# Competitive Learning for Unsupervised Anomaly Detection in Intelligent Transportation Systems

Umuralp Kaytaz*, Fikret Sivrikaya* and Sahin Albayrak*†
GT-ARC gGmbH, Berlin, Germany*
DAI-Labor, Technical University of Berlin, Berlin, Germany †
{umuralp.kaytaz@gt-arc.com, fikret.sivrikaya@gt-arc.com, sahin.albayrak@tu-berlin.de}

***Abstract*—Intelligent Transportation Systems (ITSs) are expected to have a profound impact on the quality of experience in future smart cities. Anomaly detection is an imperative for urban ITS applications to alleviate vulnerabilities that may cause accidents and fatal causalities. Previously proposed anomaly detection methods mostly require prior knowledge and domain specific training and/or optimization procedures. Therefore, in this work, we propose Competitive Learning based Anomaly Detection (CLAD) as a generic end-to-end approach for unsupervised anomaly detection using Auto Regressive Integrated Moving Average (ARIMA) forecasting model, data imaging and Centroid Neural Networks (CentNNs). Utilizing multi-dimensional time-series data obtained from diverse sensory measurements in the DIGINET-PS smart city infrastructure of TU Berlin, we compare performance of CLAD with unsupervised competitive learning as well as deep learning based anomaly detection techniques. Experimental results show that proposed approach results in higher detection accuracy and precision compared to other methods when multiple degrees of anomalies are considered.**

***Index Terms*—Anomaly Detection, Centroid Neural Network, Intelligent Transportation System, Unsupervised Competitive Learning, Smart City Infrastructure.**

## I. Introduction

Unprecedented scale of urbanization creates new challenges for the digitalizing communities around the world. Expansion of transportation network loads continues to aggravate energy consumption, traffic congestion and environmental pollution, requiring a transition from previously adopted methods to a novel connected and automated transportation paradigm [1]. Considering the importance of safety in vehicular domain, management of future urban transportation ecosystem demands seamless connectivity, ubiquitous resources and reliable interdependent operations [2].

Intelligent Transportation System (ITS) is a key enabler of important vehicular applications such as platooning, parking lot management, dynamic traffic light synchronisation and real-time safety notifications [3]. Performance of these applications depends on the online learning and processing capabilities of deployed ITS infrastructure elements such as sensors, cameras, LiDARs, On-Board Units (OBUs), Road-Side Units (RSUs) as well as standalone (SA) & non-standalone (NSA) base stations [4]. Continuous interaction between these pervasive components leads to multi-dimensional reciprocal data streams that can contain point-wise or contextual anomalies [5]. Resultingly, processing of anomalous data may cause middleware malfunctions and ADAS faults that can induce head-on collisions or unexpected events. Therefore, ensuring functional stability with anomaly detection is of vital importance in transportation management for alleviating vulnerabilities that can result in accidents and fatal causalities [6]. Although reliability in transportation ecosystem has attracted considerable attention in the past, generic unsupervised anomaly detection in ITS environment is a major remaining challenge.

An anomaly detection approach based on feature extraction and classification using closed-circuit television camera images is presented in [7]. Another work for clustering video streams of traffic surveillance systems is proposed in [8]. These works focus solely on video streams lacking the necessary complexity for the digitalized ITS environment. [9] proposes a Deep Neural Network (DNN) based anomaly detection framework that utilizes Long-Short Term Memory (LSTM) technique. Similarly, a supervised learning method using Streaming Half-Space-Trees (HS-Trees) is presented in [10] for fast anomaly detection in data streams. Moreover, both of the Deep Learning (DL) approaches presented in [11] and [12] rely on Convolutional Neural Networks (CNNs), *Auto-Regressive Integrated Moving Average* (ARIMA) model forecasting and *euclidean* distance based anomaly detection. Another DL method, Generative Adversarial Network (GAN), based anomaly detection and localization in multivariate time series data is presented in [13]. All of the proposed DL methods depend on supervised/semi-supervised training procedures that require domain specific labeling of individual data points. Furthermore, their generalizability is also restricted by the use of simple distance functions at the anomaly detection stage. On the other hand, unsupervised subsequence anomaly detection with low-dimensional embedding and graph transformation is examined in [14]. However, proposed algorithm considers uni-dimensional time series and requires a predetermined subsequence length input for the anomaly detection procedure, thus inhibits applicability and practicality in vehicular domain.

