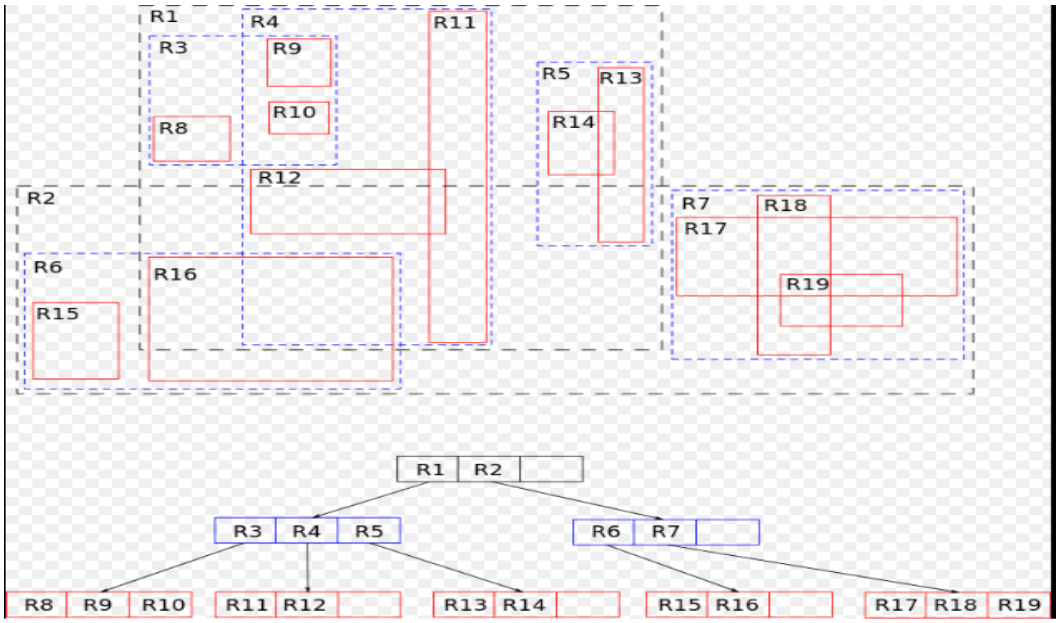

Fig. 1: Example of an R-tree for 2D Rectangles [20]

5. Query Processing over the Learned R^+ tree Index

Many scholars have worked diligently to develop a variety of indexing techniques for geographical data in order to speed up the processing of spatial queries. It has become increasingly common to parallelize the creation of geographical indexes and the execution of queries to increase efficiency.

Due to the volume, computational complexity, and length of time that complicated geographical searches need, support for high speed inquiries on spatial data has become crucial. Minimizing index generation and query execution time has always been the driving force behind spatial index and geographic query research [24]. This research trend has continued with the development of parallel processing methods for the execution of spatial queries and spatial indexes. For quick data retrieval, a lot of work has recently been put into indexing spatial data and performing spatial queries.

Implementation of Nearest Neighbor (NN) inquiries. By using R^+ tree data structure, the index, of every point is determined. And by getting benefits of machine learning method we learned the indices of each spatial objects (point, line and polygon). Then the nearest neighbor query is implemented over indices. And finally determining execution time of query processing. Using the learned R^+ index, for handling spatial query, we can create efficient algorithms using learned index.

6. Work Procedure

The steps outlined in this paper are divided into four groups:

1. Based on the grid file transformation approach, The First Contribution offers an efficient transformation method for both (line and polygon) spatial objects actual datasets. Transformation is carried out in order to make spatial object datasets with (x, y) coordinates and reduce data complexity. As well as building a spatial data warehouse as a first step.
2. In the second step, spatial indexing for each spatial item is obtained using an R^+ tree.
3. The Third step utilizes several steps: In the first step, machine learning method used in order

DOI: http://dx.doi.org/10.25098/7.1.3

The Scientific Journal of Cihan University – Sulaimaniya **PP: 46-60**
Volume (7), Issue (1), June 2023
ISSN 2520-7377 (Online), ISSN 2520-5102 (Print)

to creating indices for each spatial objects as traditional techniques. In the second step, using some useful libraries such as pandas, etc. for dealing with array of datasets, reading datasets, learning indices and calculating execution time, also well-known Classifier for classification purpose. Representing the optimal algorithm has been improved by using a prediction technique that optimizes the learned method SLI regarding the number of points and determining different distances. Finally, a new fast and reliable learned algorithm is introduced.

