Towards Self-supervised Learning for Multi-function Radar Behavior State Detection and Recognition

This paper was downloaded from TechRxiv (<https://www.techrxiv.org>).

LICENSE

CC BY 4.0

SUBMISSION DATE / POSTED DATE

07-01-2022 / 11-01-2022

CITATION

Feng, HanCong (2022): Towards Self-supervised Learning for Multi-function Radar Behavior State Detection and Recognition. TechRxiv. Preprint. <https://doi.org/10.36227/techrxiv.17976062.v1>

DOI

[10.36227/techrxiv.17976062.v1](https://doi.org/10.36227/techrxiv.17976062.v1)

Towards Self-supervised Learning for Multi-function Radar Behavior State Detection and Recognition

Han Cong Feng XinYuZhao Yin Xiong Bin Tang

School of Information and Communication Engineering, UEST

Abstract— The analysis of intercepted multi-function radar (MFR) signals has gained considerable attention in the field of cognitive electronic reconnaissance. With the rapid development of MFR, the switch between different work modes is becoming more flexible, increasing the agility of pulse parameters. Most of the existing approaches for recognizing MFR behaviors heavily depend on prior information, which can hardly be obtained in a non-cooperative way. This study develops a novel hierarchical contrastive self-supervised-based method for segmenting and clustering MFR pulse sequences. First, a convolutional neural network (CNN) with a limited receptive field is trained in a contrastive way to distinguish between pulse descriptor words (PDW) in the original order and the samples created by random permutations to detect the boundary between each radar word and perform segmentation. Afterward, the K-means++ algorithm with cosine distances is established to cluster the segmented PDWs according to the output vectors of the CNN's last layer for radar words extraction. This segmenting and clustering process continues to go in the extracted radar word sequence, radar phase sequence, and so on, finishing the automatic extraction of MFR behavior states in the MFR hierarchical model. Simulation results show that without using any labeled data, the proposed method can effectively mine distinguishable patterns in the sequentially arriving PDWs and recognize the MFR behavior states under corrupted, overlapped pulse parameters.

Index Terms— Electronic warfare; Multi-function radars, Self-supervised learning; Clustering

I. INTRODUCTION

Precisely analyzing the pulse patterns generated by multi-function radars (MFR) is a crucial task for Cognitive Electronic Warfare. MFRs can operate in numerous work modes, such as surveillance, searching, and target tracking in a time-division multiplexing way[1], [2] and scheduling various tasks by the resource management software based on the awareness of the environment[3], [4]. Besides, the MFR can vary its intra-pulse and pulse to pulse parameters such as pulse width (PW), pulse repetition interval (PRI), radio frequency (RF) on the fly. As a result, MFR signals intercepted in a non-cooperative way have a high level of agility, which makes the behavior state recognition extremely challenging.

MFRs can be effectively modeled as discrete event systems (DES), Visnevski first proposed a hierarchical model for

describing the behaviors of MFRs, in which the MFR signals are represented by radar pulses, radar words, and radar phrases. With the hierarchical model, several syntax-based methods[2], [5], as well as the statistical-based methods[6–10], have been proposed to analyze the MFR behavior state. However, most of those methods have only been proven effective for some simple radars like the Mercury[6]. Later on, researchers focus on neural networks for MFR recognition and introduce several models such as Long Short-Term Memory networks(LSTM)[7]–[9], gated recurrent unit(GRU)[10], discrete process neural network[11]. With enough training samples, neural network-based methods outperform traditional statistical-based methods under more complex tasks[12].

However, all these methods mentioned above heavily rely on prior information. For example, the statistical-based methods need to know the state transition rules to function properly while the neural network or machine learning-based approach needs a sufficient amount of labeled samples. However, marking and annotating the MFR pulse sequence or figuring out the state transition rules is not an easy task when the data is heavily corrupted.

Currently, there are only a few approaches that address the problem of blind analysis of MFR pulse patterns. Liu[13] introduces a deep recurrent neural network (RNN) coarsely cluster different pulse groups in an unsupervised way, however, a supervised classifier is still needed for pulse groups classification. Unsupervised time series clustering is introduced by Zhu et. al[14] to segment and cluster the PRI sequence. However, this model-based method is too complicated and cannot deal with spurious and lost pulses.

Self-supervised learning[15] has attracted much research interest in Computer Vision(CV) and Natural Language Processing(NLP) in the past few years. By carefully designing the pretext tasks, the trained model can extract the semantically meaningful features and performs images classification [16] or discover linguistic units in sentences[17] or audio signals [18], [19] without labels. Inspired by those works, we establish a framework to automatically analyze the intercepted MFR pulse trains. Through iterative segmenting and clustering in a self-supervised way, the hierarchical structure of MFR pulse trains can be revealed when only the raw sequences of pulse descriptor words (PDW) are given. Simulation results show that

> T-SP-28779-2022

our methods can accurately detect the state transition boundary, extracting the radar words as well as the work modes without using any labeled data, under agile and heavily overlapped pulse parameters.

The originality and novelties of this paper are mainly on three aspects. First, this method does not require prior information that most of the existing approaches heavily depend on. Second, through contrastive self-supervise learning, the model can disentangle the overlapped pulse parameters, projecting them into clustering-friendly spaces. Third, this method can perform segmenting and clustering of the intercepted MFR signals at the same time while currently, no existing approaches can.

The rest of the paper is organized as follows. Section 2 describes the MFR signal received by distributed sensor networks as well as the classical MFR hierarchical models. Section 3 explains the proposed hierarchical Self-supervised method as well as the network architecture for MFR behavior state analysis. In section 4, the simulation based on a hypothetical MFR under different kinds of no-ideal situations is carried out to test the proposed method. And finally, section 5 summarizes the whole paper.

