Contrastive Learning-Based Air Traffic Trajectory Representation: A Case Study on Incheon International Airport

Thaweerath Phisannupawong M.S. student, Korea Advanced Institute of Science and Technology, Department of Aerospace Engineering, 34141, Daejeon, South Korea. petchthwr@kaist.ac.kr

Joshua Julian Damanik Phd. student, Korea Advanced Institute of Science and Technology, Department of Aerospace Engineering, 34141, Daejeon, South Korea. joshuad@kaist.ac.kr

Han-Lim Choi Professor, Korea Advanced Institute of Science and Technology, Department of Aerospace Engineering, 34141, Daejeon, South Korea. hanlimc@kaist.ac.kr

ABSTRACT

Airspace modernization has become an interest for the Air traffic management community. Trajectory recognition as a part of an autonomous system has been developed to follow this trend. Moreover, moving objects' trajectories has been minimally explored compared to other time series. This paper introduces an artificial intelligence-based approach to learn the latent representation of trajectories for downstream recognition tasks. The contrastive trajectory representation learning framework is demonstrated on public unlabeled air traffic surveillance data. The learning model exploits the unchanged course of the aircraft conducting aeronautical procedures to define the similarity of samples. Using the contrastive objective function, the encoder learns to maximize the agreement in representation for similar subseries while setting apart the global sampled negatives from the whole dataset. The model uses sliding windows encoding to transform the trajectories into more generalizable terms. The clusterability of the encoded trajectories was compared with the clustering of the corresponding raw trajectories. Despite the lower Silhouette scores, the results suggest that using the transformed data generates a more elaborative grouping from a comprehensive point of view for both arrival and departure air traffic data.

Keywords: Contrastive Learning; Air Traffic Management; Representation Learning; Time Series Analysis; Trajectory Clustering

Nomenclature

D	=	Discriminator function
Enc	=	Encoder function
E, E'	=	Number of features, Encoding size
η	=	Size of reference sub series
N	=	Number of samples
p	=	p-value
r, θ	=	Polar positional components (radius, bearing)

S, S'	=	Number of measurements (Number of time steps), Encoding length
t	=	Time step
t_p, t_n	=	Centroid of sub series (Positive, Negative)
u_x, u_y, u_z	=	Directional cosine vector components
w	=	Encoding window size
x, y, z	=	Cartesian positional components in ENU coordinates
X	=	Trajectory array
x_t, x_p, x_n, x_{ref}	=	Sub series (Anchor, Positive, Negative, Reference)
z	=	Encoded sub series
z_t, z_p, z_n	=	Encoded sub series (Anchor, Positive, Negative)
Z	=	Encoded trajectory array
δ	=	Sliding gap
μ, σ	=	Mean, Standard deviation

1 Introduction

Trajectory-based operations (TBO) have become interested in airspace modernization to enhance the safety and capability of the strategic air traffic flow management system. The autonomous operation has been developed to flexibly manage large high-dimensional trajectory data. As a part of the system, air traffic trajectory recognition has been a topic of interest in the air traffic management community, especially for a metropolitan airport with a high-density traffic flow and complex air traffic pattern. Although the aircraft trajectories in the airspace have recognizable patterns designated by the published maneuvering procedures, the air traffic controller may assign direct instruction or advice on the deviation from the standard courses, causing a challenge to identify the non-standard trajectories. The artificial intelligence model has successfully extracted useful information from the time series to learn the pattern.

Nowadays, a massive amount of aircraft trajectory data can be obtained from a publicly available data source, such as the collected Automatic Dependent Surveillance–Broadcast (ADS-B) signal recorded by surveillance facilities. The publicly available ADS-B data contain the information on aircraft's identifications, positions, velocities, headings, and the time of broadcasting, and the data can be reorganized and preprocessed into the time-series of positional information, the trajectory of a specific aircraft; however, the data are usually unlabeled. Therefore, to extract the information from the massive amount of unlabeled trajectory data, various clustering algorithms have been widely applied to provide groups of similar trajectories.

The trajectories as time series are usually rich in information, complex, and highly dimensional; transforming them into a more generalizable representation can better perform downstream tasks such as classification and clustering. Unsupervised contrastive representation learning is successful in many applications, yet contrastive learning on time series data was less popular than vision and natural language processing. Although many real-world time series, such as sound waves, electromagnetic signals, medical signals, human activity data, etc., have been demonstrated in recent works, the trajectory data of moving vehicles still needs to be explored. This paper proposes an unsupervised contrastive representation learning model for air-traffic trajectory data, exploiting the nature of air-traffic trajectories. In the airspace, the courses are usually unchanged while the aircraft is flying between two waypoints; this duration could be considered as the reference time boundary. The proposed technique aims to put together, in the feature space, the representations of the sub-trajectories within the determinable time boundary and push away the representations of the sub-trajectories outside this region. Using the unlabeled data, the paper further demonstrates the clusterability of the embeddings compared to the pre-embedded trajectories to show that implementing downstream clustering task on the embeddings provide separability among the overlapping clusters.

2 Related Works

The auto-encoders had been a widely applied architecture for trajectory representation learning. The autoencoder architecture consists of the encoder attached to the decoder, and they are trained simultaneously to reconstruct the original inputs. In this case, the learned representation usually refers to the meaningful latent space between the encoder and decoder. Prior works [1, 2] utilized the auto-encoder architectures for trajectory feature learning to reduce the computational burden in trajectory clustering by compressing the information in trajectories into meaningful latent space. The trajectory variational autoencoder (TrajVAE) has been proposed for a more complex model in [3]. Although the VAE was built mainly for trajectory generation, the architecture extracts meaningful information from the original trajectory. Apply the masking strategy in Traj-MAE [4], the masked autoencoder can be utilized to predict future trajectory state, which is beneficial for collision prediction of vehicles. The deep trajectory clustering in [5] did not incorporate only the reconstruction loss function for trajectory feature learning but enforced the clustering loss to ensure the separation of the latent space representation.