4. Fourth step: creating models depending on three different datasets of three types of spatial objects by using linear regression approach.

In order to be able to deal with proximity queries, and finding the specific data object an efficient spatial indexing strategy is needed. However, none of traditional solutions (indexing) is effective, hence specific structures are needed to deal with spatial queries like R^+ trees. The main advantage of R^+ trees is the improved search performance, especially in the case of exact match queries. For many computer vision, data analysis, and machine learning tasks, the k nearest neighbor analysis is a crucial process. The procedure work of our proposed learned index is presented and described in fig (2). The work starts from taking spatial datasets, transformation process, and getting index to each objects elements then learned indices, after that spatial query process and getting results.

Point, line, and polygon datasets (objects latitude and longitude real datasets) are system input. Data preparation is carried out by transforming it into one-dimensional data (1D). After that, executing indexing spatial data and learning that indices. Following that, a spatial nearest neighbor query is implemented, and finally the execution time of the query is calculated.

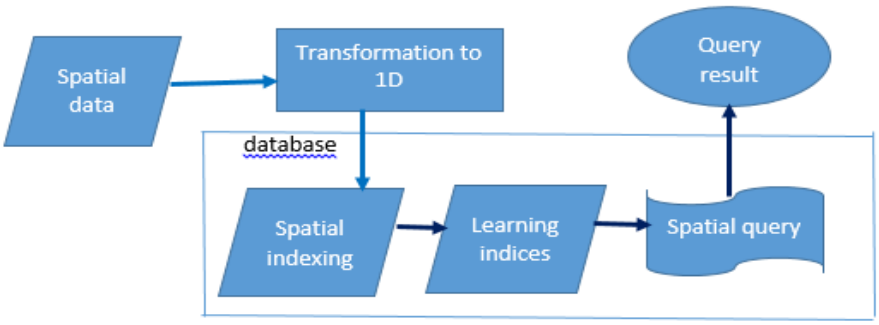

Fig. 2: Proposed System Workflow

6.1. Datasets of Point, line and Polygon Spatial Objects preparation

We focused on spatial objects like points, lines, and polygons. Each object has a unique collection of datasets, there are 1048575 records for points in our map databases, along with lines and polygons. thus as a first step, spatial indexing utilizing the previously stated R^+ tree technique as well as standard indexing is applied to the point object dataset. Following the implementation of nearest neighbor queries, we determined the individual query execution times for R^+ tree. The following phase involved learning point indexing for R^+ tree algorithm, implementing nearest neighbor queries, and computing query execution times in order to compare and contrast both approaches (traditional and

DOI: http://dx.doi.org/10.25098/7.1.3

The Scientific Journal of Cihan University – Sulaimaniya **PP: 46-60**
Volume (7), Issue (1), June 2023
ISSN 2520-7377 (Online), ISSN 2520-5102 (Print)

learned). Line and polygon with their various datasets are the research's next two goals. We also used lines and polygons as point objects, applying and implementing the work technique.

7. Proposed Algorithm Results (R+LSI)

Testing and implementing of both algorithms reached us significant results which are query execution time. The results are saved in separate tables (about 27 tables), for more description we will present in brief the average execution time tables (table: 2, 3 and 4) for each spatial objects for different sample points in different distances. Figure 4 shows an example of our work test and implantation of both techniques, for each 25 random sample points we calculated the execution time and also number of found nearest neighbors objects. Actually we just focused on the duration time. As a second step, figure (5) show the tested results tables for point spatial objects. And so on for other two spatial objects which are line and polygon. The R^+ learnt spatial index algorithm (R^+LSI), which can implements on each of the 3 categories of spatial objects; points, lines, and polygons was learnt after working with R^+ tree. We achieve the following noteworthy and important findings by conducting a finding the nearest neighbor and calculating execution times for both traditional and learnt spatial indices. Highlights from our research utilizing both conventional and newly taught spatial indices are shown in the tables below. Execution time for queries in milliseconds (second), computed at various distances (kilometer) for various points (25 points). Table (2) illustrates average execution times of nearest neighbor query for both traditional and learned algorithms using R^+ technique index data structure for point spatial objects respectively.