II. HIERARCHICAL MODELS FOR MFR SIGNAL RECEIVED BY DISTRIBUTED SENSORS

A. OBSERVATION MODEL OF MFR SIGNALS

Due to the use of phased array antennas and digital control systems, MFRs can quickly change the beam steering and the pulse to pulse parameters in real-time. For a single noncooperative receiver, this results in diverse signal amplitudes and agile waveforms. In such a case, some of the pulse groups are lost, resulting in discontinuity for the received pulse train. Some MFR may send the same pulse groups at different work modes even for some simple radars like the Mercury[6]. Thus, it is insufficient to recognize the states using a single pulse group, the context information between the pulse groups must be considered.

To overcome the problem of pulses missing in groups, this paper discusses the recognition problem under the framework of distributed sensors to ensure the high probability of receiving, as shown in Fig. 1. At each beam position, there is a sensor(drone) within the beam width to ensure a high probability of receiving. The received signals for the sensor networks are present in Fig. 2. For the distributed sensors systems, the fusion center performs the data association of the pulse trains received by individual sensors and recover the MFR pulse train. Since this paper focuses on MFR states recognition, we will not discuss the association process in detail but do take the association errors into account in the simulation section.

After the recovery of the MFR pulse train, parameter estimation can be carried out to output a sequence of PDWs in traditional manners. In this article, the PDWs are treated as the basic unit for MFR state recognition.

Fig. 1 Multifunction radar and the distributed sensors

Fig. 2 MFR signal received by the distributed sensors

B. MFR HIERARCHICAL SIGNAL STRUCTURES

Signal parameters of MFRs usually vary in wide ranges, therefore, it can be difficult to perform recognition with statistical features on a single pulse. To better represent MFR pulse trains, we adopt the hierarchical structure to describe MFR temporal patterns[20]. In the hierarchical model(Fig. 3), MFR signals are represented by radar pulses, radar words, and radar phrases, sometimes, a deeper representation including ‘Syllables’, ‘Commands’ may also be possible[8] for more complex radar systems. To simplify the discussion, we only use a 3-level representation though our method can be easily generalized to radar systems with any number of levels.

To facilitate further discussions, we formally defined some terms as follows.

Radar pulse: A single PDW with parameters measured by the noncooperative receiver.

Radar word: A limited number of PDWs that can be transmitted and received as a whole to complete a single radar detection

Radar phrase: Composed of a limited number of radar words that can complete a radar task or represent a certain working mode.

> T-SP-28779-2022

Working mode: Predefined functions of MFR such as search and track, consists of one or more radar phrases

States: Including working mode and all the hierarchical representations such as radar words, radar phrases.

Symbol level: All the layers with discrete representations in the hierarchical model, like radar words and radar phrases.

Fig. 3 Typical 3-level hierarchical representation of MFR signals

Depending on the current working mode and target state, the MFR resource management agent[3] will decide the working mode in the next timing window according to the states of the observation target, resulting in a Markov decision process(MDP)[21] for the received signals. For the noncooperative receiver, we cannot get the information about the radar target state therefore, the working mode sequence can be simply viewed as a Markov Chain, resulting in a Hidden Markov Model(HMM) [22] for states at each level. In the simulation section, for simplicity, we generated MFR pulse sequences according to a predefined HMM, even our methods can process more complicated generative models.

III. SELF-SUPERVISED MFR STATES SEGMENTATION AND CLUSTERING

A. SELF-SUPERVISED RADAR PULSES SEGMENTATION

To perform further analysis based on the MFR hierarchical model, the first step is to segment the analog-valued Radar pulses according to radar words. If there is a silence period between each radar word, the segment could be done by using the time of arrival (TOA), however, this does not hold in general as MFR can change its state instantaneously. Thus, a transition state detector must be established. In this section, a convolutional neural network (CNN) with the noise-contrastive estimation loss[23] is established to detect the boundary of each radar word by calculating the similarity between adjacent CNN output vectors.

Suppose after data association, a sequence of Radar pulses has been collected, denoted by

$$P = (pdw_1, \dots, pdw_n) \quad (1)$$

and

$$pdw_i = \begin{bmatrix} PRI \\ PW \\ \vdots \end{bmatrix} \quad (2)$$

where n is the total number of pulses, PRI, PW is short for pulse repetitive interval and pulse width respectively. The ellipsis refers to other measured parameters such as the carrier frequency.

We learn an encoder to project the pulse sequence into a latent embedding L

$$L = \{l_1, l_2, l_3, \dots, l_m\} = E(P) \quad (3)$$

where E is the CNN encoder.

The CNN encoder is trained in a self-supervised way [18], [24], as demonstrated in Fig. 4. Firstly, N negative samples are created by randomly permutating the output feature map N times. Secondly, the noise-contrastive estimation loss is calculated by using the negative samples and the original positive sample which is shown in Eq(4). In Eq(4), $\frac{l_i^T s_{j+1}}{\|l_i\| \|s_{j+1}\|}$ is the cosine similarity between latent vector l_i and l_j , which will later be used as a test statistic for pulse segmentation. The parameters of the encoder are updated according to Eq(5) via backpropagation[25], where $para$ refers to all the trainable weights in the encoder and η is the learning rate that can be tuned. By minimizing the loss function, the model is forced to output high similarity scores for the adjacent latent vector pairs while producing low similarity scores between non-adjacent ones.

To prevent the model overfit the dataset, we limit the CNN encoder receptive field so that the model can only focus on parts of the input sequence when producing one latent vector, in this way, the model will produce low similarity scores at the transition boundaries of radar states, acting as a test statistic for states segmentation, as demonstrated in Fig. 5.