In this paper, we propose Competitive Learning based Anomaly Detection (CLAD) as a generic end-to-end approach for unsupervised anomaly detection in multi-dimensional data streams of ITS applications. First, we use real-world sensory measurements obtained in the DIGINET-PS smart city infrastructure of TU Berlin for training and test-

ing datasets. Second, we train and utilize statistical ARIMA model for forecasting expected values and comparing predictions with the actual observations. Third, we perform spatio-temporal imaging of data points and apply unsupervised competitive learning based clustering using Centroid Neural Networks (CentNNs). Finally, modeling graph representation of centroid clusters as abstract states, we detect and score anomalies in multi-dimensional sensor data observations.

The remainder of the paper is organized as follows. Section II describes the system model. Section III presents preliminary information on time-series forecasting, data imaging, unsupervised competitive learning, CentNNs and graph based anomaly detection. Details of CLAD algorithm is provided in Section IV. Section V details simulation setup and analyzes experimental outcomes. Finally, we present our concluding remarks in Section VI.

## II. System Model

Adopted functional organization of the DIGINET-PS platform is provided in Fig. 1. Considering the requirements of future ITS ecosystems, DIGINET-PS encompasses sensor technology implementations, communication infrastructure deployment and intelligent coordination among road-side elements such as traffic lights, street lighting and parking slots [15]. Urban test field of the smart city platform stretches over 3.5 kms at the center of Berlin, Germany, offering a digitalized infrastructure for connected and automated vehicular applications. Road-side infrastructure of the test field consists of diverse sensors and actuators as well as RSUs, cameras and traffic lights that continuously interact with active traffic participants such as vehicles and pedestrians. On the other hand, wired and wireless networking infrastructure is facilitated for providing vehicle-to-infrastructure (V2I) and vehicle-to-vehicle (V2V) communication links.

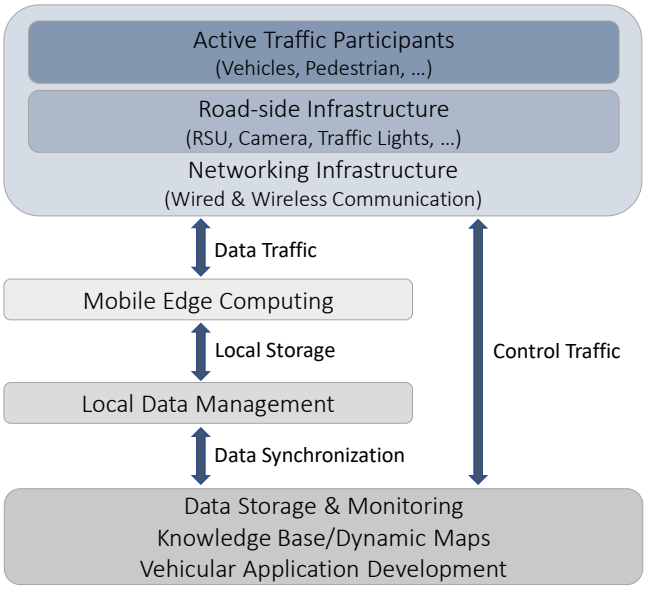


Figure 1: Organization of DIGINET-PS smart city platform

Using the communication infrastructure, data traffic containing visual recordings and sensory measurements are continuously transmitted to the Mobile Edge Computing (MEC) units for initial processing purposes. Subsequently, local data storage is performed for ad hoc computations and temporary caching. At the end of the initial processing phase, locally accumulated data is synchronized with the rest of the measurements and infrastructure resources in the global storage and monitoring system. Maintenance of the acquired data and monitoring of devices are performed for knowledge base/dynamic map utilization and further development of intelligent applications in vehicular domain [16].

## III. Preliminaries

In this section, we introduce preliminary information on ARIMA statistical model, imaging of multi-dimensional time series, unsupervised competitive learning, CentNN and anomaly scoring with graph representation. Initially, time-series forecasting using past values with ARIMA model is explained. Later on, data imaging with distance matrices is presented for data transformation and representation in learning algorithms. Followingly, competitive learning technique CentNN is discussed for unsupervised centroid based clustering. Finally, we detail graph representation for data clusters and anomaly scoring.

