# A Novel Local Geometry Capture in Pointnet++ for 3D Classification

Shivanand Venkanna Sheshappanavar and Chandra Kambhamettu
Video/Image Modeling and Synthesis (VIMS) Lab., Dept. of Computer and Information Sciences
University of Delaware, Newark, DE, USA 19716
{ssheshap, chandrak}@udel.edu

## Abstract

*Few of the recent deep learning models for 3D point sets classification are dependent on how well the model captures the local geometric structures. PointNet++ model made remarkable progress in learning local geometric structures than its predecessor PointNet. It recursively applies PointNet on nested partitions of the input 3D point set. PointNet++ model was able to extract the local region features from points by ball querying the local neighborhoods. However, ball querying is less effective in capturing local neighborhoods of high curvature surfaces or regions. In this paper, we demonstrate improvement in the 3D classification results by using ellipsoid querying around centroids, capturing more points in the local neighborhood. We extend the ellipsoid querying technique by orienting it in the direction of principal axes of the local neighborhood for better capture of the local geometry. We then take the union of points grouped by ball querying and ellipsoid querying with re-orientation to improve the PointNet++ classification results by **1.1%**. Furthermore, we demonstrate the impact of re-oriented ellipsoid querying on a state-of-the-art ball query-based model, Relation-Shape Convolutional Neural Network (RS-CNN), with a **0.8%** improvement in classification accuracy on ModelNet40 dataset.*

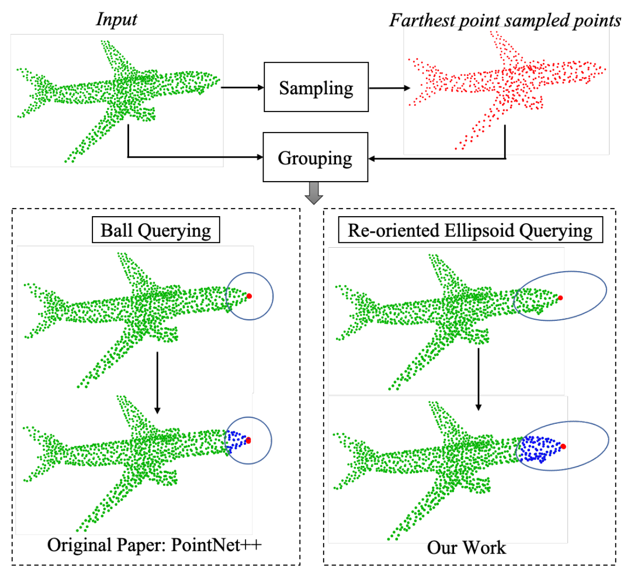


Figure 1: Visualization of Input 3D point cloud (green colored points), farthest point sampled (FPS) points (red colored points), ball queried and re-oriented ellipsoid queried points (blue colored points) at one of the FPS point on a 3D point cloud object - an airplane.

## 1. Introduction

Over the last decade, application of deep neural networks to the field of computer vision evolved tremendously and achieved state-of-the-art results for several computer vision tasks that were previously addressed by traditional computer vision algorithms. Few of the problems which have received major attention include detection, classification, tracking, and segmentation of real-world objects for 2D images as well as 3D datasets.

Convolution Neural Networks(CNNs) have been very successful with 2D image datasets. Using convolutional filters, CNNs have achieved better generalization with a reduced number of parameters. CNNs could not be applied directly to 3D datasets such as point sets because of their inability to fit into lattice grids, unlike 2D images. 3D point sets have several applications in the field of autonomous driving, robotics, virtual reality applications, large scale 3D reconstruction, etc. However, extracting features related to the shape of an object from these point sets is challenging.

3D point sets are an unordered set of points that demand permutation invariance for any learned representation. Transforming these 3D point sets to any other formats such as multi-view images or voxels have proven to increase complexity due to the sparse nature of volumetric datasets and also would lose valuable inherent geometric features. Another challenge is how to vary sampling density of the points from the underlying shape in order to pick points to

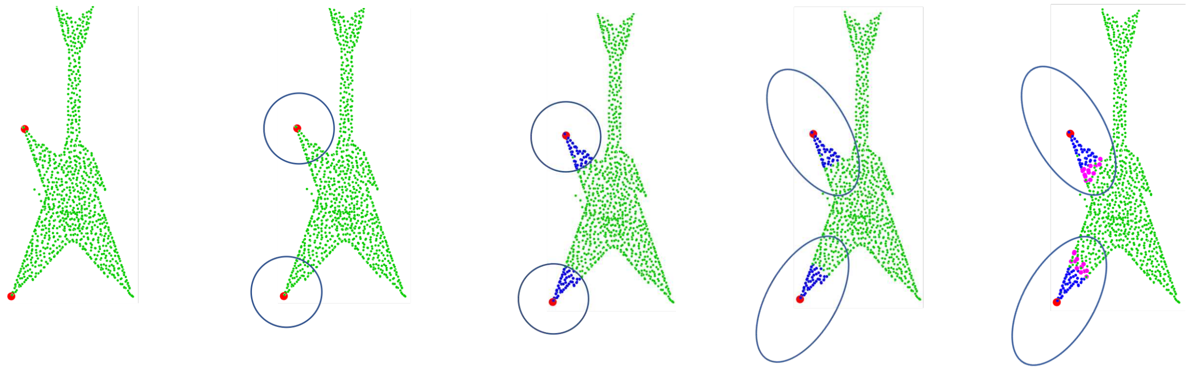

Figure 2: Visualization of ball querying followed by re-oriented ellipsoid querying. **(a)** Input points (green color) with two centroid points (red color) **(b)** Ball query at two centroid points **(c)** Neighborhood points from ball querying (blue color) **(d)** Re-oriented ellipsoid querying at the two centroid points **(e)** Neighborhood points from ellipsoid querying (blue+pink color).

generate 3D pointsets. Also, varying distribution of the 3D point sets affects the robustness of the deep networks.