Fig. 4 self-supervised learning with the CNN encoder

$$\hat{\mathcal{L}} = -\frac{1}{m-1} \sum_i \log \frac{e^{\left(\frac{l_i^T s_{i+1}}{\|l_i\| \|s_{i+1}\|} \right)}}{\sum_j e^{\left(\frac{l_i^T s_j}{\|l_i\| \|s_j\|} \right)} + e^{\left(\frac{l_i^T s_{i+1}}{\|l_i\| \|s_{i+1}\|} \right)}}; \quad (|i-j| > 1) \quad (4)$$

$$para_{new} = para_{old} - \eta \frac{\partial \hat{\mathcal{L}}}{\partial para_{old}} \quad (5)$$

> T-SP-28779-2022

Fig. 5 An exemplar latent vector produced by the CNN encoder with a receptive field of 3. when the receptive field is within one radar word (in the picture, l_1 l_2 l_3 l_4 and l_7 l_8), the adjacent latent vector is likely to produce high similarly scores, since the inputs have similar patterns. While at the boundary of two states (l_5 l_6), the inputs differ a lot, the model will be less ‘confident’, producing an abnormally high loss.

With the trained model, the similarly scores of adjacent latent vectors could be used to approximate the state transition points in the original pulse sequence, by using the expressions below.

$$T_i = I - \frac{l_i^T * l_{i+1}}{\|l_i\| * \|l_{i+1}\|} \quad (6)$$

$$pos_l = \{i \mid T_i \geq th\} \quad (7)$$

$$pos = pos_l * \frac{n}{m-1} \quad (8)$$

where T_i is the latent vector boundary test statistic, pos_l is the latent vector boundary positions, th is a preset threshold, and pos contains the final estimated state transition positions.

A segmentation example is shown in

(b)

Fig. 6(b) . The test statistic T_i has been up-sampled to the same length as the original pulse sequence. As can be seen in the figure, at each moment of radar word transition, there is a peak in T_i , which correctly detects the boundary of different radar words.

(b)

Fig. 6 A segmentation example. (a) A particular MFR Pulse sequence with 10% pulse missing. Here, we only plot the pulse repetition interval. (b)The test statistic, presetted threshold, and the radar words for this MFR Pulse sequence.

B. K-MEANS++-BASED LATENT SPACE CLUSTERING FOR RADAR WORDS EXTRACTION

In the previous section, the MFR Pulse sequences have been segmented into subsegments with the same radar word using the test statistic T_i , the next task is to cluster them into different groups so that each group corresponds to one radar word. The normal PDWs clustering methods perform pool in such task since different radar words may have seriously overlapped parameters.

We introduce the idea of latent space clustering, in which we use the latent vectors produced by the CNN encoder for clustering as each latent vector contains contextual information of radar pulses within the receptive field. By minimizing the loss(Eq.(4)) in a self-supervised way, the encoder will pull latent vectors that belong to the same state close to each other while pushing latent vectors in a different state away in terms of the cosine similarity. In theory, any latent vector within one subsegment can be chosen to represent the whole, however, the actual received pulse trains are likely to be corrupted by pulse missing and measurement noise, making the latent vectors noisy. To gain a more robust feature representation and eliminate the boundary effect, simply take the average of all latent vectors for one subsegment except those at the boundary to produce one feature vector r_s .

$$r_s = \frac{1}{S-2} \sum_{i=2}^{S-1} l_i \quad (9)$$

where S is the total number of latent vectors for subsegment s , l_i is the i th latent vector, a demonstration example is in Fig. 7.

(a)

> T-SP-28779-2022

Fig. 7 A demonstration about feature vector production for radar word W1. $l_1 : l_4$ are at the boundary, thus they are discarded when calculating the feature vector.

With the feature vector, clustering can be performed in terms of the cosine distance. We choose the K-means++ algorithm[26] due to its good interpretability and fast convergence speed, the detailed step is present below.

- Randomly Pick the first centroid from the feature vectors.
- Calculate the cosine distance from the nearest, previously determined centroid for each feature vector.
- Select a new centroid from the feature vector so that the probability of picking a point as centroid is proportional to its cosine distance from the nearest, previously picked centroid.
- Repeat steps b) and c) until k centroids have been sampled
- Performs standard K-means with cosine distance using the previous Initialized k centroids

The number of centroids can simply be determined by the elbow rule[27] with silhouette score[28] as in Fig. 8.

Fig. 8 Elbow rule with silhouette score for determining the number of clusters.

After a successful clustering, each cluster will correspond to one radar word, the next task is to cut each subsegment into

some basic units, or equivalently, the radar words. If all the radar words have a fixed number of pulses like the Mercury radar [6], the number of pulses N_i for each radar word can be estimated by

$$N_i = GCD(q_{i1} + \varepsilon_{i1}, q_{i2} + \varepsilon_{i2}, \dots, q_{in} + \varepsilon_{in}) \quad (10)$$

where GCD represent the greatest common divisor, q_{in} is the number of pulses for the n th subsegment in the i th cluster. A small integer ε_{in} is added for neighbor searching because of the segmentation errors.

If the radar words have a various number of pulses, there is no way for a passive system to perform further segmentation without any prior information.

C. SELF-SUPERVISED LEARNING FOR SYMBOL LEVEL SEGMENTATION AND CLUSTERING

Using the methods in the previous sections, we can convert a radar pulse train into a radar word sequence. Again, by using the self-supervised approach the underlay state transition points could be detected and hopefully, the symbols in the next level of the hierarchical model can be extracted.