### A. ARIMA Forecasting Model

As a well-known time-series regression method, ARIMA is a statistical model that uses linear combination of previous errors and observations. Given a time-series data of length $l$, $X = \{x_1, ..., x_l\}$ , an *Auto-Regressive* model $AR_{(p)}$ is a process that forecasts a series based on only past $p$ values, *lags*, such as

$$x_t = \zeta + \sum_{i=1}^{p} \alpha_i x_{t-i} + \theta_t \tag{1}$$

where $\theta_t$ denotes stochastic error at time $t$, $\zeta$ is the intercept and $\alpha_i$ is a linear coefficient of past values. Similarly, a *Moving Average* model $MA_{(q)}$ can be established by replacing past values with error terms resulting in a process consisting of past $q$ errors. Using $\beta_j$ as the linear operator, $MA_{(q)}$ model can be written as

$$x_t = \zeta + \sum_{j=1}^{q} \beta_j \varepsilon_{t-j} + \theta_t \tag{2}$$

In order to utilize information from both errors as well as observations, *ARIMA* model blends ideas of *AR* and *MA* models as a linear combination and includes a parameter $d$ in the prediction as the number of first order differences for transforming non-stationary time-series [17]. Consequently, $ARIMA_{(p,q,d)}$ forecasting model can be written as follows

$$z_k = \Delta^d x_k \tag{3}$$

$$z_t = \omega + \sum_{i=1}^{p} \alpha_i z_{t-i} + \sum_{j=1}^{q} \beta_j \varepsilon_{t-j} + \theta_t \tag{4}$$

### B. Data Imaging for Multi-Dimensional Time Series

Data imaging using distance matrices is an effective way of transforming multi-variate time series and revealing inter-correlations between different data sources [13]. Given $d$ dimensional data points $x_i \in \mathbb{R}^d$ in a time series $X = \{x_i, ..., x_l\}$, such that $X \in \mathbb{R}^{dxl}$, a distance matrix $e_i \in \mathbb{R}^{dxd}$ from vector $x_i = \{x_{i_1}, ..., x_{i_d}\}$ for a time period $\lambda$ can be calculated using *Minkowski* distance function as follows

$$e_{i_{j,k}} = \frac{1}{\lambda} \sum_{\delta=0}^{\lambda-1} \left( \left| x_{j_{i-\delta}} - x_{k_{i-\delta}} \right|^M \right)^{\frac{1}{M}} \tag{5}$$

where $M = 2$ can be used for *euclidean* distance based calculations. This transformation to data images $\{e_1, .., e_l\}$ enables encoding spatio-temporal information of data sources as a pairwise phase difference while providing robustness against impulse noise with time period averaging [13].

### C. Unsupervised Competitive Learning & CentNNs

Stemming from the ideas of *Hebbian Theory* and *synaptic plasticity* [18], in unsupervised competitive learning, a set of neurons $M$ compete with each other in a *"winner-takes-all"* manner by measuring similarity to a particular input vector $x_i = \{x_{i_1}, ..., x_{i_d}\}$. At a given *epoch*, namely a complete presentation of all the input data vectors, a neuron wins an input data vector if it has the most similarity compared to other neurons. This similarity condition for winning a data vector $x_i$ can be realized using *L2-norm* based nearest neighbour selection with $w_i = \{w_{i_1}, ..., w_{i_d}\}$ as the winning neuron weights

$$w_i = \min_k ||x_i - w_k||^2 \tag{6}$$

As an unsupervised competitive learning method, CentNN uses neurons as locally optimal synaptic vectors that iteratively converge to centroids of data clusters. CentNN algorithm updates neuron weights based on *minimum energy* and *status change* conditions comparing recent and previous *epochs* [19]. Considering a cluster $c$ consisting of $N^c$ members, based on the *minimum energy* condition, weights of the centroid $w_c$ in a data cluster should be chosen in a way to contain *minimum energy* such as

$$w_c = \min_j \sum_{k=1}^{N^c} ||x_k^c - w_j||^2 = \frac{1}{N^c} \sum_{k=1}^{N^c} x_k^c \tag{7}$$

where $x_k^c$ represents *k-th* data vector in cluster $c$. In CentNNs, a neuron's status becomes *winner* if it wins input data vector in current *epoch* but it did not win the same vector in previous *epoch* iteration. On the other hand, a neuron becomes *loser* if it won the given data vector in previous *epoch* but does not win the vector in the most recent *epoch* iteration [19]. Based on this, neuron weight update equations of CentNN algorithm for *winner* $i$ and *loser* $j$ neurons can be written as follows

$$\begin{aligned} w_i(n+1) &= \frac{1}{N^i - 1}[w_i(n)N^i - x_n] \\ &= w_i(n) - \frac{1}{N^i - 1}[x_n - w_i(n)] \end{aligned} \tag{8}$$

$$\begin{aligned} w_j(n+1) &= \frac{1}{N^j + 1}[w_j(n)N^j + x_n] \\ &= w_j(n) + \frac{1}{N^j + 1}[x_n - w_j(n)] \end{aligned} \tag{9}$$

where $n$ stands for the iteration index, $w(n)$ represents weight vector at the $n$-th iteration and $N^c$ is the number of members in cluster $c$.