Contrastive learning is to perform the optimization in the embedding space such that the embedding of similar samples from the positive set is put together in the feature space closer to the anchor sample. In contrast, embeddings of dissimilar samples from the negative set are pushed away from the anchor and positive set [6–8]. In this fashion, contrastive learning has been a powerful self-supervised learning technique, especially for vision [8–11], and natural language processing [12–14]. On the other hand, time series were less studied, and even less on the trajectories of moving objects compared to other types of time series signals. Contrastive learning has become popular in representation learning because the time series are often highly oscillated or too complex to be reconstructed. Therefore, the more dimensional the time series, the more challenging it is to fit the autoencoder, and the learned latent space could underrepresent the original time series. Contrastive Predictive Coding (CPC) [15] maximizes the mutual information between the encoded representation of sub-series using InfoNCE loss. The technique attaches an autoregressive model to the encoder. It maximizes the agreement between the future subseries' latent space and the prediction of them using the information encoded from the previous sub-series. The work in [16] demonstrates that Unsupervised Scalable Representation Learning is applicable to the set of time series with unequal length. The positive samples are selected randomly within the reference region using uniformly random distribution while letting the negative be the sub-series outside the interested series. The encoder is trained with the time-based triplet loss, ensuring the similarity of positive subseries and distinguishability from negative subseries. Temporal neighborhood coding (TNC) [17] determines the positive region within a sample by exploiting the local stationarity of the time series. The positive and negative sets are defined within a series without accessing others. The TNC framework was built with the encoder and the discriminator; they are trained simultaneously by encouraging the discriminator output be the same for the positive and oppositely for the negative sets. The TNC loss applies a debias parameter to ensure the contrastive learning objective of the encoder using the discriminator's decision. Some literature augments the raw input data to obtain different data with identical contexts. Representation learning via Temporal and Contextual Contrasting (TS-TCC) [18] is one of many successful works. This method demonstrated the weak and strong augmentation of the original time series. It maximizes these two augmentation's similarity while distinguishing them from all other series. The Bilinear Temporal-Spectral Fusion (BTSF) [19] demonstration shows that dropout can be used as an augmentation process. Moreover, the work considered the spectral domain of time series by combining the encoded product of time and the spectral domain into the bilinear feature map; hence, it was successful for medical signals. This paper will propose a contrasting representation of learning for air traffic trajectory as they are rarely explored in prior literature.

3 Methodology

This paper proposes an unsupervised contrastive representation learning technique for the aircraft trajectory data within the airport terminal area. The detailed explanation of methods is explained in the section. The implementation includes aircraft trajectory dataset preparation, the model training pipeline construction, and the experiments.

3.1 Dataset Preparation

The experiments in this paper were done using the publicly available historical Automatic Dependent Surveillance–Broadcast (ADS-B) data recorded in the OpenSky database [20], considering the area within 150 kilometers circumstancing the Incheon international airport (ICAO airport code: RKSI). This paper's historical trajectory dataset was from the ADS-B state vectors query released between 2018 and 2023. Since the Incheon International Airport shares the operational fixes with the Gimpo International Airport, the trajectories not belonging to the arrival and departure of the Incheon International Airport were filtered out. Besides, trajectories that are too short, too long, and incomplete were filtered out to clean the data.

The critical tracking information in an ADS-B state vector includes broadcasting time, icao24 code, latitude, longitude, velocity, heading, vertical speed, call sign, on-ground indication, barometric altitude, geometric altitude, etc. The barometric altitude was selected as a vertical position state as it is published in the Aeronautical Information Publications (AIPs) for the aircraft operator to comply. Therefore, the latitude, longitude, and barometric altitude were transformed into the Cartesian position x, y, and z in the ENU (East-North-Up) coordinates, the local tangent plane coordinates, centered at the Incheon International Airport. Since the ADS-B state vector may not be recorded constantly, the trajectory data were resampled and interpolated to fill in the missing data points. Then, the outlier states were removed, and the Savitzky-Golay filter was applied for a smoother trajectory. Finally, the smoothed trajectories were unified in step length by linear interpolation.

After obtaining the Cartesian position in x, y, and z stacked along the time steps, t, the heading and flight path are also valuable features that describe the shape of the trajectory. Moreover, when an aircraft maintains the course, it conducts a particular assigned procedure; the positional states within this period are parts of the procedural track. The directional cosine vectors were calculated for each consecutive position to describe the heading and the flight path angle. The directional vector is a vital element for the proposed method in this paper, as it will exploit the unchanged angles to determine similar samples for the learning model. Furthermore, to make the input features more expressive, the cylindrical coordinates are sometimes preferred for air-traffic management; therefore, the lateral radius, r, and bearing angle, θ, were calculated for each lateral points x, and y in the trajectories. The dataset was normalized within $[-1, 1]$ using minimum-maximum scaler. The features provide the context of how far the aircraft is from the airport and which angular sector it is located.

The trajectory data for Incheon International Airport is generally imbalanced because both arrival and departure flights are densely in the south and southeast of the airport. With this characteristic, the undersampling was done on the southbound and southeastbound trajectories, creating a balanced dataset of trajectories bound from the west, northeast, southeast, and south of the airport. Each arrival and departure flight data consists of a 5000-time series of aircraft trajectory states without labels, partitioned into 2,500 training samples and 2,500 clusterability testing samples. Each samples has 1000 measurements with eight features, including three directional features from the directional cosine vector of the flight path u_x, u_y, u_z, and five positional features from 3 Cartesian elements x, y, z and cylindrical elements, r, and θ. A separate raw trajectory dataset was prepared for comparative clustering. This dataset contains the original Cartesian positions in $x, y,$ and z and consists of 1,000 measurements.

3.2 Training Pipeline

This paper introduces the method for contrastive representation learning, exploiting the behavior of air-traffic trajectories. With this setting, it is crucial to set the boundary of sampling in such a way that the positive samples are similar and distinguishable from the negative samples. For N samples of time series, let a multivariate time series denoted as X having a dimension of $E \times S$, while E is denoted as the number of features and S is the sequence length. Pre-encoded subseries are denoted as x centering at the sampled time $t \in [\frac{w}{2}, S - \frac{w}{2}]$, comprise the states in the time window $[t - \frac{w}{2}, t + \frac{w}{2}]$.