To input the discrete tokens, each token was first converted to one-hot vectors $w = [0, \dots, 0, 1, 0, \dots, 0]^T$, where the only nonzero element indicating the symbol class. After that, the word embedding[29] technique was employed to convert each token into an embedding vector, as shown below.

$$\mathbf{e} = \mathbf{E}^* \mathbf{w} \quad (11)$$

where $\mathbf{E}$ is the embedding matrices, $\mathbf{e}$ is the embedding vector.

After embedding, a CNN encoder could be established as in Fig. 9 and the rest of the procedure is the same as described in section A and B.

Fig. 9 symbol level self-supervised learning with the word embedding CNN encoder(E-CNN).

This approach can be repeated for the extracted symbols until there is no detected state change or the symbol sequence is too short. The whole procedure is shown in Fig. 10.

Fig. 10 The iterative self-supervised procedure for extracting symbols in the hierarchical model

First, the radar words are extracted from the raw PDWs using the methods described in the previous section, converting the PDW sequences into radar word sequences. Next, an E-CNN is trained on the word sequences in a self-supervised way to segment and cluster the radar words, each cluster will be assigned with a new symbol, in such a way, the word sequences can be further converted to symbols in the deeper layers of the MFR hierarchical model such as the radar phases. Through this process, all the states of the MFR can be detected and recognized in ideal cases, however, the models can sometimes ‘skip’ some layers in the hierarchical model, which will be shown in the simulation section.

IV. SIMULATIONS

A. SIMULATIONS SETTINGS

We build the hypothetical MFR according to the Mercury radar because lots of the current researches on MFR recognition is based on it[30], [10]. The radar has five functional states namely: search(S), acquisition(A), nonadaptive track(NT), range resolution(RR), and track maintenance(TM). The functional states transition is assumed to be a Markov chain and the resulting radar phases, words sequences follow a hidden Markov model, the radar phases corresponding to each functional state is illustrated in TABLE I.

We generate the nine radar words according to the parameters in TABLE II. Common PRI modulation types are included to increase the agility of the MFR pulse pattern. We only consider PRI and PW as the input for the sake of convenience and data visualization, other parameters such as RF can be easily included by changing the network input dimension. **Fig. 12** is a visualization plot for each radar word, with measurement errors, the parameters overlap with each other, making the extraction task challenging.

Fig. 11[6] Functional states transitions of the hypothetical MFR

TABLE I Radar phrases for each operating mode

Mode	Phrases	Mod	Phrases
Search (S)	$[W_1 W_2 W_4 W_5]$	Track - Maintenance (T)	$[W_1 W_7 W_7 W_7]$
	$[W_2 W_4 W_5 W_1]$		$[W_2 W_7 W_7 W_7]$
	$[W_4 W_5 W_1 W_2]$		$[W_3 W_7 W_7 W_7]$
	$[W_5 W_1 W_2 W_4]$		$[W_4 W_7 W_7 W_7]$
	$[W_1 W_3 W_5 W_1]$		$[W_5 W_7 W_7 W_7]$
	$[W_3 W_5 W_1 W_3]$		$[W_6 W_7 W_7 W_7]$
	$[W_5 W_1 W_3 W_5]$		$[W_1 W_8 W_8 W_8]$
			$[W_2 W_8 W_8 W_8]$
Acquisition (A)	$[W_1 W_1 W_1 W_1]$		$[W_3 W_8 W_8 W_8]$
	$[W_2 W_2 W_2 W_2]$		$[W_4 W_8 W_8 W_8]$
	$[W_3 W_3 W_3 W_3]$		$[W_5 W_8 W_8 W_8]$
	$[W_4 W_4 W_4 W_4]$		$[W_6 W_8 W_8 W_8]$
	$[W_5 W_5 W_5 W_5]$		$[W_1 W_9 W_9 W_9]$
Non-Adaptive Track (N)			$[W_2 W_9 W_9 W_9]$
	$[W_1 W_6 W_6 W_6]$		$[W_3 W_9 W_9 W_9]$
	$[W_2 W_6 W_6 W_6]$		$[W_4 W_9 W_9 W_9]$
	$[W_3 W_6 W_6 W_6]$		$[W_5 W_9 W_9 W_9]$
	$[W_4 W_6 W_6 W_6]$		$[W_6 W_9 W_9 W_9]$
	$[W_5 W_6 W_6 W_6]$		$[W_7 W_7 W_7 W_7]$

> T-SP-28779-2022

Range-Resolution (R)			$[W_8W_8W_8W_8]$ $[W_9W_9W_9W_9]$
	$[W_7W_6W_6W_6]$ $[W_8W_6W_6W_6]$ $[W_8W_6W_6W_6]$ $[W_9W_6W_6W_6]$		

TABLE II Parameters for the Radar words, the PRI modulation type is set according to [31]

Radar words	PRI (us)	PW (us)	Number of pulses
W1	100	3	100
W2	50	3	80
W3	Stagger between [60,115]	5	100
W4	Stagger between [135,205]	5	100
W5	Stepping between [80,120]	5	100
W6	Dwell and Switch between [15,25]	10	100
W7	Dwell and Switch between [75,115]	5	90
W8	Wobulate between [60,100]	8	70
W9	Wobulate between [80,120]	3	60

Fig. 12 Radar words and its corresponding PRI and PW ($\sigma_{PRI} = 5us$, $\sigma_{pw} = 2us$).

B. SIMULATIONS FOR SELF-SUPERVISED RADAR PULSES SEGMENTATION AND CLUSTERING

This simulation tests the segmentation and clustering accuracy of the radar words. The dataset is obtained via simulations, with each containing 100 sequences of length 2000. Unless otherwise specified, each PRI's and PW's observation error is zero-mean and Gaussian, with a standard deviation of $\sigma_{PRI} = 1us$, $\sigma_{pw} = 0.1us$. Since our method is unsupervised, there is no need to include a test set.