Table I: Notation

| Notation | Description |
|---|---|
| $X = \{x_1, ..., x_l\}$ | Time-series data of length l |
| $x_i = \{x_{i_1}, ..., x_{i_d}\}$ | $d$-dimensional vector $x$ at index $i$ |
| $AR_{(p)}$ | *Auto-Regressive* model with past $p$ values |
| $MA_{(q)}$ | *Moving Average* model with past $q$ errors |
| $ARIMA_{(p,q,d)}$ | *Auto-Regressive Integrated Moving Average* model |
| $\zeta$ | Intercept |
| $\alpha, \beta$ | Linear coefficients |
| $\theta_i$ | Stochastic error at index $i$ |
| $z_k = \Delta^d x_k$ | $d$-times first-order difference of $x_k$ |
| $e_i$ | Data image at index $i$ |
| $w_l = \{w_{l_1}, ..., w_{l_d}\}$ | Weight vector of $l$-th neuron |
| $w^c$ | Weights of centroid in cluster $c$ |
| $w(n)$ | Weight vector at the n-th iteration |
| $N^c$ | Number of members in cluster $c$ |
| $x_k^c$ | *k-th* data vector in cluster $c$ |
| $\mathcal{G}_T(\mathcal{N}, \mathcal{E})$ | Cyclic graph of data clusters for time period $T$ |
| $X^o = \{x_1^o, ..., x_T^o\}$ | *Observed* time-series in time period $T$ |
| $X^p = \{x_1^p, ..., x_T^p\}$ | *Predicted* time-series in time period $T$ |
| $p_i^o \in e_i^o$ | *Observed* data image point at index $i$ |
| $p_i^p \in e_i^p$ | *Predicted* data image point at index $i$ |
| $C^a$ | Cluster $a$ |
| $A^k$ | Anomaly of $k$-th degree |

### D. Graph based Anomaly Scoring

Graph representation of low-dimensional time series transformations has been shown to be an effective method for detecting anomalies of subsequence elements in an unsupervised manner [14]. Inspired by this approach, we define $\mathcal{G}(\mathcal{N}, \mathcal{E})$ as a cyclic *graph*, where *node set* $\mathcal{N}$ is the set of clusters learned from data images $\{e_1^o, e_1^p, ..., e_T^o, e_T^p\}$ of *observed* $X^o = \{x_1^o, ..., x_T^o\}$ and *predicted* $X^p = \{x_1^p, ..., x_T^p\}$ multi-dimensional time series over a time period $T$ and *edge set* $\mathcal{E}$ represents bidirectional edges between neighbouring centroids. Considering a *graph* $\mathcal{G}(\mathcal{N}, \mathcal{E})$ and a centroid $C^a \in \mathcal{N}$, we define *normality* $(A^0)$ as both *observation* $p_i^o \in e_i^o$ and *prediction* $p_i^p \in e_i^p$ data image points from a given time index $i$ belonging to the same cluster, namely $\{p_i^o, p_i^p\} \in C^a$. Similarly, degree of anomaly $A^k$ for data image points $\{p_j^o, p_j^p\}$ from a time instance $j$ can be calculated based on the number of one-hop cluster neighbours $k$ between the *observed* $\{p_j^o\}$ and *predicted* $\{p_j^p\}$ points as shown in Fig. 2.

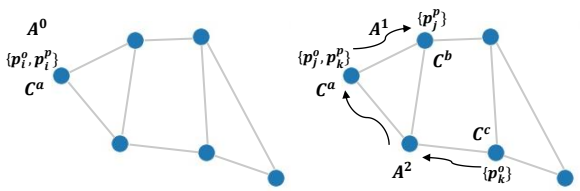


Figure 2: Graph based anomaly scoring

## IV. Approach

Next, we describe our CLAD architecture and algorithms for prediction and unsupervised anomaly detection in ITS data streams. Initially, we elaborate on CLAD algorithm by detailing prediction using *observed* data, imaging of both *observed* and *predicted* time-series, clustering with CentNN approach and graph based anomaly detection. Later on, we further explain data image clustering using unsupervised competitive learning with multiple neurons that converge to cluster centroids.

