



Degree Project in Computer Science and Engineering, specializing in Machine Learning

Second cycle, 30 credits

Incremental Learning Based Unknown Drone Classification

Open-Set Recognition, Clustering, and Incremental Learning for
UAV RF Signal Identification

JULIE LIU

Incremental Learning Based Unknown Drone Classification

**Open-Set Recognition, Clustering, and Incremental
Learning for UAV RF Signal Identification**

JULIE LIU

Master's Programme, Machine Learning, 120 credits

Date: December 4, 2025

Supervisors: Irshad A. Meer, Mustafa Özger

Examiner: Cicek Cavdar.

School of Electrical Engineering and Computer Science

Swedish title: Incremental Learning Based Unknown Drone Classification

Swedish subtitle: Öppen signaligenkänning, klusterbildning och stegvis inläring för
identifiering av RF-signaler för drönare

Abstract

Unmanned aerial vehicles (UAVs) are increasingly used in civilian and military domains, raising concerns regarding security, privacy, and airspace safety. Reliable detection and recognition of UAVs are therefore critical. Among various sensing modalities, machine learning with radio frequency (RF) signals offers a low-cost and robust solution, capable of operating effectively even under adverse environmental conditions. However, most existing RF-based recognition systems assume a closed-set scenario, where all UAV types are known during training. This assumption is unrealistic in practice, as novel and unknown UAVs frequently emerge. As a result, conventional classifiers suffer from overconfident misclassifications, retraining from scratch is inefficient, and incremental learning approaches face catastrophic forgetting when adapting to new classes.

This thesis proposes a unified framework that addresses these challenges by combining open-set recognition, clustering-based novel class discovery, and incremental learning with memory replay. First, a signal-semantic open-set recognition method separates unknown RF signals from known classes. Next, novel UAV categories are discovered through clustering with automatic cluster number selection using K-Means and Gaussian Mixture Models (GMM), supported by loss functions that enforce intra-class compactness and inter-class separability. Finally, an incremental learning module incorporates these new categories into the model while mitigating catastrophic forgetting via a replay strategy with minimal storage overhead.

The framework is validated on a real-world dataset comprising 18 known UAV classes and 6 unknown classes. Experiments demonstrate that the proposed approach achieves clear separation of unknown classes in semantic space, robust cluster validity under noisy conditions, and effective incremental learning with minimal sample replay. Overall, the results establish the framework as a practical and scalable solution for RF-based UAV recognition in dynamic and open-world environments.

Keywords

Class Incremental Learning (CIL), Exemplar Replay (Exemplar Budget), Novel Category Discovery (NCD), Pseudo Labeling, Unsupervised Clustering, Catastrophic Forgetting

Sammanfattning

Obemannade luftfarkoster (UAV) används i allt större utsträckning inom civila och militära områden, vilket väcker oro kring säkerhet, integritet och luftrumssäkerhet. Tillförlitlig detektering och igenkänning av UAV:er är därför avgörande. Bland olika sensormodaliteter erbjuder maskininlärning med radiofrekvenssignaler (RF) en lågkostnads- och robust lösning som kan fungera effektivt även under ogynnsamma miljöförhållanden. De flesta befintliga RF-baserade igenkänningssystem antar dock ett slutet scenario, där alla UAV-typer är kända under träning. Detta antagande är orealistiskt i praktiken, eftersom nya och okända UAV:er ofta dyker upp. Som ett resultat lider konventionella klassificerare av överdrivna felklassificeringar, omskolning från grunden är ineffektivt och stegvisa inlärningsmetoder möter katastrofala glömskande när de anpassar sig till nya klasser.

Denna avhandling föreslår ett enhetligt ramverk som hanterar dessa utmaningar genom att kombinera öppen uppsättning igenkänning, klusterbaserad upptäckt av nya klasser och stegvis inlärning med minnesåterspelning. Först separerar en signalsemantisk öppen uppsättning igenkänningsmetod okända RF-signaler från kända klasser. Därefter upptäcks nya UAV-kategorier genom klusterbildning med automatiskt klusternummerval med hjälp av K-Means och Gaussian Mixture Models (GMM), med stöd av förlustfunktioner som framtvingar kompakthet inom klasser och separerbarhet mellan klasser. Slutligen införlivar en inkrementell inlärningsmodul dessa nya kategorier i modellen samtidigt som katastrofal glömska minskas via en återuppspelningsstrategi med minimal lagringsoverhead.

Ramverket valideras på en verklig datauppsättning bestående av 18 kända UAV-klasser och 6 okända klasser. Experiment visar att den föreslagna metoden uppnår tydlig separation av okända klasser i semantiskt rum, robust klustervaliditet under bullriga förhållanden och effektiv inkrementell inlärning med minimal återuppspelning av prover. Sammantaget etablerar resultaten ramverket som en praktisk och skalbar lösning för RF-baserad UAV-igenkänning i dynamiska och öppna miljöer.

Nyckelord

klass-inkrementellt lärande (CIL), exemplar-replay (exemplarbudget), upptäckt av nya kategorier (NCD), pseudomärkning, osuperviserad klustring,

katastrofal glömska

Acknowledgments

I would like to express my deepest gratitude to my supervisor, Irshad, and co-supervisor, Mustafa, for the continuous guidance, encouragement, and invaluable feedback throughout the course of this thesis. Their expertise and patience have been fundamental to the completion of this work.