Two typical types of corruption are considered namely, missing and spurious pulses, which are mainly caused by the deinterleaving failure. The pulses missing can assume to be uniform while spurious pulses satisfy a Poisson distribution[32]. Besides, we also regard the case when pulses are lost in groups namely, group missing which is caused by strong directivity of the radar antenna and the data association errors of the passive sensors. A sketch of those corruptions is present in Fig. 12.

Fig. 13 Typical types of corruptions for the received pulse trains.

A 5-layer lighting encoder is built for radar pulses feature extraction, the hyperparameters for each layer are shown in TABLE III. We intentionally limit the kernel size to 2 and the stride for the first two layers is set to be 2 to suppress the noise in the test statistic. The network is implemented with PyTorch[33] and optimized with Adam[34] with a learning rate of 0.001 and a batch size of 100. For each positive training data, we create 5 negative samples per positive sample ($N=5$) to estimate the contrast loss(Eq(4)), the training convergences after 100 epochs.

TABLE III Network hyperparameters for self-supervised radar pulses segmentation

Layers number	Layer type	Number of filters	Kernel size	Stride	Output dimension
1	convolution	64	2	2	64
2	convolution	64	2	2	64
3	convolution	64	2	1	64

> T-SP-28779-2022

4	convolution	64	1	1	64
5	Fully connected	--	--	--	8

We first evaluate the model's capacity for pulse segmentation, refer to the metrics in automatic speech segmentation[35], place a fixed-size search-region around each reference boundary, and verify whether the segmentation algorithm has produced the correct number of boundaries within these regions. If the algorithm produces more than one boundary in that region, label it as over-segmentation and if no output is produced, label it as under-segmentation. Based on that, some metrics are defined as follows, the search-region size is set to be ± 20 pulses and the threshold for detection is set to be 0.15, subsegments with less than 10 pulses will be thrown away as the latent vectors are too short for clustering.

1) over-segmentation error

$$e_{over} = \frac{\text{num of over segmented boundary}}{\text{total num of reference boundary}} \quad (12)$$

2) under-segmentation error

$$e_{under} = \frac{\text{num of under segmented boundary}}{\text{total num of reference boundary}} \quad (13)$$

3) pulses keeping rate

$$kp = \frac{\text{num of pulse in the segmented pulse groups}}{\text{num of pulse in the original sequence}} \quad (14)$$

This metric measures the pulse utilization rate.

After clustering, the latent space clustering can be performed on the segmented pulses. To measure the clustering accuracy, first, a radar word label is assigned to each subsegment as a ground truth. If one subsegment contains only one word, there is no ambiguity. However, when over-segmentation or under-segmentation occurs, one subsegment may contain multiple words, in such cases, the ground-truth classes are assigned according to the longest words in that subsegment. Next, the ground-truth classes are assigned to each cluster. If the number of detected clusters matches the number of ground-truth classes, the labels for each cluster are assigned using a Hungarian matching algorithm[16]. When over-clustering(the number of clusters > the number of ground-truth classes) or under-clustering(the number of clusters < the number of ground-truth classes) occurs, simply assign each cluster with a label according to the majority of ground-truth classes in that cluster. With the assigned labels, the clustering accuracy $acc_{cluster}$ can be easily calculated.

TABLE IV Analysis of segmentation and clustering performance on different types of corruptions

Uniform missing rate %	0	10	20	30
------------------------	---	----	----	----

e_{over} %	0	14.56	10.14	6.60
e_{under} %	0	0	1.41	1.73
kp %	97.69	97.58	97.12	97.18
$acc_{cluster}$ %	100	100	100	100
Ratio of spurious pulses %	5	10	20	30
e_{over} %	8.10	8.00	5.26	2.64
e_{under} %	1.65	4.06	14.34	23.47
kp %	97.12	98.11	97.73	98.50
$acc_{cluster}$ %	100	99.99	99.23	84.83 (detect only 7 clusters)
Pulse group missing rate %	5	10	20	30
e_{over} %	4.11	18.61	26.89	26.02
e_{under} %	0	0	0	3.11
kp %	97.61	97.29	96.38	95.89
$acc_{cluster}$ %	100	100	100	100
Observation errors %	$\sigma_{PRI} = 1\mu s$ $\sigma_{pw} = 0.1\mu s$	$\sigma_{PRI} = 2\mu s$ $\sigma_{pw} = 0.3\mu s$	$\sigma_{PRI} = 3\mu s$ $\sigma_{pw} = 1\mu s$	$\sigma_{PRI} = 5\mu s$ $\sigma_{pw} = 2\mu s$
e_{over} %	0	0	0	3.70
e_{under} %	0	0.10	0	3.11
kp %	97.69	97.52	96.66	95.51
$acc_{cluster}$ %	100	100	100	94.89 (detect 11 clusters)

TABLE IV shows the quantitative results for the pulse segmentation and clustering task, in most cases, the algorithm can accurately detect the transition boundary of each radar word and keep most pulses after segmentation, as the model can make use of the contextual information to deal with the corruptions. Noticeably, the spurious pulses cause a higher under-segmentation error. This is reasonable since it is hard for the model to distinguish the true transition boundary when there are several spurious pulses around it. Furthermore, it is observed that when more pulses are missed in groups, the over-segmentation error rises as the model wrongly detects the edge of missing groups as the transition boundary. For radar words clustering, two types of pulse missing have less impact on the clustering accuracy. Even if there are some over and under segmented pulses, the algorithm correctly detects the number of clusters and reaches nearly 100% clustering accuracy. The 2-D TSNE embeddings [36] results for the lantern vectors under

> T-SP-28779-2022

20% uniform or group missing are shown in **Fig. 14(a),(b)**, respectively. It can be viewed that 9 radar words form 9 distinct clusters with no overlapping, explaining the high quality of clustering. Under high observation errors, the model split W6 into multiple groups (**Fig. 14(c)**), causing over-clustering. On the contrary, a high ratio of spurious pulses results in under-clustering, as the model project W5, W3; W1, W9 close to each other in the lantern space (**Fig. 14(d)**).