The statistical tests have been proven to be an efficient method to set the boundary of the region of positive reference for time series [17]. As the sampling procedure in this paper exploits the unchanged direction during aircraft flying, the distribution in the directional features should have a constant trend with low variance within a window. Therefore, the statistical tests can be applied to test whether the distribution of the anchor time window x_t and the region of reference x_{ref} are indifferently normally distributed. The framework employs the two-sample t-test for means and two-sample Levene's test for variances. To test the indifference in means and variances, the null hypotheses in testing are $H_{o,t-test} : \mu_t = \mu_{ref}$, and $H_{o,Levene} : \sigma_t^2 = \sigma_{ref}^2$). To combine the p-values from two statistical tests, Fisher's method was applied to calculate $p_{combined} = 1 - F\left(-2\left(\ln(p_{\text{t-test}}) + \ln(p_{\text{levene}})\right); 4\right)$. The centering time t is first sampled for a trajectory sample $X_i, i \in [1, N]$, creating an anchor subseries x_t of window size w.

To construct the positive set, the region of reference x_{ref} is repeatedly determined by the combined test; in each test, the size of x_{ref} increases until the test returns the p-value, indicating the difference in two distributions. The maximum size of the reference region is denoted as η. The assumption was made similarly to [17] that the signals closer to the anchor subseries in time are more similar to it. For this assumption, the positive sampling modeled by Gaussian distribution was applied. The centroid of positive samples has Gaussian distribution $t_p \sim \mathcal{N}(t, \eta)$, and the positive subseries x_p comprise the states in the time windows $[t_p - \frac{w}{2}, t_p + \frac{w}{2}]$.

On the other hand, another assumption was made for negative samples that the context of positive does not exist in different time series samples, even though there might be some, similar to the negative-sampling word-embedding method [16, 21]. Thus, the negative subseries are randomly selected from other trajectories. Two uniform distributions are applied. First, the candidate trajectories were randomly selected from the dataset X, excluding X_i denoted as the subset $X \setminus \{X_i\}$ by allowing duplicates to be sampled. Then, the time centriods of negative set t_n are sampled from the uniform distribution $t_n \sim U(\frac{w}{2}, S - \frac{w}{2})$. Note that both candidate trajectories sampling and negative centroids sampling were conducted using the same sample size. The negative set can now be constructed by matching the candidate trajectories to the centroids; as the results, x_n comprise the states in the time windows $[t_n - \frac{w}{2}, t_n + \frac{w}{2}]$ of the time series $X_j \in X \setminus \{X_i\}$, while $i, j \in [1, N]$. Fig 1 visualizes the positive and negative sampling.

The representation, z of any subseries x, are obtained from the differentiable encoder model; $z = \text{Enc}(x)$. The anchor, positive, and negative subseries representations are denoted as z_t, z_p, and z_n, respectively. The framework utilized the Transformer encoder model presented in [22]. In the encoder's architecture, the linear projection layer expands the feature dimensions of subseries x for more expressiveness; then, the class token c as learnable parameters are appended as the first-time element. After passing the inputs to the transformer encoder, the class token is extracted and mapped to the encoding size E' as the representation z. The architecture of the Transformer encoder used in this paper is illustrated in Fig 2. According to the assumption of the negative sampling method stated that some samples in the negative set are the positive samples; to encourage the similarity of z_t and z_p, and distinguishability of z_t and z_n, the TNC loss [17] were implemented, for it has been proven to be robust for bias sampling in the Positive-Unlabeled (PU) learning. The original TNC loss function was modified according to the sampling procedures, and the loss function used in this framework is expressed in equation (1). The w parameter represents the probability of having positive samples in the negative set.

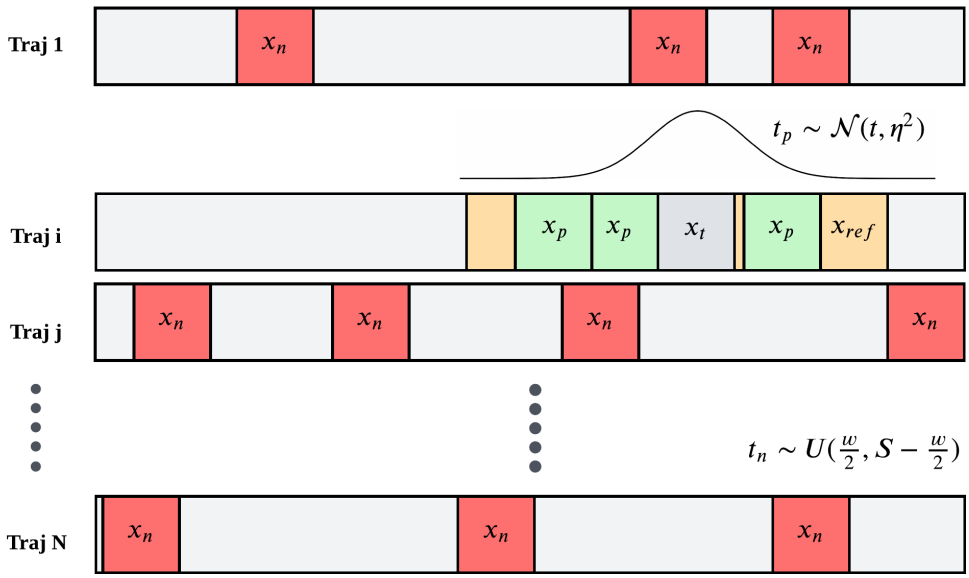


Fig. 1 Sampling visualization for $x_{ref}, x_p,$ and x_n

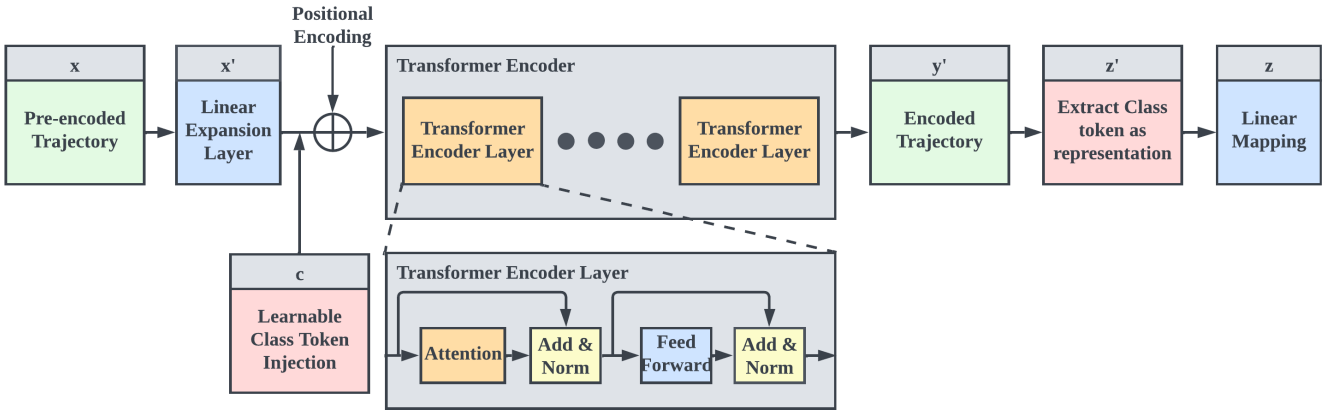


Fig. 2 Architecture of Transformer Encoder used in trajectory representation learning