Recent deep neural network models such as PointNet [1] started a trend in processing raw 3D point cloud data directly into multi-layer perceptrons (MLPs) and aggregate global features using max-pooling for object classification of 3D point sets. This model's successor, PointNet++ [2] partitions the input point sets and applies hierarchical structuring to multiple subsets in the form of local neighborhood captured around a set of points obtained by farthest point sampling of the input point cloud and then uses MLPs with max-pooling to extract global features.

Hierarchical structuring is achieved through multiple set abstraction layers by processing point sets to produce new but fewer points. Each of these set abstraction layers consists of three layers: sampling, grouping, and pointnet layer. The sampling layer uses farthest point sampling to select a set of points called centroids of the local regions from input points. Using these centroids and the original point sets, grouping layer finds neighboring points around each of the centroids capturing the local neighborhood. PointNet layer encodes local region patterns from the grouped points into feature vectors using a miniaturized Pointnet model.

Ball querying, defined in the grouping layer, is the central idea that captures the local neighborhood around centroids on the object surface. With each centroid point as the center, a ball of radius $r$ captures the local neighborhood points. However, a sphere that lacks orientation gets restricted from accessing the maximum overlap of the object surface. In the case of rolling ball [3], few sample points could not be reached by the pivoting ball when the curvature of the manifold is more significant than a given threshold. Visualization of grouping in PointNet++, as in figure 1, shows fewer points captured by ball querying. An increase in the radius to fetch more points around the centroid would not improve overall classification accuracy, as we have tried doing this in our experiments. This is most likely because points far away from the centroid, not relevant to the local shape are being picked due to increase in the ball size.

Alternatively, an ellipsoid, due to its shape, gives an advantage over a sphere if aligned correctly to the local neighborhood. In this paper, we demonstrate how a re-oriented ellipsoid querying around each of the centroids, increases the capture of local neighborhoods. Our ellipsoid based querying has captured more relevant neighborhood points around a given centroid, as shown in figure 2. With a sample of two centroids, the ellipsoid with a different orientation at each centroid captures more points. The points from the ball query determine the orientation of the ellipsoid.

Following two techniques are our key contributions to achieve better 3D object classification accuracy for 3D point sets. First technique queries the neighborhood only once, while the second technique queries the neighborhood twice:

- Ellipsoid querying: capture points from the input points around each centroid (sampled points) using ellipsoid for better capture of the local neighborhood.
- Re-oriented ellipsoid querying: capture points from the input points around each centroid using the ball or ellipsoid querying, compute principal axes from the captured points, reorient input points and query again for better capture of the local neighborhood.

## 2. Related Works

An increase in 3D point sets data has also increased the focus on learning features from 3D point sets. Volumetric model [4] has converted these 3D point sets into volumetric grids in order to apply 3D convolution. However, such conversion would result in the quantization of 3D points because the 3D grid imposes lower resolution resulting in quantization loss of the shape. Also, operations on the volumetric grids would result in higher computational costs mainly because the voxel-based representation of 3D data is inefficient due to sparse occupancy of the voxels.

Despite the efforts from models such as OctNet [5], which reduced the problem of sparse voxels, they still faced higher memory occupancy problems. Deep neural network models that used voxels as a replacement for 3D point sets required high-resolution grids in order to keep quality representation, suffering a higher memory consumption. Other approaches [6, 7] that converted 3D data into 2D RGB with depth image in order to exploit CNNs could not capture proper geometric relationships between 3D points because the neighboring pixels in the 2D format could be geometrically away from each other in 3D format.

The PointNet [1] model was a pioneer in using a raw 3D point cloud without incurring any pre-processing overhead. The model popularized the use of max-pooling in deep learning models for 3D point set classification. Although max-pooling failed to capture local features, it did considerably good in capturing global features compared to average pooling. The idea of using multi-layer perceptrons (MLP) for 3D classification, originally proposed in [8] proved to improve the results when combined with max-pooling in PointNet model. However, neither application of MLP to individual points nor max-pooling of features would significantly capture neighborhood information.

Among the models [2, 9, 10] to better capture local structures, PointNet++ [2] partitions the point sets to produce common hierarchical structures across the partitions, and recursively applies pointnet as a local feature learner to learn contextual representation. In a convolutional setting, the pointnet shares local feature weights. This hierarchy would segment a given point set into smaller clusters before passing it on to the pointnet. For high dimensional features, pointnet is called recursively on these smaller clusters. PointNet++ also uses max-pooling to aggregate the features and employs single scale, multi-scale, and multi-resolution grouping methods to extract features. These methods start from small local regions and extend to more significant regions or use different resolutions of the regions to extract features. Max-pooling in PointNet and PointNet++ limits PointNet's ability to examine the contextual neighborhood structure of the points.

PointGrid [11] uses less memory compared to volumetric models and proposes an integrated point and grid hybrid 3D convolutional network model to represent the local geometry shape details better. Models such as [10], which mined local structures, captured additional information of neighborhood point's type. E.g., if a given point is a corner point or a convex point or a concave point or a planar point. Use the concept of kernel correlation to measure the geometric affinity of point sets. Construct k-nearest neighbor graphs to utilize neighborhood information. On each of the node's neighborhood, they recursively apply max-pooling with the insight that local points share similar geometric structures improving results compared to PointNet.

To exploit surface deformations, a geometric approach [12] uses different ways of applying convolution operations to point sets, especially to local regions called patches. PointCNN [13] transforms the input points using a $\mathcal{X}$-operator. PointCNN simultaneously weighs and permutes the input features associated with the points generalizing CNNs. Although SPLATNet [14] loses geometric information, it applies bilateral convolution to an input point cloud with sparse and efficient lattice filters and computes hierarchical and spatially-aware features.

PCNN [15] uses an extended version of volumetric convolution on the point cloud, providing a flexible framework. PCNN maintains the point cloud format and extend it over an ambient space into a continuous volumetric function. Deep parametric continuous CNNs [16] introduced a new learnable operator called parametric continuous convolution. Its deep neural network is based on computable relations among points, but the network does not explicitly learn from local to global like classic CNN.