Fig. 14 2-D TSNE plot for the latent vectors under various types of corruptions. Individual points in each plot are colored according to the corresponding radar words labels.

C. SIMULATIONS FOR SELF-SUPERVISED WORK MODE SEGMENTATION AND CLUSTERING

In this section, we assume the radar words have been successfully extracted by the method described in the previous section. Considering the data imbalance problem of the MFR sequence[13], the HMM transition matrix for data generation is carefully designed so that the MFR has a higher probability to stay in the search state. A dataset containing 100 radar words sequences of length 2000 is generated for training and testing, one sample is shown in **Fig. 15**.

A 6-layer E-CNN with an embedding layer is built and the hyperparameters are shown in TABLE V. The kernel size for each CNN layer is the same as the encoder for processing radar pulses, except for the stride. This time, the stride is set to be 1 to produce a more accurate edge detection as some of the work modes only last for a short time (see **Fig. 15(b)**). The optimization procedure is the same as the encoder for radar pulses.

The receptive field for the E-CNN is of size 8, which is greater than the length of one radar phase. Therefore, it is impossible to detect the transition boundary for the radar phases. We have tried to limit the receptive field down to 4 or 2, however, the model does not converge due to limited temporal information. So, we leave the extraction of such short units to future research.

TABLE V Network hyperparameters for the E-CNN used for self-supervised radar pulses segmentation

Layers number	Layer type	Number of filters	Kernel size	Stride	Output dimension	Dictionary size
1	Embedding	--	--	--	8	10
2	convolution	64	2	1	64	--
3	convolution	64	2	1	64	--
4	convolution	64	2	1	64	--
5	convolution	64	1	1	64	--
6	Fully connected	--	--	--	8	--

Fig. 15 One sample radar words sequence and its corresponding work modes, work mode 1 to 5, one each for search, acquisition, nonadaptive track, range resolution, and track maintenance.

Referring to [9], 4 types of corruption are applied to the dataset and the results are listed in TABLE VI. The search-region size is set to be ± 5 words and the threshold for detection is set to be 0.2, subsegments with less than 10 words are discarded.

TABLE VI Analysis of work mode segmentation and clustering performance on different types of corruptions, all the metrics in the table have the same meaning as in the previous section.

Missing %	0	5	10	20
$e_{over} \%$	14.86	24.30	31.25	44.49
$e_{under} \%$	6.63	16.57	17.07	16.34
$kp \%$	99.50	96.57	93.12	84.77
$acc_{cluster} \%$	98.86	97.51	97.11	95.07
Missing in blocks %	5	10	20	30
$e_{over} \%$	15.73	18.90	24.76	22.21

> T-SP-28779-2022

$e_{under} \%$	7.42	9.22	9.61	13.04
$kp \%$	99.36	98.79	96.82	96.95
$acc_{cluster} \%$	98.13	97.79	97.70	96.90
Additional %	5	10	15	20
$e_{over} \%$	27.15	47.06	65.36	70.59
$e_{under} \%$	5.31	5.48	5.70	3.05
$kp \%$	95.20	85.97	68.88	72.11
$acc_{cluster} \%$	98.31	98.63	97.66	97.59
Additional in blocks	5	10	20	30
$e_{over} \%$	13.25	17.06	22.15	21.42
$e_{under} \%$	7.43	11.21	10.67	13.45
$kp \%$	99.47	98.54	98.41	98.00
$acc_{cluster} \%$	97.46	98.51	96.97	96.63

Fig. 16 2-D TSNE plot for the latent vectors under various types of corruptions. Individual points in each plot are colored according to the corresponding work mode labels.

The results show that missing or additional in blocks causes less harm than the single ones, which is indicated by the over-segmentation, under-segmentation error, as well as the pulse keeping rate. Notice, the segmentation performance is generally worse than that in radar pulses when compared with the results in TABLE IV. This is because each radar word consists of a fixed number of radar pulses while the work modes are driven by a stochastic model. Interestingly, those corruptions seem to have little damage to the clustering accuracy as the proposed algorithm correctly identifies the number of clusters, reaching more than 95% accuracy in all cases. When looking at the TSNE plot (Fig. 16), work modes are quite well separated even the words sequences is heavily corrupted.

D. SELF-SUPERVISED SEMI-SUPERVISED LEARNING

So far, the segmentation and clustering performance of the encoders learned in a self-supervised way has been tested under various conditions, in this section, we further explore the capability of the trained E-CNN for semi-supervised learning, as self-supervised learning can boost the performance on the downstream tasks[37]. We train several sequence-to-sequence work modes classifiers for benchmarking. A dataset containing 250 radar words sequences of length 2000 with 10% missing, additional, missing in blocks, additional in blocks is generated and 80% of it is used for training, the rest is used for testing. The self-supervised models are pretrained on the whole training set while the parameters of the baseline models are randomly initialized.

Fig. 17 Semi-supervised work modes recognition accuracy for different models under a different number of labeled samples. The recurrent network (RNN) based method[10], [13] is added for comparison. The GRU-SSL is pretrained with ELMO-based methods[38] and the E-CNN-SSL is pretrained using the proposed contrastive loss.