Distances	**Number of found objects**	**Nearest Neighbor Query Execution Time for P (15.2 , 9.26)**	
		Without Machine Learning	**With Machine Learning**
5 (km)	46	0.6477	0.0049
50 (km)	4562	0.6624	0.0030
100 (km)	18280	0.6690	0.0019
150 (km)	40602	0.6362	0.0029
200 (km)	67043	0.6531	0.0039
250 (km)	96853	0.6366	0.0033

Fig. (4): Testing and Implementing Result Table of Tradition and Learned Methods in Different Distances

DOI: http://dx.doi.org/10.25098/7.1.3

The Scientific Journal of Cihan University – Sulaimaniya **PP: 46-60**
Volume (7), Issue (1), June 2023
ISSN 2520-7377 (Online), ISSN 2520-5102 (Print)

Query Execution Time of 25 Different points in Different Distances- Traditional Method						
Points	**5 (km)**	**50(km)**	**100(km)**	**150(km)**	**200(km)**	**250(km)**
p1	0.6333	0.6415	0.6238	0.6139	0.6226	0.6303
p2	0.6636	0.6624	0.6529	0.6455	0.6472	0.6629
p3	0.6881	0.6969	0.6723	0.6529	0.6449	0.6439
p4	0.6658	0.6568	0.6461	0.6457	0.6491	0.6433
p5	0.6679	0.6602	0.6565	0.6522	0.6515	0.6470
p6	0.6510	0.6638	0.6329	0.6527	0.6390	0.6628
p7	0.6629	0.6505	0.6387	0.6400	0.6331	0.6491
p8	0.6665	0.6666	0.6469	0.6516	0.6387	0.6316
p9	0.6580	0.6669	0.6371	0.6533	0.6428	0.6338
p10	0.6642	0.6679	0.6389	0.6378	0.6347	0.6430
p11	0.6559	0.6516	0.6449	0.6359	0.6501	0.6414
p12	0.6586	0.6581	0.6479	0.6592	0.6343	0.6387
p13	0.6617	0.6586	0.6445	0.6594	0.6416	0.6470
p14	0.6523	0.6741	0.6506	0.6586	0.6328	0.6579
p15	0.6466	0.6788	0.6486	0.6381	0.6481	0.6431
p16	0.6688	0.6703	0.6386	0.6511	0.6579	0.6356
p17	0.6575	0.6717	0.6428	0.6464	0.6338	0.6341
p18	0.6613	0.6635	0.6463	0.6518	0.6432	0.6516
p19	0.6611	0.6725	0.6536	0.6445	0.6484	0.6337
p20	0.6598	0.6633	0.6390	0.6455	0.6651	0.6983
p21	0.6410	0.6571	0.6330	0.6551	0.6986	0.6985
p22	0.6634	0.6666	0.6449	0.6451	0.6490	0.6493
p23	0.6477	0.6624	0.6690	0.6362	0.6532	0.6366
p24	0.6623	0.6604	0.6600	0.6471	0.6474	0.6472
p25	0.6652	0.6524	0.6560	0.6414	0.6556	0.6404
Everage ET	**0.6594**	**0.6638**	**0.6466**	**0.6464**	**0.6465**	**0.6480**

Fig. (5): Representation the Tests Table - Traditional Method

DOI: http://dx.doi.org/10.25098/7.1.3

The Scientific Journal of Cihan University – Sulaimaniya **PP: 46-60**
Volume (7), Issue (1), June 2023
ISSN 2520-7377 (Online), ISSN 2520-5102 (Print)