**Algorithm 1:** Competitive Learning based Anomaly Detection (CLAD)

**Input:** $x_i^o \in X^o : x_i^o = \{x_{i_1}^o, ..., x_{i_d}^o\} \leftarrow$ *observed* sensor data stream from ITS infrastructure
**Output:** $\mathcal{A} \leftarrow$ detected anomalies
1 **Compute** $X^p \leftarrow ARIMA_{(p,q,d)}(X^o)$
2 **Initialize** global data image $e^G$
3 **Initialize** set of anomaly scores $\mathcal{S}$
4 **for** *each data point* $\{x_i^o \in X^o\}$ *and* $\{x_i^p \in X^p\}$ **do**
5 **Compute** $e_i^o \leftarrow data\_imaging(x_i^o)$
6 **Compute** $e_i^p \leftarrow data\_imaging(x_i^p)$
7 **Update** $e^G \leftarrow get\_points(e_i^o, e_i^p)$
8 **Compute** centroids $\{\mathcal{C}\} \leftarrow CNNC(e^G)$
9 **Get** graph $\mathcal{G} \leftarrow construct\_graph(\{\mathcal{C}\})$
10 **for** *each data image point* $\{p_j^o \in e_j^o\}$ *and* $\{p_j^p \in e_j^p\} | \{e_j^o, e_j^p\} \in e^G$ **do**
11 **Compute** $\{A_j\} \leftarrow score\_anomaly(e^G, \mathcal{G})$
12 **Update** $\mathcal{S} \leftarrow update\_anomaly\_list(\{A_j\})$

### A. Competitive Learning based Anomaly Detection (CLAD)

Algorithm 1 is executed for detecting and scoring anomalies in multi-dimensional *observed* data streams obtained from a given ITS infrastructure. Detection procedure is triggered upon storage of multiple sensor measurements in a discretized and sequential manner. Initially, $ARIMA$ statistical forecasting model is used for predicting upcoming values of *observed* sensor measurements using predetermined parameters $\{p, q, d\}$ (Line 1). Later on, a global data image $e^G$ is initialized for accumulating data image transformations in a planar domain (Line 2). Similarly, a set $\mathcal{S}$ is also initialized for storing and returning the detected anomalies (Line 3). Data imaging procedure is iteratively carried out for each of the data points in *observed* and *predicted* time series (Lines 4-7). During this procedure, data is transformed and represented in terms of distance matrices containing multiple data image points that encode spatio-temporal correlations of diverse measurements (Lines 5-6). Followingly, global data image $e^G$ is updated using newly computed *observed* and *predicted* data image points $\{e^o, e^p\}$ (Line 7). Using unsupervised competitive learning approach CentNN, unsupervised clustering is performed on the global data image $e^G$ (Line 8). Resultingly, a set of centroid locations $\{\mathcal{C}\}$ are returned as the neurons in CentNN converge to locally optimal centroids of data clusters with the iterative weight update procedure. Considering data clusters as abstract states and neighborhood relationship of data clusters as edges, namely $\mathcal{N}$ and $\mathcal{E}$, a cyclic graph $\mathcal{G}(\mathcal{N}, \mathcal{E})$ is constructed using centroid coordinates on the global data image $e^G$ (Line 9). Consequently, each *observed* data image point $\{p_j^o \in e_j^0\}$ in the global data image $e^G$ is compared with its *predicted* counterpart $\{p_j^p \in e_j^p\}$ in terms of cluster membership and anomalous localization of data image point couples are scored (Lines 10-11). At the end of graph based anomaly detection process set $\mathcal{S}$ is updated with latest anomaly information (Line 12).

**Algorithm 2:** Centroid Neural Network based Clustering (CNNC)

**Input:** $\{p_i^o, p_i^p\} \in e^G : \forall i \in \mathbb{N}$
**Output:** $\{\mathcal{C}\} \leftarrow$ set of centroids
**Data:** $M \leftarrow$ set of neurons, $k \leftarrow$ number of clusters
1 **Compute** $c \leftarrow compute\_centroid(e^G)$
2 **Initialize** $k = 2$
3 **Initialize** $\{w_{n_1}, w_{n_2}\} \leftarrow set\_weights(c) : n_1, n_2 \in M$
4 **for** $k \leq M$ **do**
5 **for** *each point* $p \in e^G$ **do**
6 **Compute** *winner* neuron $\mathcal{W}$ of current *epoch*
7 **Compute** *loser* neuron $\mathcal{L}$ of current *epoch*
8 **Update** $\{\mathcal{W}, \mathcal{L}\} \leftarrow update\_weights(w_{\mathcal{W}}, w_{\mathcal{L}}, p)$
9 **if** $k \neq |M|$ **then**
10 **Compute** $w_{n_{k+1}} \leftarrow split\_based\_on\_error(e^G)$
11 **Update** $k+ = 1$