Fig. 17 shows the performance for different models with or without self-supervised training. The RNN-based model generally performs better than E-CNN as it can consider the whole sequence when producing the output, whereas the E-CNN can only focus on the information locally. This property makes RNN a good classifier but not a good segmenter as it can accurately predict the transition boundary with high confidence.

Under a limited number of labeled samples, self-supervised pretraining can effectively increase the recognition accuracy, however, as the number of labeled samples increase, the benefit disappears, the models even end up with worse performance. This may be due to the small scale of the dataset, as researchers in NLP use millions of sentences[38] for pretraining.

IV. CONCLUSION

This paper introduces Self-supervised learning to solve the problems of MFR behavior states recognition. By minimizing the contrastive loss in a self-supervised way, the encoder can learn good representations of the original sequences and produce representative latent vectors for each state. Using the

> T-SP-28779-2022

similarity between adjacent latent vectors, MFR state transition moment can be detected, and therefore, the original pulse sequences can be segmented. With the segmented sequences, the latent vectors can be further clustered into different groups, each of which represents distinguish states. This segment-clustering process can be conducted iteratively, accomplishing the analysis of MFR states. Our method is fully unsupervised and robust to erroneously received pulses thus is perfect for Non-cooperative applications. This method can also be extended to other areas with the discrete event systems as they also involve the hierarchical structure[39] as the MFR.

However, there are some limitations of the proposed method and further investigations are required. Firstly, the proposed method works in an offline manner, when facing newly arrived pulses that belong to different MFR states, the whole models need to be re-trained on the enlarged dataset, which is inefficient. So further study can focus on the MFR states recognition with incremental learning. Secondly, this study was only carried out on a simulated dataset, further researches are yet necessary to test the proposed method in the real EW environment with more complicated MFRs.

REFERENCES AND FOOTNOTES

A. References

- [1] A. J. Orman, C. N. Potts, A. K. Shahani, and A. R. Moore, "Scheduling for a multifunction phased array radar system," *European Journal of Operational Research*, vol. 90, no. 1, pp. 13–25, Apr. 1996, doi: 10/cc7f87.
- [2] A. Wang and V. Krishnamurthy, "Signal Interpretation of Multifunction Radars: Modeling and Statistical Signal Processing With Stochastic Context Free Grammar," *IEEE Trans. Signal Process.*, vol. 56, no. 3, pp. 1106–1119, Mar. 2008, doi: 10/fp6jng.
- [3] R. Klemm, U. Nickel, C. Gierull, P. Lombardo, H. Griffiths, and W. Koch, Eds., "Array radar resource management," in *Novel Radar Techniques and Applications Volume 1: Real Aperture Array Radar, Imaging Radar, and Passive and Multistatic Radar*, Institution of Engineering and Technology, 2017, pp. 135–171. doi: 10.1049/SBRA512F_ch3.
- [4] A. B. Charlish, "Autonomous agents for multi-function radar resource management."
- [5] L. Dai, B. Wang, B. Cai, and J. Liu, "A method for states estimation of multi-function radar based on stochastic context free grammar," *Journal of Air Force Engineering University*, vol. 15, no. 3, pp. 31–39, 2014.
- [6] N. A. Visnevski, "Syntactic modeling of multi-function radars," 2005.
- [7] Y. Li, M. Zhu, Y. Ma, and J. Yang, "Work modes recognition and boundary identification of MFR pulse sequences with a hierarchical seq2seq LSTM," *IET Radar, Sonar & Navigation*, vol. 14, no. 9, pp. 1343–1353, Sep. 2020, doi: 10.1049/iet-rsn.2020.0060.
- [8] S. Apfeld, A. Charlish, and G. Ascheid, "Modelling, Learning and Prediction of Complex Radar Emitter Behaviour," in *2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA)*, Dec. 2019, pp. 305–310. doi: 10.1109/ICMLA.2019.00057.
- [9] S. Apfeld, A. Charlish, and G. Ascheid, "Identification of Radar Emitter Type with Recurrent Neural Networks," in *2020 Sensor Signal Processing for Defence Conference (SSPD)*, Edinburgh, 2020, pp. 1–5. doi: 10.1109/sspd47486.2020.9271988.
- [10] X. Xu, D. Bi, and J. Pan, "Method for functional state recognition of multifunction radars based on recurrent neural networks," *IET Radar, Sonar & Navigation*, vol. 15, no. 7, pp. 724–732, Jul. 2021, doi: 10/gjw75z.
- [11] X. Dong and S. Cheng, "Radar Working Modes Recognition Based on Discrete Process Neural Network," *IOP Conf. Ser.: Mater. Sci. Eng.*, vol. 394, p. 042088, Aug. 2018, doi: 10/gncw7.
- [12] S. Apfeld, A. Charlish, and G. Ascheid, "The Value of Memory: Markov Chain versus Long Short-Term Memory for Electronic Intelligence," in *2021 IEEE Radar Conference (RadarConf21)*, May 2021, pp. 1–6. doi: 10/gncfwn.
- [13] Z.-M. Liu, "Recognition of Multifunction Radars Via Hierarchically Mining and Exploiting Pulse Group Patterns," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 56, no. 6, pp. 4659–4672, Dec. 2020, doi: 10/gmcmqx.
- [14] M. Zhu, Y. Li, and S. Wang, "Model-Based Time Series Clustering and Interpulse Modulation Parameter Estimation of Multifunction Radar Pulse Sequences," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 57, no. 6, pp. 3673–3690, Dec. 2021, doi: 10/gnp8r5.
- [15] L. Jing and Y. Tian, "Self-Supervised Visual Feature Learning With Deep Neural Networks: A Survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 11, pp. 4037–4058, Nov. 2021, doi: 10/gg8fm7.
- [16] W. Van Gansbeke, S. Vandenhende, S. Georgoulis, M. Proesmans, and L. Van Gool, "SCAN: Learning to Classify Images without Labels," *arXiv:2005.12320 [cs]*, Jul. 2020, Accessed: Dec. 19, 2021. [Online]. Available: <http://arxiv.org/abs/2005.12320>
- [17] L. Wang, Z. Li, and X. Zheng, "Unsupervised Word Segmentation with Bi-directional Neural Language Model," *arXiv:2103.01421 [cs]*, Mar. 2021, Accessed: Jul. 21, 2021. [Online]. Available: <http://arxiv.org/abs/2103.01421>
- [18] F. Kreuk, J. Keshet, and Y. Adi, "Self-Supervised Contrastive Learning for Unsupervised Phoneme Segmentation," Jul. 2020, Accessed: Nov. 10, 2021. [Online]. Available: <https://arxiv.org/abs/2007.13465v2>
- [19] A. T. Liu, P. Hsu, and H. Lee, "Unsupervised End-to-End Learning of Discrete Linguistic Units for Voice Conversion," *Interspeech 2019*, pp. 1108–1112, Sep. 2019, doi: 10/gm95dt.
- [20] N. Visnevski, V. Krishnamurthy, S. Haykin, B. Currie, F. Dilkes, and P. Lavoie, "Multi-function radar emitter modelling: a stochastic discrete event system approach," in *42nd IEEE International Conference on Decision and Control (IEEE Cat. No.03CH37475)*, Dec. 2003, vol. 6, pp. 6295–6300 Vol.6. doi: 10.1109/CDC.2003.1272306.
- [21] R. BELLMAN, "A Markovian Decision Process," *Journal of Mathematics and Mechanics*, vol. 6, no. 5, pp. 679–684, 1957.