Query Execution Time of 25 Different points in Different Distances- Learned Method						
Points	**5 (km)**	**50(km)**	**100(km)**	**150(km)**	**200(km)**	**250(km)**
p1	0.0064	0.0029	0.0100	0.0100	0.0027	0.0100
p2	0.0040	0.0040	0.0030	0.0030	0.0039	0.0050
p3	0.0020	0.0021	0.0051	0.0056	0.0066	0.0067
p4	0.0049	0.0030	0.0030	0.0021	0.0845	0.0085
p5	0.0040	0.0020	0.0030	0.0020	0.0414	0.0521
p6	0.0019	0.0030	0.0020	0.0040	0.0050	0.0050
p7	0.0060	0.0032	0.0030	0.0030	0.0039	0.0040
p8	0.0050	0.0040	0.0040	0.0030	0.0040	0.0025
p9	0.0040	0.0030	0.0030	0.0030	0.0039	0.0040
p10	0.0040	0.0030	0.0023	0.0030	0.0030	0.0038
p11	0.0069	0.0030	0.0030	0.0040	0.0046	0.0047
p12	0.0050	0.0030	0.0030	0.0020	0.0025	0.0025
p13	0.0060	0.0040	0.0030	0.0030	0.0041	0.0042
p14	0.0050	0.0040	0.0040	0.0040	0.0039	0.0027
p15	0.0050	0.0040	0.0030	0.0030	0.0039	0.0040
p16	0.0049	0.0051	0.0098	0.0099	0.0089	0.0088
p17	0.0060	0.0030	0.0030	0.0030	0.0039	0.0039
p18	0.0040	0.0040	0.0020	0.0030	0.0030	0.0042
p19	0.0020	0.0030	0.0030	0.0030	0.0039	0.0039
p20	0.0050	0.0030	0.0030	0.0030	0.0054	0.0065
p21	0.0050	0.0030	0.0030	0.0040	0.0062	0.0065
p22	0.0050	0.0030	0.0030	0.0030	0.0052	0.0043
p23	0.0050	0.0030	0.0020	0.0030	0.0039	0.0034
p24	0.0062	0.0030	0.0030	0.0030	0.0014	0.0071
p25	0.0060	0.0040	0.0030	0.0040	0.0034	0.0032
Everage ET	**0.0048**	**0.0033**	**0.0036**	**0.0037**	**0.0089**	**0.0069**

Fig. (6): Representation the Tests Table - Learned Method

(The table that shows in figure (5 and 6) just related to point spatial object)

DOI: http://dx.doi.org/10.25098/7.1.3

The Scientific Journal of Cihan University – Sulaimaniya **PP: 46-60**
Volume (7), Issue (1), June 2023
ISSN 2520-7377 (Online), ISSN 2520-5102 (Print)

Table (2): Traditional and Learned Index - point objects

Execution Time Average of Nearest neighbors query **without** Machine learning(in second)		Execution Time Average of Nearest neighbors query with Machine learning(in second)
5km	0.65938	0.00477
50km	0.66380	0.00328
100km	0.64663	0.00356
150km	0.64644	0.00374
200km	0.64650	0.00894
250km	0.64803	0.00686

Table three illustrate nearest neighbors query's average time to execute (second) of both traditional and learned algorithms using R^+ tree technique for line spatial objects respectively.

Table (3): Traditional and Learned Index - Line object

Execution Time Average of Nearest neighbors query **without** Machine learning(in second)		Execution Time Average of Nearest neighbors query **with** Machine learning(in second)
5 km	0.33681	0.00509
50 km	0.34030	0.00338
100 km	0.33743	0.00334
150 km	0.34090	0.00332
200 km	0.34169	0.04259
250 km	0.34359	0.04286

Table four illustrate average execution time (s) of nearest neighbors query for both traditional and learned algorithms using R^+ tree technique for polygon spatial objects respectively.

Table (4): Traditional and Learned Index - Polygon object

Execution Time Average of Nearest neighbors query without Machine learning(in second)		Execution Time Average of Nearest neighbors query without Machine learning(in second)
5km	0.01459	0.00357
50 km	0.02087	0.00266
100 km	0.02488	0.00250
150 km	0.03953	0.00253
200 km	0.05455	0.02679
250 km	0.06282	0.03239

DOI: http://dx.doi.org/10.25098/7.1.3

The Scientific Journal of Cihan University – Sulaimaniya **PP: 46-60**
Volume (7), Issue (1), June 2023
ISSN 2520-7377 (Online), ISSN 2520-5102 (Print)

Graph of different query execution time in R^+ tree technique for point spatial object dataset (real dataset) in various distances in kilometer (km) clearly presented in figure (7) for traditional and learned algorithms respectively.