PointConv[17] performs convolution on 3D pointsets by using non-uniform sampling and then compensates by using a technique called PointConv, which uses an inverse density scale in order to re-weight the continuous function learned. PointCNN[13] is unable to achieve the desired permutation-invariance, for point clouds. Deep kd-network models such as [18] achieve results comparable to PointNet++[2] but are memory inefficient mainly because they depend upon partitioning the bounding volume instead of partitioning the local geometric shape. PointWeb [19] uses the interaction between points in each local neighborhood region by exhaustively capturing context information between all point pairs. Although Dynamic Graph Convolutional Neural Networks (DGCNNs) [20] captures better local geometric features by an innovative approach called EdgeConv, in many cases, it is unreliable in maintaining permutation invariance.

The geometry sharing method [21] uses the Eigendecomposition of grouped points. However, it restricts itself to the Eigendecomposition of points queried by the kNN method with fixed k value. For ball querying, the number of points in each group varies, making it challenging to process. State-of-the-art models such as Relation Shape CNN [22] learn geometric topology constraint among points and DensePoint [23] learns sufficient contextual semantic information to get the grasp of the elusive shape of an object. Structural Relation Network model [24] processes the sub cloud features twice to reason between geometrical and relational features.

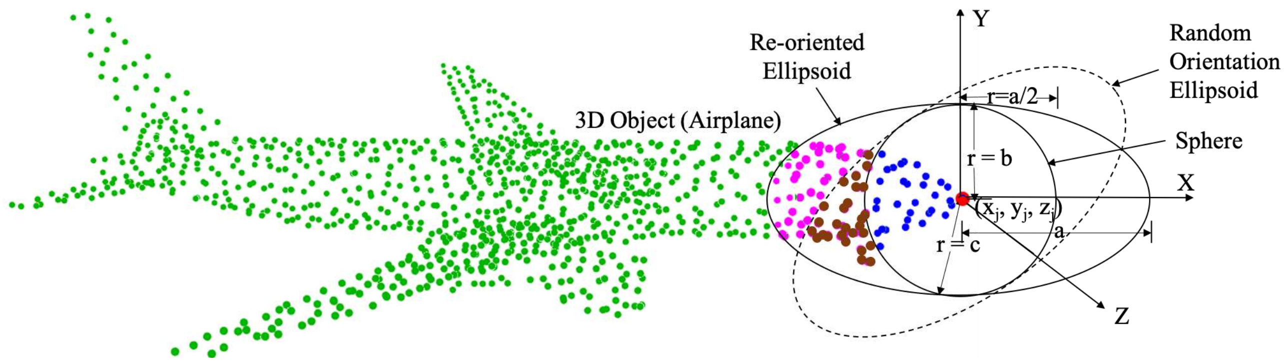


Figure 3: Visualization of ball, random ellipsoid and re-oriented ellipsoid with the centroid $(x_j, y_j, z_j)$, given $a$ = $2r$ and $b$ = $c$ = $r$. The red point is the centroid, blue points are ball queried points, blue+brown points are queried by random orientation ellipsoid, and blue+brown+pink points are captured by re-oriented ellipsoid. A random orientation ellipsoid represented by dashed lines captures lesser points unless its orientation is the same as the re-oriented ellipsoid (Best viewed in color).

Since the inception of PointNet++, several models have attempted to capture local geometry of 3D objects. However, the idea of capturing local geometry using geometric shapes such as ellipsoid instead of using a ball sphere was unexplored. Our paper focuses on enhancing the geometrical aspect of capturing local geometry of objects by investigating the impact of ellipsoid based querying on the 3D point set for 3D object classification. The idea is to exploit the ellipsoid's re-orientation capability compared to the uniform ball based querying, capturing more geometrically meaningful points from the local neighborhood.

## 3. Our Method

A point $(x_i, y_i, z_i)$ is selected from the input point set $xyz_1$, if it lies on or within the surface of the ball sphere with radius $r$ and center ($x_j$, $y_j$, $z_j$) (from the farthest point set $xyz_2$). Euclidean distance $d_{si}$ of the point from the center of the sphere is calculated using equation 1. When this distance $d_{si}$ is less than 1, the point is within the sphere, and if the distance is equal to 1, then the point is on the surface of the sphere. Such a point$(x_i, y_i, z_i)$ with distance $d_{si}$ less than or equal to 1 is selected.

$$d_{si} = \sqrt{\frac{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2}{r^2}} \tag{1}$$

Similarly, a point $(x_i, y_i, z_i)$ from the input point set $xyz_1$ within the volume of the ellipsoid with axis-lengths $a$, $b$, $c$ and center $(x_j, y_j, z_j)$ (from the farthest point set $xyz_2$) is selected by calculating the distance $d_{ei}$ of the point from the ellipsoid center using equation 2. All those points whose distance value $d_{ei}$ from the center of the ellipsoid is less than 1 are within the ellipsoid and equal to 1 are on the surface of the ellipsoid. Such a point with distance $d_{ei}$ less than or equal to 1 is selected.

$$d_{ei} = \sqrt{\frac{(x_i - x_j)^2}{a^2} + \frac{(y_i - y_j)^2}{b^2} + \frac{(z_i - z_j)^2}{c^2}} \tag{2}$$

The phrase "re-orientation of ellipsoid" is used, though in reality the input points are rotated around the centroid point. Using the ellipsoid formula (equation 2) these rotated points are captured if they are within the ellipsoid. The rotation matrix required to re-orient the ellipsoid in the principal direction of the local neighborhood points $\mathcal{G}_j(x, y, z)$ is obtained from the Eigenvectors of the co-variance matrix $cov(x, y, z)_j$, computed from the unique point sets $\mathcal{N}_j(x, y, z)$ obtained by ball querying the local neighborhood. After re-orientation of the ellipsoid using the rotation matrix, the neighborhood is queried again enriching the local neighborhood capture.