$$\mathcal{L} = -\mathbb{E}_{x_t \sim X}\left[\mathbb{E}_{x_p \sim x_{ref}}\left[\log D(z_t, z_p)\right] + \mathbb{E}_{x_n \sim X \setminus \{X_i\}}\left[(1-w)\log(1 - D(z_t, z_n)) + w \log D(z_t, z_n)\right]\right] \quad (1)$$

A Discriminator is employed to approximate the probability of a representation, z is the same as the anchor representation z_t; the sigmoid output of the discriminator is denoted as $D(z_t, z)$. For the architecture, the discriminator in this framework is a simple multi-headed binary classifier that returns 1 when z_t and z are the same and returns 0 when z_t and z are different. The encoder and the discriminator jointly learn to optimize the loss function in equation 1 using the newly sampled anchor sample x_t positive samples x_p, and negative samples x_n in each training epoch. After the training, the discriminator will not be used; only the encoder will be used to transform the original subtrajectory into representation.

For this paper's trajectory data, the window size was set at $w = 20$ measurements; the linear projection layer expands eight features to the dimension of 64. The transformer encoder has three layers; each consists of single head attention and 256 units in the feed-forward layer with 0.1 drop rate. The token mapping layer transforms 64 elements of the token into the encoding size of 32. There are five revisiting times for training for a single time series sample. Since the trajectory data significantly rely on in-batch negatives, the sampling size for both positive and negative sets is 1000 samples.

3.3 Experiments

Contrastive representation learning typically aims to obtain a better performance on the downstream tasks, generally referring to classification and clustering. However, since the experimental dataset is unlabeled, the analysis of representation learning performance was done by testing the clusterability of the encoeded data. According to the framework architecture, the encoder transforms the sub-trajectories into representation; the whole trajectory can be transformed using sliding windows encoding. For the time $t \in [\frac{w}{2}, S-\frac{w}{2}]$ with adjustable sliding gaps δ. The encoded trajectories have the dimension of $E' \times S'$, while E' is adjustable encoding size, and S' is encoding length following equation (2). For example, the experiment throughout this paper set the sliding gap at $\delta = 5$; the encoding length, $S' = 200$.

$$S' = \left\lfloor \frac{(S - \frac{w}{2}) - \frac{w}{2}}{\delta} \right\rfloor + 1 + 2\left(\frac{w}{2 \times \delta}\right) \tag{2}$$

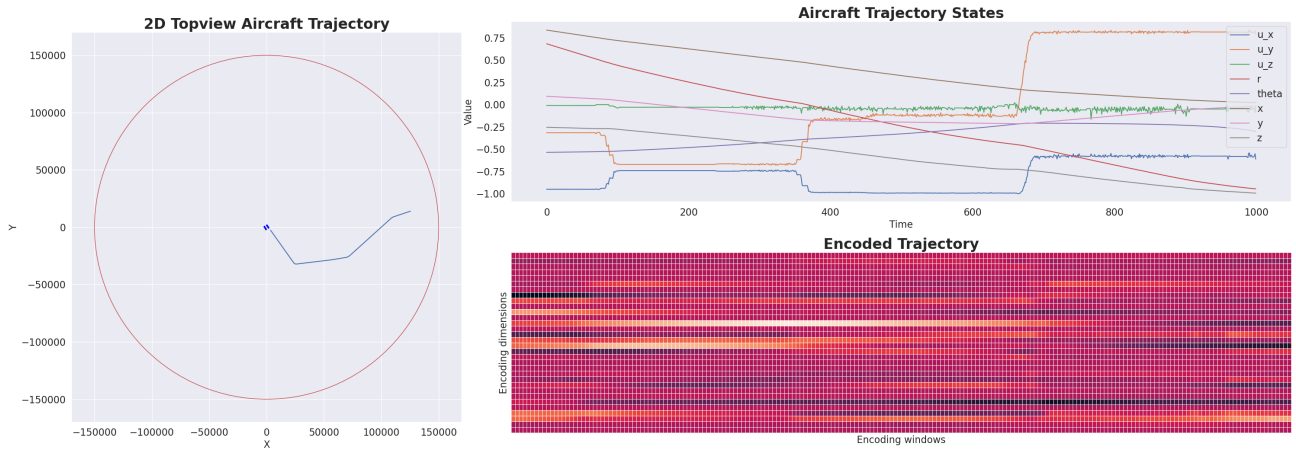
The clustering experiment compared the clusterability between the raw trajectories and encoded data. The Hierarchical Density-Based Spatial Clustering of Applications with Noise (HDBSCAN) [23] was selected in this experiment, for it can identify clusters of various shapes and densities without requiring the number of clusters to be specified. Moreover, unlike DBSCAN, which requires the epsilon parameter to specify the radius of the neighborhood around each sample, HDBSCAN does not require such a parameter to be set as such radius is automatically determined. The distance metric between two samples in HDBSCAN is measured using Dynamic Time Warping (DTW) [24]. The DTW was employed for both since raw 3-dimensional trajectory data and encoded trajectories have temporal dynamics. Unlike point-to-point Euclidean distance, DTW uses dynamic programming to approximate the best-aligned point pairing to calculate the distance between two temporal sequences. After obtaining the cluster assignments, the Silhouette scores were calculated for every cluster the HDBSCAN algorithm can group. The higher scores indicate that the object is well-matched with its cluster and poorly matched to the neighboring cluster. The clustering results were visualized using t-SNE, which produced 2-dimensional scatter plots to illustrate the groupings of similar samples, along with grouped trajectories plots for comprehensible view. The framework was implemented using Python 3.11 and Pytorch 2.0.1, trained on a machine with NVIDIA GeForce RTX 4090 GPU. The testing data analysis was done in both benchmarking and comprehensive plotting of the clusters.