### B. Centroid Neural Network based Clustering (CNNC)

CLAD procedure uses Centroid Neural Network based Clustering (CNNC) algorithm for updating neuron weights and computing locally optimal centroids with unsupervised competitive learning approach. After construction of the global data image $e^G$, Algorithm 2 is run for training a predetermined number of neurons that converge to centroids of data clusters. First, an overall centroid $c$ of all available data image points is computed (Line 1). Later on, a neuron

counter variable $k$ and weights of two neurons $\{w_{n_1}, w_{n_2}\}$ are randomly initialized using $c$ and a random error $\epsilon$ such as $w_{n_{1,2}} = c \pm \epsilon$ (Lines 2-3). This initialization is performed to avoid training of CentNN from getting stuck at an undesired local optimum [19]. Followingly, a training procedure is repeated until all of the clusters are stable and neuron weights do not require to be updated (Lines 4-11). During the training phase of CentNN, weight update is performed by iterating over each data point $p \in e^G$ and determining winner $\mathcal{W}$ and loser $\mathcal{L}$ neurons at the most recent *epoch* (Lines 5-8). However, a data cluster $j$ with the most error is split unless desired number of neurons are included in the training procedure (Lines 9-10). A weight vector is initialized $w_{n_{k+1}} = w_{n_j} + \epsilon$ for a newly joining neuron using a random error $\epsilon$ and weight of neuron $j$. At the end of each epoch, neuron counter variable $k$ is updated (Line 11).

## V. Simulations

### A. Simulation Setup

Simulation results of anomaly detection methods have been obtained using open-source neural-network library Keras, python-based statistical estimation module *statsmodels* and network analysis package *NetworkX*. Experimental data have been collected by utilizing DIGINET-PS platform resources for *vehicle velocity (Velo)*, *vehicles-per-minute (Vpm)* count, *Cooperative Awareness Message (CAM)* count, *Decentralized Environmental Notification Message (DENM)* count and *lane change activity percentage (Lane)* measurements. A total of 11100 data points were accumulated with 5 minute intervals over a period of 10 days. Furthermore, $[2/3, 1/3]$ *train-test split* ratio and linear scaling based normalization were adopted for obtaining the experimental datasets. Multivariate *normal distribution* samples were added as synthetic anomalies to the %1 of measurement data for evaluation purposes.

Table II: Hyperparameters

| Hyperparameter | Value |
|---|---|
| CLAD | |
| Number of epochs | 1000 |
| ARIMA (p,q,d) | (5,1,0) |
| Number of neurons | 10 |
| Epsilon $\epsilon$ | 0.05 |
| GAN | |
| Number of epochs | 10000 |
| Generator feed-forward network layers | 15-10 |
| Discriminator feed-forward network layers | 25-10-1 |
| Optimizer | Adam |
| Batch size | 128 |
| Latent dimensions | 10 |
| Loss function | Binary cross-entropy |
| SOM | |
| Number of epochs | 10000 |
| Map shape | 2x5 |
| Learning rate | 0.01 |
| K-Means | |
| Number of epochs | 1000 |
| Number of clusters | 10 |

In order to compare performance of our proposed approach with other competitive learning based unsupervised anomaly detection methods, we have implemented Self-Organizing Map (SOM) and K-Means algorithms with default hyper-parameters presented in Table II. Distance measurement to cluster centroids was used alongside with thresholding for classification of anomalies in unsupervised anomaly detection techniques and the average of multiple threshold values was calculated for statistical performance analysis. Additionally, as a semi-supervised DL based method, GAN and *euclidean* Distance Function (DF) based anomaly detection was selected similar to the approach proposed in [13]. For a fair comparison, ARIMA model was utilized in the prediction phase of unsupervised anomaly detection techniques.