Choosing ellipsoid semi-axis lengths comparable with the radius of the sphere ball, we first demonstrate an improvement in overall classification accuracy of the PointNet++ single scale model. Taking the improvement in accuracy as motivation, we then propose our method to reorient the ellipsoid to capture maximum local neighborhood. We take union of local points captured from both queries. Each centroid point $(x_j, y_j, z_j)$ is used twice to query points around it. The first query is either a ball or an ellipsoid query whereas the second query is always a reoriented ellipsoid query.

Figure 3 shows the surface area of a 3D object that overlaps the ellipsoid volume is greater than or equal to the surface area overlapping the volume of the sphere. This is true when we assume both the sphere and the ellipsoid are centered at the same point $(x_j, y_j, z_j)$. Also, at least two of the axis-lengths of the ellipsoid are less than or equal to the radius of the sphere irrespective of the ellipsoid orientation. We keep this assumption of $b$=$c$=$r$ and $a$=$2r$, where $r$ is the

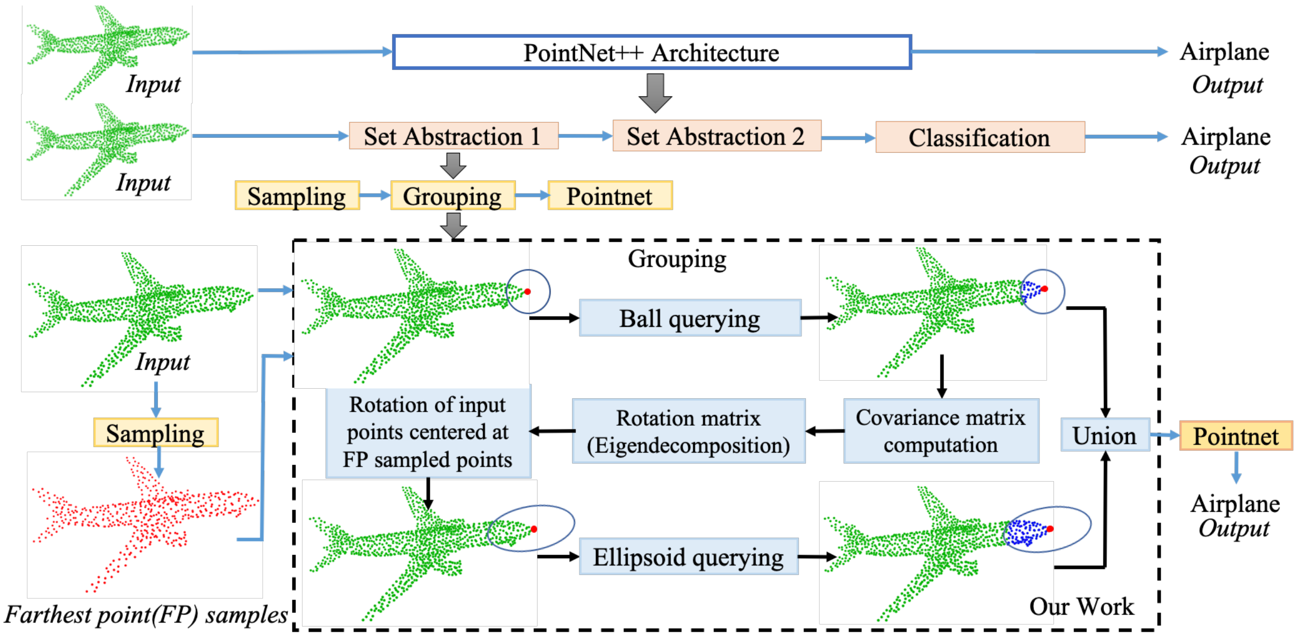


Figure 4: Ellipsoid querying for grouping in detail: Points selected by ellipsoid querying are used to create a co-variance matrix. Eigenvectors are calculated from the co-variance matrix to derive a rotation matrix. Ellipsoid is re-oriented using rotation matrix to re-capture more points. An union of points obtained by querying the local neighborhood twice is processed by pointnet. More captured points (blue color) are visible on the bottom rightmost airplane (Best viewed in color).

radius of the sphere, in defining the axis-lengths of the ellipsoid. For example as in figure 3, if the radius of the sphere is $r$ = 0.2 and the axis-lengths of the ellipsoid are $a$ = 0.4, $b$ = 0.2 and $c$ = 0.2 with the center of both the sphere and the centroid as $(x_j, y_j, z_j)$, then the neighborhood around the centroid queried by using ellipsoid querying is bigger than or as big as the neighborhood queried by using the sphere. With this understanding, we applied ellipsoid querying in the local neighborhood around the centroid and observed improvements in the accuracy.

The sphere with no specific orientation gets lesser overlap with the object to capture neighboring points around the centroid. Ellipsoid unlike a sphere has orientation that can be exploited to re-orient it in a way that maximum number of points on the object overlaps within the volume of the ellipsoid for querying. Also, for each centroid the orientation of the ellipsoid can be made different in order to maximize the neighborhood coverage with more relevant points further improving the 3D classification accuracy.

We adopted PointNet++ network as shown in figure 4 and further developed the grouping layer in the set abstraction levels. Our intuition is that the increase in the number of points by using re-oriented ellipsoid boosts the existing PointNet++ framework in learning a better representation. The sampling layer employs farthest point sampling(FPS) to sample out a fixed number of points from the original point set $(xyz_1)$. These sampled points are called the centroids $(xyz_2)$. The grouping layer uses the input point set and the centroids for ball querying (equation 1). Querying results in a subset of input 3D point set around each of the centroid point representing its local neighborhood.

Although Pointnet++ limits the number of points (samples) for querying in the local neighborhoods, unlike k-nearest neighbors (kNN) with k points queried every time, the number of points queried by ball querying is not fixed. Hence, grouping results in a range of $[1, samples]$ of points. Our work in this paper uses a re-oriented ellipsoid to increase the number of relevant points queried from the local neighborhood. For a single-scale model, each set abstraction layer queries points in two stages. In the first stage, ball querying around a centroid $(x_j, y_j, z_j)$ gives a subset of points $\mathcal{G}_j(x, y, z)$ from the input 3D point set within the ball's volume. At least three unique points are required to compute a 3x3 co-variance matrix. A co-variance matrix $cov(x, y, z)_j$ is computed for each of the centroids $(x_j, y_j, z_j)$ that have at least three unique queried points.