4 Results and Discussion

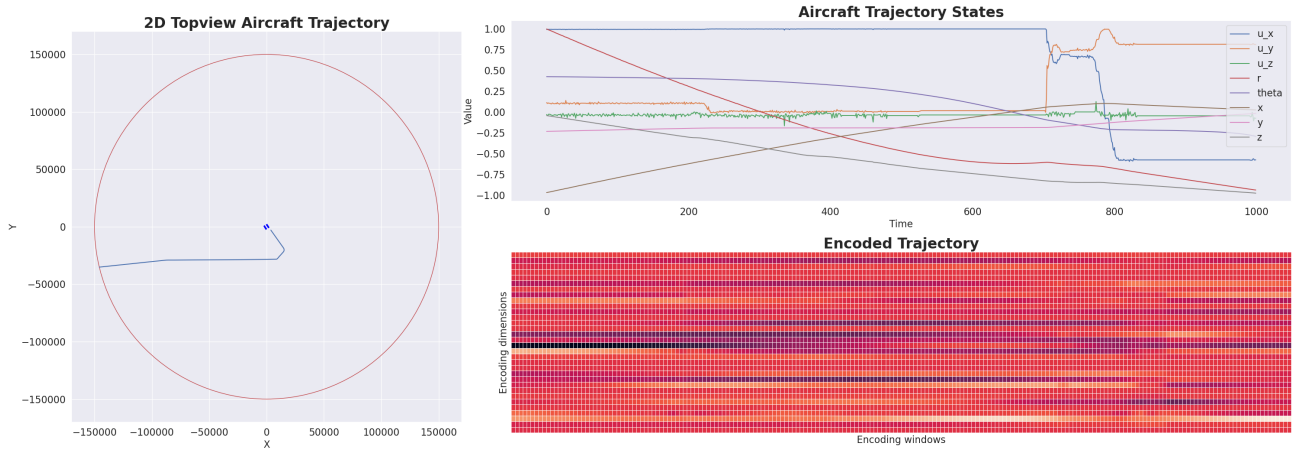
This section elaborates on the encoded trajectories' characteristics, supported by illustrative examples. It then presents the clusterability results of the encoded trajectories with the experiment setting discussed in the previous section. Further discussion and analysis of these results are also provided.

4.1 Encoded Trajectory

With the window size $w = 20$ and sliding gap $\delta = 5$, the encoded trajectories Z are two-dimensional arrays, having the extent of 200×32. Sliding through the windows of x_t over the time step, each column with 32 elements is the embedded z_t, representing a sliding window x_t with the size of 20×8. As illustrated in Fig 3, the embeddings capture the states within each sliding window. In a broader context, these embeddings collectively represent the temporal dynamics of the entire trajectory. A smaller sliding gap can also capture the transition when the aircraft changes its course, thereby providing a better representation of the trajectory's continuous nature; consequently, it takes a longer inference time for encoding. As illustrated in Figure 3, the encoded representations of eastbound and westbound trajectories show distinct patterns. The local positive sampling encourages the maximization of the coherence within the reference part of individual trajectories. On the other hand, the global negative sampling effectively increases the dissimilarity between representations across different trajectories, reflecting the unique



(a)



(b)

Fig. 3 Example Trajectory Plots with Trajectory states and Encoded Trajectory. (a) shows the eastbound trajectory, and (b) shows the westbound trajectory.

representation of each direction the aircraft is traveling. All trajectories were encoded and standardized, retaining their characteristics, and stored for use in the clustering experiment. The clusterability results will provide a comprehensive view of similarity and distinction for the entire dataset.

4.2 Clusterability

The clusterability experiment was conducted on both raw trajectory and encoded trajectory, as well as departure and arrival scenarios. The raw 3-dimensional trajectory dataset contains 1000 measurements, the same size as the trajectory states dataset; however, to reduce the computational time for the DTW distance calculation, the raw trajectory was downsampled to 250 measurements while retaining most of the vital information. The results obtained using HDBSCAN with Dynamic Time Warping (DTW) distance are as follows: 2-dimensional t-SNE visualization of clusters, individual plots for separated trajectories, and the average silhouette score for each group.

4.2.1 Arrival data analysis

Refers to appendix A.1, Table 1 shows the Silhouette scores for arrival trajectories analysis. Comprehensive visualizations, including t-SNE and individual cluster trajectory plots, are presented in Fig. 4. The raw trajectories effectively separate the trajectories that are different in early approach patterns. For example, as evident in the clusters pair 6-7 and 9-10, the trajectories in each pair have different

entry points and later have a similar pattern; they are separated into two clusters. Nevertheless, using raw trajectories poorly differentiates the landed runways. This is because, in the Euclidean space, the distance between runways is closer than between entry points. In contrast, as seen in Fig. 4c and Fig. 4a, HBDSCAN clustering using the encoded data provides more apparent separations. In these separated clusters, the clusters pair 1-2, 7-11, 9-10, and 12-13 obtained by the encoded data represent the distinctly situated runways for the reason that in the latent space, the representation does not represent the physical position; however, they represent the piece of information setting apart from their negatives. The comprehensive analysis of the encoded data clustering effectively separates the overlapping clusters. The lower Silhouette scores for the encoded data could be the results from the global negative sampling that encourage the model to distinguish, from the anchor sample, the positive samples within the negative set. This technique may cause some trajectories to be outliers, even if they could be considered a cluster member.

4.2.2 Departure data analysis

Using the same metrics as the arrival trajectories, in the appendix A.2, the Silhouette scores for departure trajectory analysis are presented in Table 2, while Fig. 5 provides the t-SNE visualization and separately plots the trajectories grouped in the clusters. According to Fig. 5c and Fig. 5a, the using encoded trajectory performs more effectively on clustering, observable from a better clusters separation in t-SNE space comparing the t-SNE representation of the raw trajectory. This is because, with enough generalization ability, contrastive learning can group similar trajectories, although some sub-series identical to them are sampled in the negative set. HDBSCAN generated 8 clusters for the raw departure trajectory data, and the algorithm does not separate the cluster of distinct runways due to the mentioned characteristic of raw trajectories in Euclidean space. Although the parameter of minimum samples per cluster can be adjusted lower to obtain the fine-grained group, the number of samples considered as noise consequently increase. On the other hand, HDBSCAN generates more detailed clusters on the encoded data. In addition, the algorithm can recognize detailed departure patterns, for it can differentiate almost all patterns available in the dataset with few noise samples. Similar to the clustering on the encoded arrival data, the clusters of trajectories departed on different runways are well clustered for the same separability advantage as this occurred for the arrival dataset. The results in the comprehensive view for encoded data are visually better than those using raw trajectories, even though the Silhouette scores are lower. Same as the results from arrival data, the data groups together in t-SNE space, but it might be internally distant measured by DTW.