> T-SP-28779-2022

- [22] N. Visnevski, S. Haykin, V. Krishnamurthy, F. A. Dilkes, and P. Lavoie, "Hidden Markov models for radar pulse train analysis in electronic warfare," in *Proceedings. (ICASSP '05). IEEE International Conference on Acoustics, Speech, and Signal Processing, 2005.*, Mar. 2005, vol. 5, p. v/597-v/600 Vol. 5. doi: 10/dqt6bv.
- [23] M. Gutmann and A. Hyvarinen, "Noise-contrastive estimation: A new estimation principle for unnormalized statistical models," p. 8.
- [24] S. Schneider, A. Baevski, R. Collobert, and M. Auli, "wav2vec: Unsupervised Pre-training for Speech Recognition," *arXiv:1904.05862 [cs]*, Sep. 2019, Accessed: Dec. 03, 2021. [Online]. Available: <http://arxiv.org/abs/1904.05862>
- [25] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," p. 4, 1986, doi: 10/cvjdpk.
- [26] D. Arthur and S. Vassilvitskii, "k-means++: The Advantages of Careful Seeding," p. 11.
- [27] C. Yuan and H. Yang, "Research on K-Value Selection Method of K-Means Clustering Algorithm," *J*, vol. 2, no. 2, pp. 226–235, Jun. 2019, doi: 10/ghcr9m.
- [28] P. J. Rousseeuw, "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis," *Journal of Computational and Applied Mathematics*, vol. 20, pp. 53–65, Nov. 1987, doi: 10/fdxwqh.
- [29] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient Estimation of Word Representations in Vector Space," *arXiv:1301.3781 [cs]*, Sep. 2013, Accessed: Sep. 25, 2021. [Online]. Available: <http://arxiv.org/abs/1301.3781>
- [30] J. Ou, Y. Chen, F. Zhao, J. Liu, and S. Xiao, "Novel Approach for the Recognition and Prediction of Multi-Function Radar Behaviours Based on Predictive State Representations," *Sensors*, vol. 17, no. 3, Art. no. 3, Mar. 2017, doi: 10/ghbj9h.
- [31] R. G. Wiley, *ELINT: The interception and analysis of radar signals*. Artech House, 2006.
- [32] X. Li, Z. Liu, and Z. Huang, "Attention-based radar PRI modulation recognition with recurrent neural networks," *IEEE Access*, vol. 8, pp. 57426–57436, 2020, doi: 10.1109/ACCESS.2020.2982654.
- [33] A. Paszke *et al.*, "PyTorch: An Imperative Style, High-Performance Deep Learning Library," in *Advances in Neural Information Processing Systems*, 2019, vol. 32. Accessed: Dec. 08, 2021. [Online]. Available: <https://papers.nips.cc/paper/2019/hash/bdbca288fee7f92f2bfa9f7012727740-Abstract.html>
- [34] D. Kingma and J. Ba, *Adam: A method for stochastic optimization*. 2014.
- [35] O. J. Räsänen, U. K. Laine, and T. Alotaar, "An Improved Speech Segmentation Quality Measure: the R-value," p. 4, doi: 10/gnr9vx.
- [36] G. Hinton and S. Roweis, "Stochastic Neighbor Embedding," p. 8.
- [37] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," *arXiv:1810.04805 [cs]*, May 2019, Accessed: Dec. 20, 2021. [Online]. Available: <http://arxiv.org/abs/1810.04805>
- [38] M. E. Peters *et al.*, "Deep contextualized word representations," *arXiv:1802.05365 [cs]*, Mar. 2018, Accessed: Dec. 26, 2021. [Online]. Available: <http://arxiv.org/abs/1802.05365>
- [39] C. G. Cassandras and S. Lafortune, *Introduction to discrete event systems*, 2. ed., corr. At 2. print. New York, NY: Springer, 2010.