We also propose to adjust the co-variance matrix computation function CovarAdjMatx (algorithm 1). We used centroid itself as the mean of the unique grouped points $\mathcal{N}_j(x, y, z)$ if the Euclidean distance between the centroid$(x_j, y_j, z_j)$ and actual mean $(\bar{x}_j, \bar{y}_j, \bar{z}_j)$ of grouped points is greater than or equal to one-fourth the length of

**Algorithm 1** Compute Adjusted Co-variance Matrix

**function** COVARADJMATX($\mathcal{G}_j(x,y,z), (x_j, y_j, z_j), a$)
▷ Input $\mathcal{G}_j(x,y,z)$ , matrix of grouped points
▷ Input $x_j$,$y_j$,$z_c$ is a centroid point
▷ Input $a$ is the major axis-length of the ellipsoid
▷ Output $cov(x,y,z)_j$ is a 3x3 symmetrical matrix
$\mathcal{N}_j(x,y,z) \leftarrow unique(\mathcal{G}_j(x,y,z))$
$n_j \leftarrow count(\mathcal{N}_j(x,y,z))$
$\bar{x}_j, \bar{y}_j, \bar{z}_j \leftarrow \frac{\sum_{i=1}^{n} x_i}{n_j}, \frac{\sum_{i=1}^{n} y_i}{n_j}, \frac{\sum_{i=1}^{n} z_i}{n_j}$
$dist_i \leftarrow \sqrt{(\bar{x}_j - x_j)^2 + (\bar{y}_j - y_j)^2 + (\bar{z}_j - z_j)^2)}$
**if** $dist_i \geq a/4.0$ **then**
$\mathcal{N}(x_i, y_i, z_i) \leftarrow \mathcal{N}(x_i, y_i, z_i) - (x_j, y_j, z_j)$
**else**
$\mathcal{N}(x_i, y_i, z_i) \leftarrow \mathcal{N}(x_i, y_i, z_i) - (\bar{x}_j, \bar{y}_j, \bar{z}_j)$
**end if**
$cov(x,y,z)_j \leftarrow \frac{\mathcal{N}_|(x,y,z)^T * \mathcal{N}_|(x,y,z)}{n_j - 1}$ ▷ * is matmul
**return** $cov(x,y,z)_j$
**end function**

major axis as threshold else we used the actual mean of the grouped points. The threshold can be greater than or equal to half the radius (which is also one fourth the major-axis length). If threshold is less than half the radius then the actual mean of grouped points is closer to the centroid making the adjustment and hence the ellipsoid re-orientation less effective. The primary reason behind this adjustment is to orient the ellipsoid in the direction of grouped points $\mathcal{G}_j(x,y,z)$, mainly when all grouped points occur on one side of the centroid. E.g., in the left part of figure 1, all grouped points from the ball query (blue points) are to the left of the centroid (red point). At this stage, each centroid has a co-variance matrix $cov(x,y,z)_j$. Eigenvalues and Eigenvectors are computed by the Eigendecomposition of the symmetric co-variance matrix using the Jacobi method [25]. Eigenvectors corresponding to the descending ordered Eigenvalues are used as the rotation matrix $R_\nu$ to re-orient the input points centered at the centroid $(x_j, y_j, z_j)$ using equation 3 where $xyz_1$ represents the input point cloud and $xyz_2$ are the centroids from the sampling layer. The second stage captures re-oriented points that are on or inside the ellipsoid.

$$\begin{bmatrix} x_i' \\ y_i' \\ z_i' \end{bmatrix} = R_\nu \begin{bmatrix} x_i - x_j \\ y_i - y_j \\ z_i - z_j \end{bmatrix} i \in xyz_1, \text{ for each } j \in xyz_2 \tag{3}$$

Each centroid goes through the above-explained steps (algorithm 2) i.e., ball/ellipsoid querying, co-variance matrix computation, Eigendecomposition, rotation of input points around the centroid, and capturing rotated points within the ellipsoid. These newly captured points are additional points to the first query points by taking a union set that has all of the first ball/ellipsoid query points and few or all of the re-oriented ellipsoid query points. If the total number of points in the union is less than or equal to the samples requested in querying, then union set itself is the new query points. However, if the total number of points in the union is more than the samples requested, then the difference in the number of samples and the number of ball query points are added into the union from the re-oriented ellipsoid. These enriched grouped points have a better representation of the local neighborhood. Pointnet layer encodes these grouped points into local feature vectors, used in classification.

**Algorithm 2** Re-oriented Ellipsoid Querying

**function** ELLIPSOIDQRY($(a,b,c), xyz_1, xyz_2, samples$)
▷ Input: ellipsoid semi-axis lengths (a,b,c), input point set ($xyz_1$), Farthest Point Sampled points($xyz_2$), $samples$
▷ Output: $idx, pcount$
$idx, pcount \leftarrow query(a,b,c,xyz_1,xyz_2,samples)$
$G \leftarrow group\_point(xyz_1, idx)$
**for** each centroid $c_j$ in $xyz_2$ **do**
**if** $pcount_i \geq 3$ **then**
$cov(x,y,z)_j \leftarrow$ COVARADJMATX($G_j$)
$\lambda$,$R_\nu \leftarrow Eigendecompose(cov(x,y,z)_j)$
**for** each $p_i$ in $xyz_1$: **do**
$p_i' = R_\nu * (p_i - c_j)^T$ ▷ * is matmul
$d_e \leftarrow \sqrt{\frac{p_{xi}'}{a^2} + \frac{p_{yi}'}{b^2} + \frac{p_{zi}'}{c^2}}$
**if** $d_e \leq 1.0$ & $pcount < samples$ **then**
$idx_c \leftarrow idx_c \cup idx(p_i)$
$pcount \leftarrow pcount + 1$
**end if**
**end for**
**end if**
**end for**
**return** $idx, pcount$
**end function**