4.3 Discussion

Both analyses suggest that using the encoded trajectory data for downstream clustering tasks results in more fine-grained groups because the contrastive representation learning encourages the local neighboring subseries similarity and ensures distinction from the global negatives. For this reason, the encoder was optimized to set apart the subseries close to Euclidean distance to be distant in feature space. Therefore, the HDBSCAN on the encoded data can differentiate the points close together in physical space such as runways. However, the Silhouette scores are lower than the experiment on encoded and raw data. The global negative sampling might be responsible for this issue because the optimization also sets apart the positive samples within the negative set. To solve this problem further, fine-tuning for the loss function's weight parameter and model architecture search should be performed. Moreover, the results in this paper also rely on the performance of the clustering algorithm; therefore, fine-tuning the hyperparameters or changing the clustering algorithm could output differently. As the results of departure data and arrival data reflect different behaviors, experimenting on other datasets could provide additional insights into the adaptability and generalizability of the method proposed in this paper.

5 Conclusion

This paper aims to construct a contrastive learning model specifically for trajectory data, as it is the underexplored type of time series to facilitate downstream tasks like classification and clustering. Using an air-traffic management setting, the experimental data in this paper is collected from a publicly available ADS-B data source. As a result, the unlabeled dataset consists of arrival and departure trajectories circumstancing the Incheon International Airport. The learning framework applied statistical t-test and Levene's test to the directional features to test the identical distribution along the aircraft's track to determine the region of the unchanged flight path. The model was trained to encourage the similarity along the positive reference time and ensure the distinction to the globally sampled negatives. The lower Silhouette scores are evidence of setting apart the positive samples, resulting in a more considerable distance. However, the results on HDBSCAN show more clusters in well-separated visualization on the encoded testing data, emphasizing the distinguishability and familiarization of the trained encoder, further reflecting that the stationarity of the path direction can be used as the local context for trajectory data. Therefore, in future works, the fine-tuning of hyperparameters and the architecture search could be performed to obtain better results. Other clustering performance indices, such as the Davies-Bouldin score (DBI) or the Adjusted Rand score (ARI) with labels, can also be measured to confirm the results better. Finding the most appropriate algorithm could be challenging because the results are sensitive to the clustering algorithm. Furthermore, since the method in this paper determines the reference period from single time series samples, testing positivity across the samples is recommended for future works. The authors present this method as a preliminary step to encourage further studies on various trajectory datasets, learning models, and architectures to demonstrate further the versatility and broad applicability of contrastive learning for trajectory representation.

Appendix

A.1 Experimental Results: Arrival Traffic Data

Table 1 Cluster Silhouette Scores for Arrival Trajectories: Raw and Encoded

(a) (a) Arrival Raw Trajectories

Metric	Score
Overall Score	0.5846
No Noise Score	0.6311
Noise Samples	102
Cluster -1	-0.5086
Cluster 0	0.7588
Cluster 1	0.7172
Cluster 2	0.7495
Cluster 3	0.5667
Cluster 4	0.6721
Cluster 5	0.6582
Cluster 6	0.5712
Cluster 7	0.5925
Cluster 8	0.5529
Cluster 9	0.6745

(b) (b) Arrival Encoded Trajectories

Metric	Score
Overall Score	0.1680
No Noise Score	0.2084
Noise Samples	229
Cluster -1	-0.2329
Cluster 0	0.2376
Cluster 1	0.2319
Cluster 2	0.2137
Cluster 3	0.2325
Cluster 4	0.1412
Cluster 5	0.2430
Cluster 6	0.1879
Cluster 7	0.2789
Cluster 8	0.2968
Cluster 9	0.2174
Cluster 10	0.1689
Cluster 11	0.1613
Cluster 12	0.3010