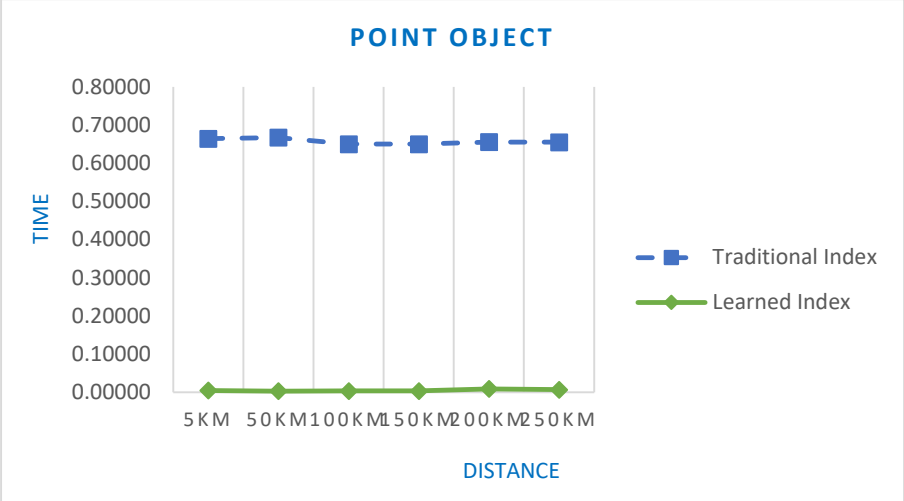

Fig. (7): Traditional and Average Execution Time of Point object

Figure (8) for traditional and learnt algorithms, respectively, clearly display the graph of varied query execution times in R^+ tree approach for line spatial object datasets (real dataset) in various distances in kilometer

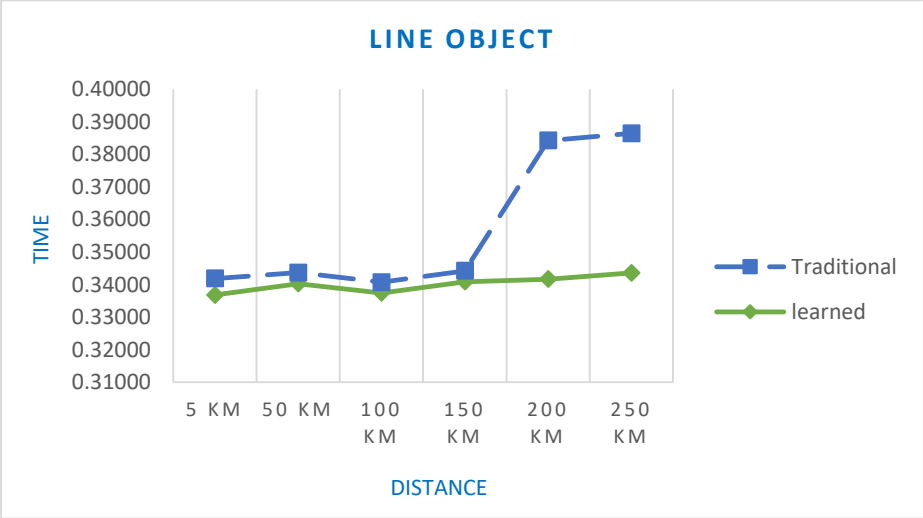

Fig. (8): Traditional and Average Execution Time of Line object

Graph of different query execution time in R^+ tree technique for polygon spatial object dataset (real dataset) in various distances in kilometer (km) clearly presented in figure (9) for traditional and learned algorithms respectively.