## 4. Experiments

We evaluated the task of 3D point set classification on the ModelNet40 benchmark dataset [26]. This data set has 40 categories of objects comprising of a total of 12,311 shapes. The officially split of the benchmark dataset is 9843 objects for training and 2468 objects for testing, respectively. We have retained this split as is in training and testing our model (PointNet++ model with re-oriented ellipsoid querying) for a fair comparison with PointNet++[2] and other deep learning models such as Relation-Shape CNN [22]. These models uniformly sample 1,024 points from the 3D object into a point cloud, i.e., ($x$,$y$,$z$) coordinates from the object's mesh faces. Mesh faces are discarded after sampling. The point cloud is then re-scaled to fit into the unit sphere.

As a pre-requisite step, we reproduced the 3D classification results of PointNet++ for single and multi-scale models. Following are the alternative techniques to ball querying that we propose for the grouping layer of PointNet++.

- Firstly, use a fixed orientation ellipsoid for querying in place of a sphere.
- Secondly, use a ball query-based re-oriented ellipsoid querying. Then use re-oriented ellipsoid query points for further processing, and discard ball query points.
- Thirdly, use a combination of a fixed orientation ellipsoid (points discarded after computing rotation matrix) with a re-oriented ellipsoid (similar to the second technique processed further).
- Fourth technique is similar to the second technique, except the ball queried points considered as a primary set of points, and points queried by the re-oriented ellipsoid as additional points. i.e., perform a union of the ball and re-oriented ellipsoid query points limited by the number of samples of the set abstraction layer.
- Fifth technique is similar to the fourth but use fixed orientation ellipsoid for the first querying, and the re-oriented ellipsoid query points considered as additional points. All of the techniques except the first involve the Eigendecomposition of the first query points.
- Sixth and Seventh techniques are similar to fourth and fifth but with Eigenvalues as additional features.

The primary challenge in using ellipsoid querying for a given model that previously used ball query with radius $r$ is in defining the axis-lengths ($a$,$b$,$c$) of the ellipsoid. We simplified the problem in case of single-scale models by considering two of the three axis-lengths of the ellipsoid to be equal to the radius of the ball (i.e., $b$=$c$=$r$). We followed the technique of doubling the radius in PointNet++, and used multiples of $a$, $b$, $c$ from first set abstraction in the second set abstraction. We used PointNet++ with a single scale grouping model to accommodate the ellipsoid querying technique with ellipsoid axis-lengths as parameters, replacing the radius $r$ from the ball querying method. Although, the architecture for single scale grouping has three set abstraction layers, only the first two support sampling, and grouping.

Table 1 is our adaptation of the singe scale grouping model in PointNet++. We have used the ellipsoid semi-axis lengths $b$=$c$=$r$ and $a$=2$r$ where $r$ is the radius used for ball querying in PointNet++ while we doubled the sample sizes in the set abstraction step. We also observed that doubling the radius of the sphere (equal to major-axis of ellipsoid) and doubling the sample size in the set abstraction layers of PointNet++ single-scale model would not improve over the classification accuracy of 90.7%.

| SA | FPS | semi-axis[$a$,$b$,$c$] | sample | MLPs |
|---|---|---|---|---|
| 1 | 512 | [0.4, 0.2, 0.2] | 64 | [64,64,128] |
| 2 | 128 | [0.8, 0.4, 0.4] | 128 | [128,128,256] |

Table 1: PointNet++ model with two set abstraction (SA) layers for single-scale grouping with re-oriented ellipsoid.

Similarly, Relation-Shape CNN architecture has been modified in the single scale grouping sections to accommodate ellipsoid querying with ellipsoid semi-axis lengths as parameters, replacing the radius $r$ from the ball querying method. The architecture for single scale grouping (SSG) has three set abstraction (SA) layers, with the first two supporting sampling and grouping as described in table 2. For Ellipsoid Querying in the single scale grouping of PointNet++, we have used a sample size twice the sample size of the original PointNet++ SSG model, while we used the same sample sizes in case of RS-CNN SSG model.

| SA | FPS | semi-axis[$a$,$b$,$c$] | sample | MLPs |
|---|---|---|---|---|
| 1 | 512 | [0.25, 0.15, 0.15] | 48 | [input$_c$, 128] |
| 2 | 128 | [0.5, 0.3, 0.3] | 64 | [128,512] |

Table 2: RS-CNN model with two set abstraction layers for single-scale grouping with re-oriented ellipsoid.

During the training procedure, we retain the data augmentation techniques of PointNet++ and RS-CNN in our experiments involving the respective model, which are random scaling of objects, perturbing the object and point locations. Size of input point cloud is 1024 points. A 50% dropout rate is applied to fully connected layers. Both models use adam optimizer. During the testing phase, PointNet++ uses a majority voting of 12 votes while RS-CNN uses 10 votes. The source code is available at `https://github.com/VimsLab/EllipsoidQuery`.

## 5. Evaluation and Results

Table 3 gives an overview of the comparison of our results with recent deep neural networks for classification of 3D point sets on ModelNet40, a benchmark dataset. Our implementation of querying in PointNet++ with the union of points from ellipsoid querying followed by re-oriented ellipsoid querying with the concatenation of Eigenvalues as additional features (i.e., seventh technique) performs as good or even better than few of the models such as [1, 2, 10, 18, 24, 27, 28, 29, 30, 31, 32, 33, 34, 35]. Primarily, we borrowed the PointNet++ model architecture to show how ellipsoid querying has improved 3D classification results. A significant 1.1% increase in the accuracy achieved for PointNet++ single scale grouping model. Figure 5 shows improvements in 3D classification results with the seven techniques described in the experiments section.