Table III: Statistical Metrics for Anomaly Detection

| Metric | Formula |
|---|---|
| Accuracy | (True Positives + True Negatives) / Sample Count |
| Precision | True Positives / (True Positives + False Positives) |
| Recall | True Positives / (True Positives + False Negatives) |

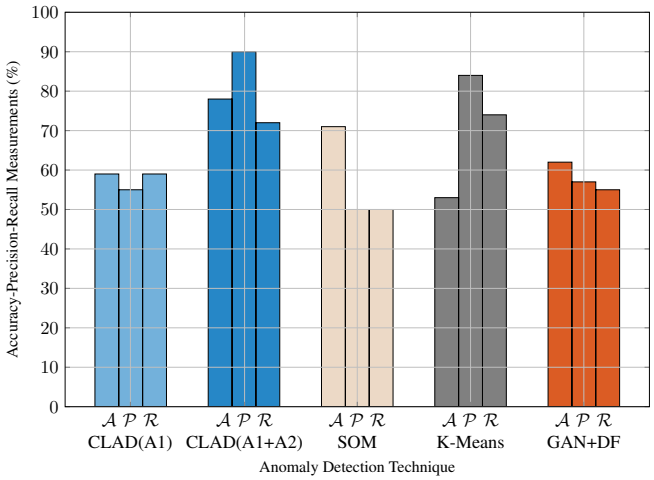


Figure 3: Accuracy($\mathcal{A}$) - Precision($\mathcal{P}$) - Recall($\mathcal{R}$) statistics of anomaly detection techniques

### B. Simulation Results

We first present evaluation of anomaly detection methods using statistical performance metrics shown in Table III. Fig. 3 shows the comparison of anomaly detection performance in terms of accuracy, precision and recall measurements. It is evident that the CLAD algorithm provides higher accuracy and precision of anomaly detection compared to other methods when both first and second degree anomalies are considered, CLAD(A1+A2). This shows that proposed model can differentiate true anomalies as well as falsely labeled anomalies (*false positives*) better than other techniques. On the other hand, low detection accuracy is observed with K-means model and CLAD procedure when only first degree anomalies are involved, CLAD(A1). However, we also see that K-means results in the highest recall values among all the selected anomaly detection algorithms.

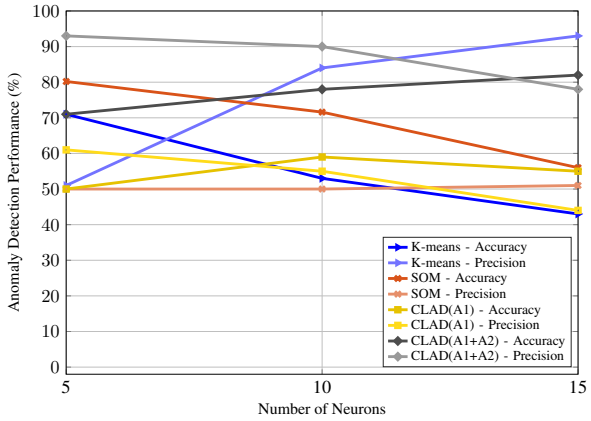


Figure 4: Performance of unsupervised competitive learning based anomaly detection with changing number of neurons

Fig. 4 depicts the performance comparison of unsupervised competitive learning based anomaly detection methods for changing number of neurons in the computation procedure. We observe higher accuracy and lower precision values when the number of neurons increases with the CLAD(A1+A2) method due to the expansion of vertex degrees and cluster count in the graph representation. Contrarily, the proposed algorithm results in lower performance measurements when only first degree anomalies are employed, CLAD(A1). As more neurons compete for data point collection in the unsupervised clustering procedure, the accuracy of traditional techniques, namely K-means and SOM, decreases drastically while their precision performance of detecting true anomalies (*true positives*) increases.

## VI. Conclusion and Future Work

In this paper, we present CLAD as a generic end-to-end approach for unsupervised anomaly detection and scoring in multi-dimensional data streams of ITS platforms. Initially, CLAD algorithm has been proposed utilizing ARIMA forecasting, data imaging, CentNN based competitive clustering and graph representation techniques. Later on, we compared performance of CLAD with unsupervised competitive learning and semi-supervised DL based anomaly detection methods while incorporating various statistical metrics. Furthermore, we have used real-world sensory measurements obtained in the DIGINET-PS smart city infrastructure of TU Berlin for the preparation of training and testing datasets in our experiments. We see that CLAD demonstrates higher accuracy and precision performance in anomaly detection compared to other approaches when multiple degrees of anomalies are considered. Moreover, we observe that the accuracy of anomaly detection increases as more neurons are involved in the computation procedure. In our future work we aim to focus on the likelihood of time-series predictions and improving performance of anomaly detection using learning and representation techniques.