DOI: http://dx.doi.org/10.25098/7.1.3

The Scientific Journal of Cihan University – Sulaimaniya **PP: 46-60**
Volume (7), Issue (1), June 2023
ISSN 2520-7377 (Online), ISSN 2520-5102 (Print)

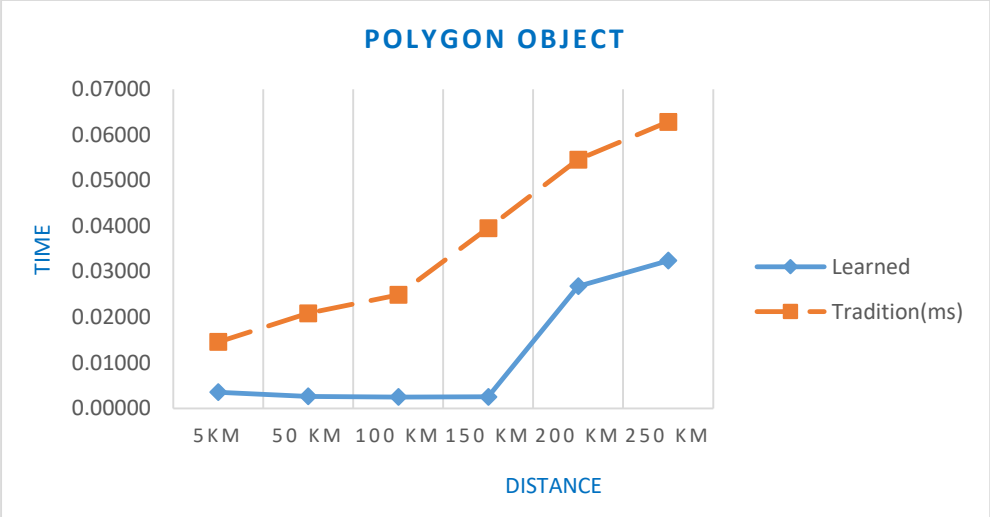

Fig. (9): Traditional and Learned Average Execution Time of Polygon object

8. Evaluation Criteria

The main goal of our study is comparing two learnt and traditional spatial indexing algorithms. Utilizing execution time in millisecond (s) as a performance metric. In the first step of work using R^+ data structure in case of getting index for each spatial objects datasets point, line and polygon in both traditional and learned indexing methods as second step. Then implementing nearest neighbors query for 25 different points on all various spatial object elements, point, line and polygon. And calculating execution time in both methods for each spatial data type in order to evaluate query performance of mentioned methods. Also various performance evaluation indicators were selected for the assessment of the model including coefficient of determination (R^2), and mean square error (MSE), their values for point, line and polygon for R^+ LSI are shown table (5):

Table (5): MSE and R^2 Values of R^+LSI

NNQ - R^+ LSI		
Spatial Objects	Mean Square Error(MSE)	R^2
point	1.9322	0.9981
line	2.1414	0.8877
polygon	2.8065	0.8889

DOI: http://dx.doi.org/10.25098/7.1.3

The Scientific Journal of Cihan University – Sulaimaniya **PP: 46-60**
Volume (7), Issue (1), June 2023
ISSN 2520-7377 (Online), ISSN 2520-5102 (Print)

9. Conclusion

This study presents the evaluation between two different indexing methods, traditional and learning, and implementing nearest neighbor query in both algorithms. Using real datasets of point, line and polygon spatial objects. Our work has proved that spatial indexing based on machine learning, (R^+ Spatial Learned Index) has super advantage over traditional in term of less query execution time including all spatial objects of a kind. We have reached these conclusions, objects indexing with R^+ tree has an advantage over Quadtree and Hilbert curve (our previous work) [25], because of its less query execution time in traditional method also learned index. The performance of the learned models was challenged using both R^2, MSE measurements. We have demonstrated that by leveraging the distribution of the data being indexed, trained indexes may offer important advantages.

10. Future Work

As a future work, it would be of high interest to compare the performance of this solution with different spatial index. Also, further improvement and investigation on algorithm presented in this paper should be done. First of all, investigating if there are other features of Z-order-curve, apart from Z-value jumps dependent on y values, that could be used to improve accuracy. Secondly it may be possible to introduce one more layer of machine learning models, learned on the errors made by linear regressions.