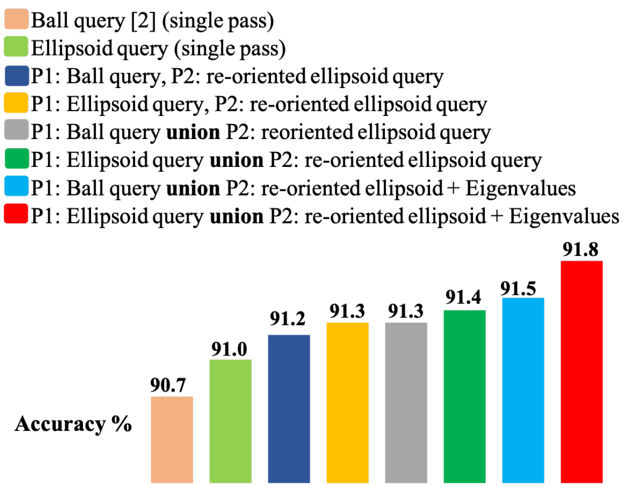


Figure 5: Improvements in accuracy of PointNet++ (SSG model). Two passes, P1: pass 1, and P2: pass 2.

We expected similar improvements for the deep learning models which use ball querying in the grouping layer and chose the state-of-the-art model Relation-Shape CNN [22] to test. We observed a 0.8% improvement over the 92.7% accuracy of RS-CNN for a single scale model, as shown in figure 6. Ellipsoid based re-oriented ellipsoid querying implementation of single scale grouping RS-CNN model outperforms recent SSG models such as [21, 23]. For RS-CNN, the accuracy did not improve by concatenation of Eigenvalues as additional features. For multi-scale models, the task of finding axis-lengths for up to six ellipsoids is challenging and remains open mainly because the number of combinations of ellipsoid dimensions for different scales are more and searching for the right combination requires a significant number of trials, which is time and resource expensive.

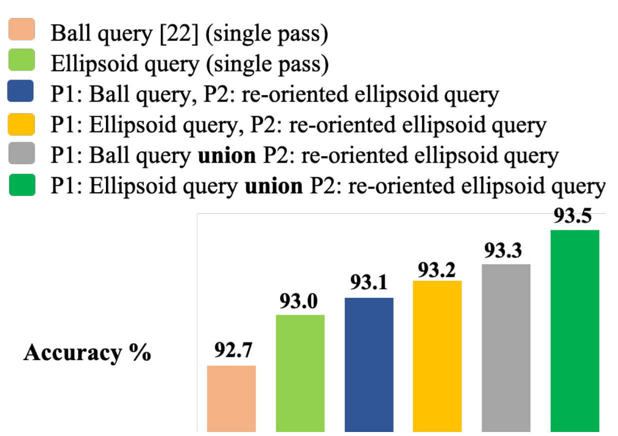


Figure 6: Improvements in accuracy of Relation-Shape CNN (SSG model). Two passes, P1: pass 1, and P2: pass 2.

Table 3: Few Recent Deep learning models with accuracies on ModelNet40 dataset for 3D Object classification. ✓indicates SSG model with ball querying.

| **Method** | **SSG** | **Acc** |
|---|---|---|
| PointwiseCNN [32] | ✓ | 86.1 |
| Deep Sets [29] | - | 87.1 |
| PointNet (vanilla) [1] | - | 87.2 |
| PointNet [1] | - | 89.2 |
| MO-Net [28] | - | 89.3 |
| 3D Capsule Networks [34] | - | 89.3 |
| JustLookUp[30] | - | 89.5 |
| Kd-Net(depth 10) [18] | - | 90.6 |
| PointNet++ [2] | ✓ | 90.7 |
| MC Convolution [35] | - | 90.9 |
| KCNet [10] | - | 91.0 |
| SFCNN [31] | - | 91.4 |
| SRN-PointNet++ [24] | ✓ | 91.5 |
| PAT [27] | - | 91.7 |
| ConvPoint [33] | - | 91.8 |
| Kd-Net(depth 15) [18] | - | 91.8 |
| DGCNN [20] | - | 92.2 |
| PCNN [15] | - | 92.3 |
| PointWeb [19] | - | 92.3 |
| Relation-ShapeCNN [22] | ✓ | 92.7 |
| KPConv(rigid) [36] | - | 92.9 |
| InterpCNN [37] | - | 93.0 |
| DensePoint [23] | ✓ | 93.2 |
| GS-Net [21] | ✓ | 93.3 |
| Geo-CNN [38] | ✓ | 93.4 |
| **Ours** | | |
| Ellipsoid PointNet++ | ✓ | 91.0 |
| Ellipsoid RSCNN | ✓ | 93.0 |
| Re-oriented Ellipsoid PointNet++ | ✓ | 91.8 |
| Re-oriented Ellipsoid RSCNN | ✓ | **93.5** |

## 6. Conclusion

In this work, ellipsoid querying, a novel approach in capturing local geometry of 3D point cloud objects, has been proposed. We also introduced seven techniques that involve ellipsoid querying for better coverage of the local neighborhood. We have demonstrated that neighborhood querying using the re-oriented ellipsoid gives better coverage of local neighborhoods in the grouping layer of PointNet++ model than ball querying. Our re-oriented ellipsoid querying also improved the classification results of the Relation-Shape CNN model with single-scale grouping. With a boost in the 3D classification accuracy for PointNet++ by **1.1%** and Relation-Shape CNN by **0.8%**, our ellipsoid-based re-oriented ellipsoid querying method is promising in improving the 3D point cloud classification results of models that use ball querying.

## References


[1] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 652–660, 2017.

[2] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in neural information processing systems*, pages 5099–5108, 2017.

[3] Fausto Bernardini, Joshua Mittleman, Holly Rushmeier, Cláudio Silva, and Gabriel Taubin. The ball-pivoting algorithm for surface reconstruction. *IEEE transactions on visualization and computer graphics*, 5(4):349–359, 1999.

[4] Charles R Qi, Hao Su, Matthias Nießner, Angela Dai, Mengyuan Yan, and Leonidas J Guibas. Volumetric and multi-view cnns for object classification on 3d data. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5648–5656, 2016.