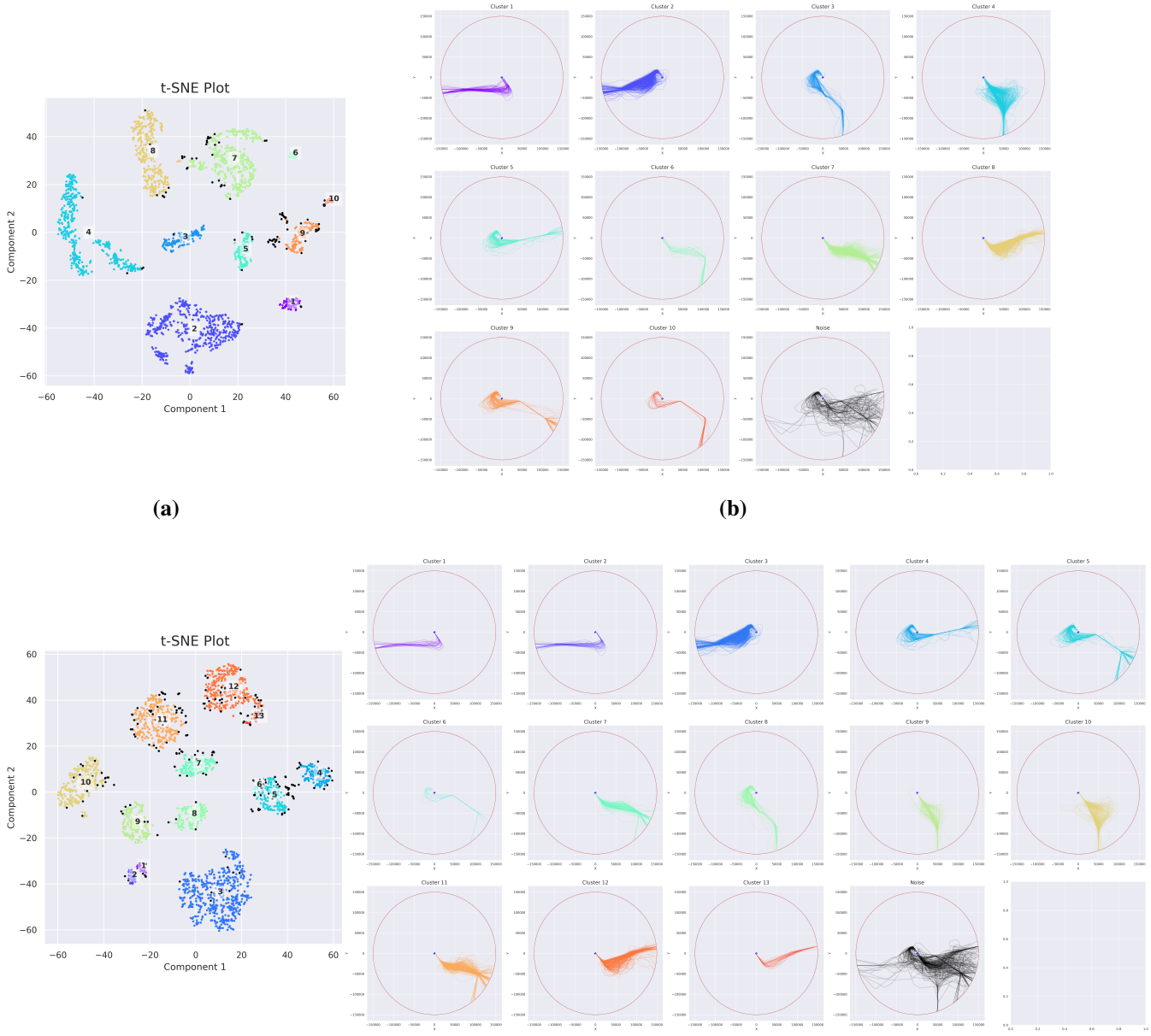


Fig. 4 Comprehensive Visualization of Trajectory Clustering: (a) t-SNE plot illustrating the clusters separation of raw trajectories, (b) Individual clustering of raw trajectories, (c) t-SNE plot illustrating the clusters separation of encoded trajectories, and (d) Individual clustering of encoded trajectories.

A.2 Experimental Results: Departure Traffic Data

Table 2 Cluster Silhouette Scores for Departure Trajectories: Raw and Encoded

(a) Departure Raw Trajectories

Metric	Score
Overall Score	0.5927
No Noise Score	0.5985
Noise Samples	11
Cluster -1	-0.7041
Cluster 0	0.7628
Cluster 1	0.4533
Cluster 2	0.6040
Cluster 3	0.7285
Cluster 4	0.5460
Cluster 5	0.9148
Cluster 6	0.4432
Cluster 7	0.4962

(b) Departure Encoded Trajectories

Metric	Score
Overall Score	0.2073
No Noise Score	0.2123
Noise Samples	27
Cluster -1	-0.2550
Cluster 0	0.2968
Cluster 1	0.4028
Cluster 2	0.1706
Cluster 3	0.2290
Cluster 4	0.3770
Cluster 5	0.3373
Cluster 6	0.1408
Cluster 7	0.3608
Cluster 8	0.4113
Cluster 9	0.3277
Cluster 10	0.4423
Cluster 11	0.1448
Cluster 12	0.1622
Cluster 13	0.1817
Cluster 14	0.1114
Cluster 15	0.1346
Cluster 16	0.1877
Cluster 17	0.3355