Reference

1. Güting, R.H., *An introduction to spatial database systems.* the VLDB Journal, 1994. **3**(4): p. 357-399.
2. Jia, L., et al., *Efficient 3D Hilbert Curve Encoding and Decoding Algorithms.* Chinese Journal of Electronics, 2022. **31**(2): p. 277-284.
3. Comer, D., *Ubiquitous B-tree.* ACM Computing Surveys (CSUR), 1979. **11**(2): p. 121-137.
4. Groh, F., et al., *Ggnn: Graph-based gpu nearest neighbor search.* IEEE Transactions on Big Data, 2022.
5. Liu, Y., et al., *Research on Hybrid Index Based on 3D Multi-Level Adaptive Grid and R+ Tree.* IEEE Access, 2021. **9**: p. 146010-146022.
6. Gu, T., et al., *A Reinforcement Learning Based R-Tree for Spatial Data Indexing in Dynamic Environments.* arXiv preprint arXiv:2103.04541, 2021.
7. Sellis, T., N. Roussopoulos, and C. Faloutsos, *The R+-Tree: A Dynamic Index for Multi-Dimensional Objects*. 1987.
8. Böhm, C., S. Berchtold, and D.A. Keim, *Searching in high-dimensional spaces: Index structures for improving the performance of multimedia databases.* ACM Computing Surveys (CSUR), 2001. **33**(3): p. 322-373.
9. Samet, H., *The quadtree and related hierarchical data structures.* ACM Computing Surveys (CSUR), 1984. **16**(2): p. 187-260.
10. Rosenberg, J.B., *Geographical data structures compared: A study of data structures supporting region queries.* IEEE transactions on computer-aided design of integrated circuits and systems, 1985. **4**(1): p. 53-67.

DOI: http://dx.doi.org/10.25098/7.1.3

11. Hinrichs, K. and J. Nievergelt, *The grid file: a data structure designed to support proximity queries on spatial objects.* ETH Eidgenössische Technische Hochschule Zürich, Institut für Informatik, 1983. **54**.
12. Kamel, I. and C. Faloutsos. *On packing R-trees*. in *Proceedings of the second international conference on Information and knowledge management*. 1993.
13. Kamel, I. and C. Faloutsos, *Hilbert R-tree: An improved R-tree using fractals*. 1993.
14. Zhang, L., et al., *Query method for nearest region of spatial line segment based on Hilbert curve grid.* International Journal of Innovative Computing Information and Control, 2019. **15**(4): p. 1287-1307.
15. Triebel, R., P. Pfaff, and W. Burgard. *Multi-level surface maps for outdoor terrain mapping and loop closing*. in *2006 IEEE/RSJ international conference on intelligent robots and systems*. 2006. IEEE.
16. Zhang, H., Y. Dong, and D. Xu, *Accelerating exact nearest neighbor search in high dimensional Euclidean space via block vectors.* International Journal of Intelligent Systems, 2022. **37**(2): p. 1697-1722.
17. Tian, Y., et al., A Learned Index for Exact Similarity Search in Metric Spaces. arXiv preprint arXiv:2204.10028, 2022.
18. Guttman, A. R-trees: A dynamic index structure for spatial searching. in Proceedings of the 1984 ACM SIGMOD international conference on Management of data. 1984.
19. Nathan, V., et al. Learning multi-dimensional indexes. in Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data. 2020.
20. Ding, J., et al. ALEX: an updatable adaptive learned index. in Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data. 2020.
21. Janiesch, C., P. Zschech, and K. Heinrich, Machine learning and deep learning. Electronic Markets, 2021. **31**(3): p. 685-695.
22. Kraska, T., et al. *The case for learned index structures*. in *Proceedings of the 2018 international conference on management of data*. 2018.
23. Kang, M.-A. and K.-J. Li. *Query processing methods for connectivity search in visual databases using R+-tree*. in *Working Conference on Visual Database Systems*. 1995. Springer.
24. Singh, H. and S. Bawa, *A survey of traditional and mapreducebased spatial query processing approaches.* ACM SIGMOD Record, 2017. **46**(2): p. 18-29.
25. Najeeb, G.M. and N.A. Ali, *In the Context of Spatial Data, a Comparison between Learning and Traditional Indexing.* Harbin Gongye Daxue Xuebao/Journal of Harbin Institute of Technology, 2022. **54**(5): p. 1-8.

DOI: http://dx.doi.org/10.25098/7.1.3