[5] Gernot Riegler, Ali Osman Ulusoy, and Andreas Geiger. Octnet: Learning deep 3d representations at high resolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3577–3586, 2017.

[6] Saurabh Gupta, Ross Girshick, Pablo Arbeláez, and Jitendra Malik. Learning rich features from rgb-d images for object detection and segmentation. In *European conference on computer vision*, pages 345–360. Springer, 2014.

[7] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

[8] Siamak Ravanbakhsh, Jeff Schneider, and Barnabas Poczos. Deep learning with sets and point clouds. *arXiv preprint arXiv:1611.04500*, 2016.

[9] Paul Guerrero, Yanir Kleiman, Maks Ovsjanikov, and Niloy J Mitra. Pcpnet learning local shape properties from raw point clouds. In *Computer Graphics Forum*, volume 37, pages 75–85. Wiley Online Library, 2018.

[10] Yiru Shen, Chen Feng, Yaoqing Yang, and Dong Tian. Mining point cloud local structures by kernel correlation and graph pooling. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4548–4557, 2018.

[11] Truc Le and Ye Duan. Pointgrid: A deep network for 3d shape understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 9204–9214, 2018.

[12] Michael M Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42, 2017.

[13] Yangyan Li, Rui Bu, Mingchao Sun, Wei Wu, Xinhan Di, and Baoquan Chen. Pointcnn: Convolution on x-transformed points. In *Advances in Neural Information Processing Systems*, pages 820–830, 2018.

[14] Hang Su, Varun Jampani, Deqing Sun, Subhransu Maji, Evangelos Kalogerakis, Ming-Hsuan Yang, and Jan Kautz. Splatnet: Sparse lattice networks for point cloud processing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2530–2539, 2018.

[15] Matan Atzmon, Haggai Maron, and Yaron Lipman. Point convolutional neural networks by extension operators. *arXiv preprint arXiv:1803.10091*, 2018.

[16] Shenlong Wang, Simon Suo, Wei-Chiu Ma, Andrei Pokrovsky, and Raquel Urtasun. Deep parametric continuous convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2589–2597, 2018.

[17] Wenxuan Wu, Zhongang Qi, and Li Fuxin. Pointconv: Deep convolutional networks on 3d point clouds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9621–9630, 2019.

[18] Roman Klokov and Victor Lempitsky. Escape from cells: Deep kd-networks for the recognition of 3d point cloud models. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 863–872, 2017.

[19] Hengshuang Zhao, Li Jiang, Chi-Wing Fu, and Jiaya Jia. Pointweb: Enhancing local neighborhood features for point cloud processing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5565–5573, 2019.

[20] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *arXiv preprint arXiv:1801.07829*, 2018.

[21] Mingye Xu, Zhipeng Zhou, and Yu Qiao. Geometry sharing network for 3d point cloud classification and segmentation. *arXiv preprint arXiv:1912.10644*, 2019.

[22] Yongcheng Liu, Bin Fan, Shiming Xiang, and Chunhong Pan. Relation-shape convolutional neural network for point cloud analysis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8895–8904, 2019.

[23] Bin Fan Gaofeng Meng Jiwen Lu Shiming Xiang Liu, Yongcheng and Chunhong Pan. Densepoint: Learning densely contextual representation for efficient point cloud processing. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5239–5248, 2019.

[24] Yueqi Duan, Yu Zheng, Jiwen Lu, Jie Zhou, and Qi Tian. Structural relational reasoning of point clouds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 949–958, 2019.

[25] Gene H Golub and Charles F Van Loan. *Matrix computations*, volume 3. JHU press, 2012.

[26] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1912–1920, 2015.

[27] Jiancheng Yang, Qiang Zhang, Bingbing Ni, Linguo Li, Jinxian Liu, Mengdie Zhou, and Qi Tian. Modeling point clouds with self-attention and gumbel subset sampling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3323–3332, 2019.

[28] Mor Joseph-Rivlin, Alon Zvirin, and Ron Kimmel. Momen (e) t: Flavor the moments in learning to classify shapes. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 0–0, 2019.

[29] Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Russ R Salakhutdinov, and Alexander J Smola. Deep sets. In *Advances in neural information processing systems*, pages 3391–3401, 2017.

[30] Hongxin Lin, Zelin Xiao, Yang Tan, Hongyang Chao, and Shengyong Ding. Justlookup: One millisecond deep feature extraction for point clouds by lookup tables. In *2019 IEEE International Conference on Multimedia and Expo (ICME)*, pages 326–331. IEEE, 2019.

[31] Yongming Rao, Jiwen Lu, and Jie Zhou. Spherical fractal convolutional neural networks for point cloud recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 452–460, 2019.

[32] Binh-Son Hua, Minh-Khoi Tran, and Sai-Kit Yeung. Pointwise convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 984–993, 2018.

[33] Alexandre Boulch. Convpoint: continuous convolutions for point cloud processing. *Computers & Graphics*, 2020.

[34] Yongheng Zhao, Tolga Birdal, Haowen Deng, and Federico Tombari. 3d point capsule networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1009–1018, 2019.

[35] Pedro Hermosilla, Tobias Ritschel, Pere-Pau Vázquez, Àlvar Vinacua, and Timo Ropinski. Monte carlo convolution for learning on non-uniformly sampled point clouds. *ACM Transactions on Graphics (TOG)*, 37(6):1–12, 2018.

[36] Hugues Thomas, Charles R Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J Guibas. Kpconv: Flexible and deformable convolution for point clouds. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 6411–6420, 2019.

[37] Jiageng Mao, Xiaogang Wang, and Hongsheng Li. Interpolated convolutional networks for 3d point cloud understanding. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1578–1587, 2019.

[38] Shiyi Lan, Ruichi Yu, Gang Yu, and Larry S Davis. Modeling local geometric structure of 3d point clouds using geo-cnn. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 998–1008, 2019.