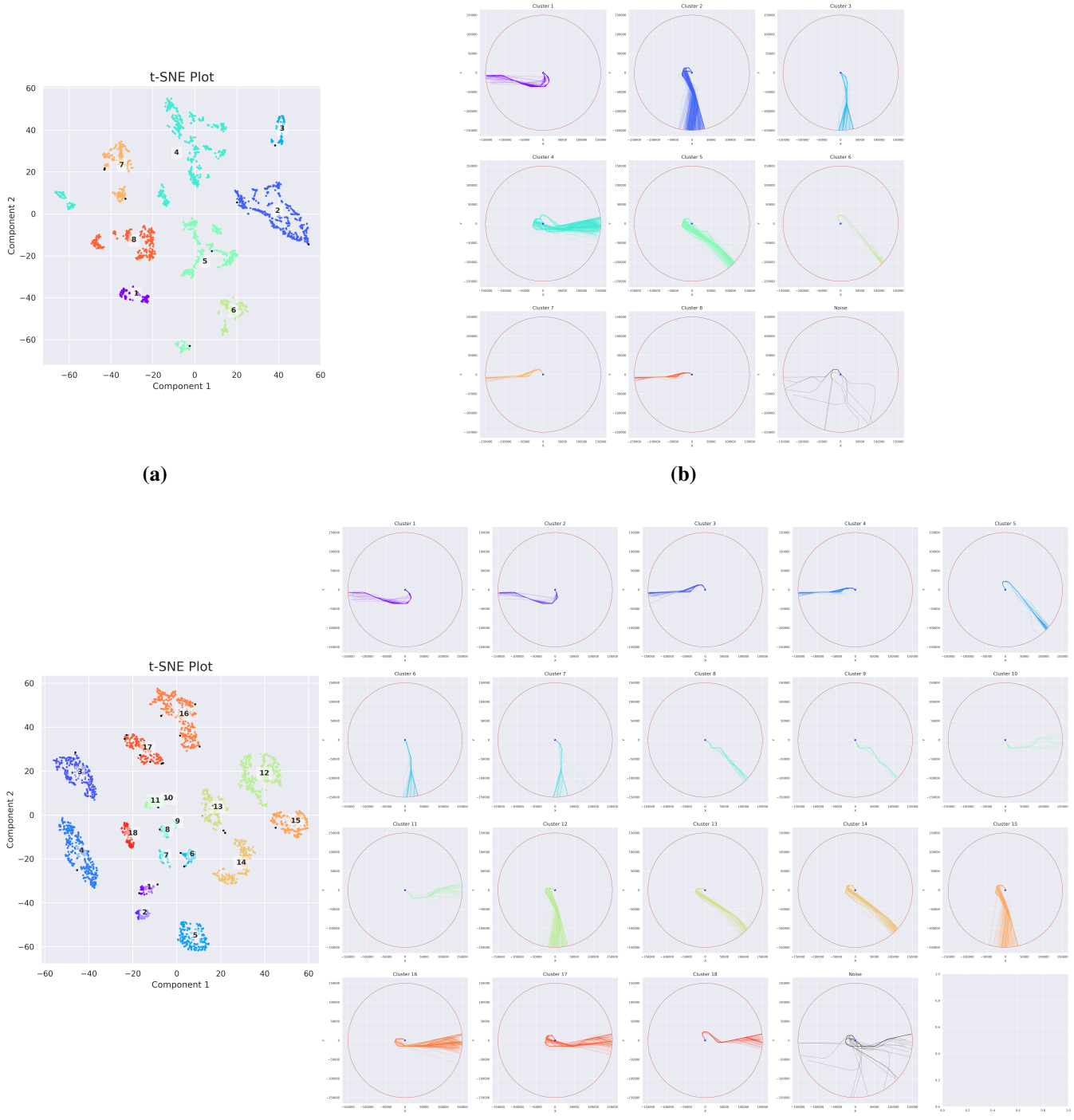


Fig. 5 Comprehensive Visualization of Trajectory Clustering for Departures: (a) t-SNE plot illustrating the clusters separation of raw departure trajectories, (b) Individual clustering of raw departure trajectories, (c) t-SNE plot illustrating the clusters separation of encoded departure trajectories, and (d) Individual clustering of encoded departure trajectories.

References

[1] Xiao Chu, Xianghua Tan, and Weili Zeng. A clustering ensemble method of aircraft trajectory based on the similarity matrix. *Aerospace*, 9(5), 2022. ISSN: 2226-4310. DOI: 10.3390/aerospace9050269.

[2] Weili Zeng, Zhengfeng Xu, Zhipeng Cai, Xiao Chu, and Xiaobo Lu. Aircraft trajectory clustering in terminal airspace based on deep autoencoder and gaussian mixture model. *Aerospace*, 8(9), 2021. ISSN: 2226-4310. DOI: 10.3390/aerospace8090266.

[3] Xinyu Chen, Jiajie Xu, Rui Zhou, Wei Chen, Junhua Fang, and Chengfei Liu. Trajvae: A variational autoencoder model for trajectory generation. *Neurocomputing*, 428:332–339, 2021. ISSN: 0925-2312. DOI: https://doi.org/10.1016/j.neucom.2020.03.120.

[4] Hao Chen, Jiaze Wang, Kun Shao, Furui Liu, Jianye Hao, Chenyong Guan, Guangyong Chen, and Pheng-Ann Heng. Traj-mae: Masked autoencoders for trajectory prediction, 2023.

[5] Xavier Olive, Luis Basora, Benoit Viry, and Richard Alligier. Deep trajectory clustering with autoencoders. In *ICRAT 2020, 9th International Conference for Research in Air Transportation*, 2020.

[6] S. Chopra, R. Hadsell, and Y. LeCun. Learning a similarity metric discriminatively, with application to face verification. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 539–546 vol. 1, 2005. DOI: 10.1109/CVPR.2005.202.

[7] R. Hadsell, S. Chopra, and Y. LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742, 2006. DOI: 10.1109/CVPR.2006.100.

[8] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. *CoRR*, abs/1503.03832, 2015.

[9] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. A simple framework for contrastive learning of visual representations. *CoRR*, abs/2002.05709, 2020.

[10] Hyun Oh Song, Yu Xiang, Stefanie Jegelka, and Silvio Savarese. Deep metric learning via lifted structured feature embedding. *CoRR*, abs/1511.06452, 2015.

[11] Alexander Kolesnikov, Xiaohua Zhai, and Lucas Beyer. Revisiting self-supervised visual representation learning. *CoRR*, abs/1901.09005, 2019.

[12] Andriy Mnih and Koray Kavukcuoglu. Learning word embeddings efficiently with noise-contrastive estimation. *Advances in neural information processing systems*, 26, 2013.

[13] Deshui Miao, Jiaqi Zhang, Wenbo Xie, Jian Song, Xin Li, Lijuan Jia, and Ning Guo. Simple contrastive representation adversarial learning for NLP tasks. *CoRR*, abs/2111.13301, 2021.

[14] Tianyu Gao, Xingcheng Yao, and Danqi Chen. Simcse: Simple contrastive learning of sentence embeddings. *CoRR*, abs/2104.08821, 2021.

[15] Aäron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *CoRR*, abs/1807.03748, 2018.

[16] Jean-Yves Franceschi, Aymeric Dieuleveut, and Martin Jaggi. Unsupervised scalable representation learning for multivariate time series. *CoRR*, abs/1901.10738, 2019.

[17] Sana Tonekaboni, Danny Eytan, and Anna Goldenberg. Unsupervised representation learning for time series with temporal neighborhood coding. *CoRR*, abs/2106.00750, 2021.

[18] Emadeldeen Eldele, Mohamed Ragab, Zhenghua Chen, Min Wu, Chee Keong Kwoh, Xiaoli Li, and Cuntai Guan. Time-series representation learning via temporal and contextual contrasting. *CoRR*, abs/2106.14112, 2021.

[19] Ling Yang and Shenda Hong. Unsupervised time-series representation learning with iterative bilinear temporal-spectral fusion, 2022.

[20] Matthias Schäfer, Martin Strohmeier, Vincent Lenders, Ivan Martinovic, and Matthias Wilhelm. Bringing up opensky: A large-scale ads-b sensor network for research. In *IPSN-14 Proceedings of the 13th International Symposium on Information Processing in Sensor Networks*, pages 83–94, 2014. DOI: 10.1109/IPSN.2014.6846743.

[21] Yoav Goldberg and Omer Levy. word2vec explained: deriving mikolov et al.'s negative-sampling word-embedding method. *CoRR*, abs/1402.3722, 2014.

[22] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017.

[23] Claudia Malzer and Marcus Baum. Hdbscan(ϵ
): An alternative cluster extraction method for HDBSCAN. *CoRR*, abs/1911.02282, 2019.

[24] Mohammad Shokoohi-Yekta, Bing Hu, Hongxia Jin, Jun Wang, and Eamonn J. Keogh. Generalizing dtw to the multi-dimensional case requires an adaptive approach. *Data Mining and Knowledge Discovery*, 31:1–31, 2016.