## VII. Acknowledgment

This work was supported in part by the European Commission under the grant agreement number 825496 (5G-MOBIX) and by the German Federal Ministry of Transport and Digital Infrastructure under the grant number 01MM20004 (BeIntelli).

## References

[1] X. Ma, H. Zhong, Y. Li, J. Ma, Z. Cui, and Y. Wang, "Forecasting transportation network speed using deep capsule networks with nested lstm models," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–12, 2020.

[2] Y. Bi, C. Lin, H. Zhou, P. Yang, X. Shen, and H. Zhao, "Time-constrained big data transfer for sdn-enabled smart city," *IEEE Communications Magazine*, vol. 55, no. 12, pp. 44–50, 2017.

[3] S. Zeadally, M. A. Javed, and E. B. Hamida, "Vehicular communications for its: standardization and challenges," *IEEE Communications Standards Magazine*, vol. 4, no. 1, pp. 11–17, 2020.

[4] A. H. Sodhro, S. Pirbhulal, G. H. Sodhro, M. Muzammal, L. Zongwei, A. Gurtov, A. R. L. de Macêdo, L. Wang, N. M. Garcia, and V. H. C. de Albuquerque, "Towards 5g-enabled self adaptive green and reliable communication in intelligent transportation system," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–9, 2020.

[5] M. H. Hassan, A. Tizghadam, and A. Leon-Garcia, "Spatio-temporal anomaly detection in intelligent transportation systems," *Procedia Computer Science*, vol. 151, pp. 852–857, 2019.

[6] K. K. Santhosh, D. P. Dogra, and P. P. Roy, "Anomaly detection in road traffic using visual surveillance: A survey," *ACM Comput. Surv.*, vol. 53, no. 6, Dec. 2020.

[7] S. S. Sarikan and A. M. Ozbayoglu, "Anomaly detection in vehicle traffic with image processing and machine learning," *Procedia Computer Science*, vol. 140, pp. 64–69, 2018, cyber Physical Systems and Deep Learning, Chicago, Illinois November 5-7, 2018.

[8] M. U. Farooq, N. A. Khan, and M. S. Ali, "Unsupervised video surveillance for anomaly detection of street traffic," *International Journal of Advanced Computer Science and Applications (IJACSA)*, vol. 12, no. 8, pp. 270–275, 2017.

[9] T. S. Buda, B. Caglayan, and H. Assem, "Deepad: A generic framework based on deep learning for time series anomaly detection," in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 2018, pp. 577–588.

[10] S. C. Tan, K. M. Ting, and T. F. Liu, "Fast anomaly detection for streaming data," in *Twenty-Second International Joint Conference on Artificial Intelligence*, 2011.

[11] M. Munir, S. A. Siddiqui, A. Dengel, and S. Ahmed, "Deepant: A deep learning approach for unsupervised anomaly detection in time series," *IEEE Access*, vol. 7, pp. 1991–2005, 2018.

[12] M. Munir, S. A. Siddiqui, M. A. Chattha, A. Dengel, and S. Ahmed, "Fusead: unsupervised anomaly detection in streaming sensors data by fusing statistical and deep learning models," *Sensors*, vol. 19, no. 11, p. 2451, 2019.

[13] Y. Choi, H. Lim, H. Choi, and I.-J. Kim, "Gan-based anomaly detection and localization of multivariate time series data for power plant," in *2020 IEEE International Conference on Big Data and Smart Computing (BigComp)*, 2020, pp. 71–74.

[14] P. Boniol and T. Palpanas, "Series2graph: Graph-based subsequence anomaly detection for time series," *Proc. VLDB Endow.*, vol. 13, no. 12, p. 1821–1834, Jul. 2020.

[15] "Diginet-ps project," https://diginet-ps.de/en/home/, 2017.

[16] F. Sivrikaya, M. A. Khan, C. Bila, and S. Albayrak, "Reciprocal impact of autonomous vehicles and network resource management," in *2017 IEEE Vehicular Networking Conference (VNC)*, 2017, pp. 231–234.

[17] G. E. Box and D. A. Pierce, "Distribution of residual autocorrelations in autoregressive-integrated moving average time series models," *Journal of the American statistical Association*, vol. 65, no. 332, pp. 1509–1526, 1970.

[18] J. L. McClelland, D. E. Rumelhart, P. R. Group *et al.*, *Parallel distributed processing*. MIT press Cambridge, MA, 1986, vol. 2.

[19] D.-C. Park, "Centroid neural network for unsupervised competitive learning," *IEEE Transactions on Neural Networks*, vol. 11, no. 2, pp. 520–528, 2000.