I am also sincerely thankful to my examiner, Cicek, for her constructive comments and valuable support. In particular, I am grateful for the provision of computing resources and server access, which made it possible to conduct the experiments required for this research.

Finally, I wish to extend my heartfelt appreciation to my family and loved ones for their unconditional love, patience, and support. Their encouragement has been a constant source of strength and motivation throughout this journey.

Contents

1	Introduction	1
1.1	Problem	2
1.2	Goals	2
1.3	Research Methodology	2
1.4	Delimitations	3
1.5	Ethics and Sustainability	3
1.6	Structure of the thesis	4
2	Background	5
2.1	Unmanned Aerial Vehicles	5
2.2	Unmanned Aerial Vehicles	6
2.3	Radio Frequency Signals	7
2.4	Machine Learning	10
2.4.1	Closed-Set Recognition	10
2.4.2	Open-Set Recognition	10
2.4.3	Clustering	11
2.4.4	Incremental Learnin	11
2.5	Related Work	14
2.5.1	Drone Recognition	14
2.5.2	Closed-Set Radio Frequency Classification	14
2.5.3	Open-Set Radio Frequency Classification	16
2.5.4	Clustering	17
2.5.5	Incremental Learning	17
2.5.6	Summary	19
3	System Model & Problem Formulation	21
3.1	Model Selection and Justification	21
3.2	Time Frequency Spectrum Feature Construction	25
3.3	Semantic Construction and Optimization	26

3.3.1	Texture Extractor (TE)	26
3.3.2	Position Extractor (PE)	27
3.3.3	Fusion	28
3.3.4	Semantic Optimization	29
3.4	Open-Set Recognition	30
3.5	Clustering	31
3.5.1	Clustering Validity Indices	31
3.5.2	Clustering number k	31
3.6	Incremental Learning	32
3.6.1	Task Definition	32
3.6.2	Quality Control Filter	33
3.6.3	Test Data Per-Class Holdout	33
3.6.4	Training Data Replay Budget	34
3.6.5	Network Architecture, Loss Functions, and Knowledge Distillation Gating	34
4	Implementation and Parameters	37
4.1	Environment and Reproducibility	37
4.2	Data	37
4.3	From Raw Radio Frequency to Fixed Size Time Frequency Spectrum	38
4.4	Network Implementation	39
4.4.1	Texture Extractor	40
4.4.2	Position Extractor and Fusion	42
4.5	Open-Set Recognition	43
4.6	Clustering	44
4.7	Quality Control Filter	47
4.8	Incremental Learning: Test Holdout and Replay	48
4.9	Student Teacher KD and Gate	49
4.10	Summary of Default Hyperparameters	50
5	Results	51
5.1	Open-Set Recognition Result	52
5.2	Clustering Result	52
5.3	Incremental Learning Result	59
6	Conclusion	61

List of Figures

2.1	Time frequency spectrum for DJI Phantom 4 Pro[31]	8
2.2	Time–frequency waterfall representation of Unmanned Aerial Vehicle (UAV) control and video signals.[32]	9
3.1	Comparison of different models’ accuracy.	24
3.2	An overview of the proposed open-world RF-based UAV recognition framework integrating open-set recognition, novel class discovery, and incremental learning with replay.	25
3.3	Knowledge Distillation (KD) pipeline. Left: with teacher, the student is optimized with cross–entropy on all classes and KD alignment on old classes. Right: without teacher, the student is trained only with cross–entropy over all classes.	36
4.1	Overview of Texture Extractor (TE)/Position Extractor (PE) + fusion	41
4.2	Dilated convolution with dilation rate $d=1$ and $d=2$ [112].	41
4.3	Overview of open set recognition	44
5.1	Overall system model for drone classification and incremental learning.	51
5.2	Normalized confusion matrix under open-set testing.	53
5.3	Prediction type distribution in Open-Set Recognition (OSR). The majority of samples are correctly identified as either known or unknown, while errors remain limited.	53
5.4	Comparison between the Elbow Method and the Composite Score Method for determining the optimal number of clusters. The Elbow Method suggests $k = 4$ based on diminishing returns in inertia, while the Composite Score Method identifies the global maximum at $k = 7$, representing the optimal choice.	54

5.5	Per cluster purity pies with sample counts and qualitative ratings.	55
5.6	t-SNE visualization of (a) predicted clusters and (b) true unknown classes. Strong correspondence validates semantic consistency of clustering.	56
5.7	Cluster \rightarrow dominant <i>true</i> class mapping with sample counts. .	57
5.8	Purity against sample count.	58
5.9	Cluster purity bars.	58
5.10	Accuracy on holdout sets vs. old class replay cap (<code>old_max</code>). Old class accuracy exhibits catastrophic forgetting at 0 and recovers fully with small replay, new class accuracy remains $\approx 98\%$ across budgets.	59

List of Tables

- 2.1 Comparison of continual learning scenarios 12
- 2.2 Comparison of related works in RF-based UAV recognition.
Closed: Closed set RF Classification; Open: Open-set RF
Classification; IL: Incremental Learning. 20
- 3.1 Table of Notations 22
- 4.1 Overview of the DroneRFb-Spectra dataset used in this thesis. 39
- 4.2 Class index list used in the dataset. 40
- 4.3 Default hyperparameters and training setup. 50
- 5.1 Detailed comparison of clustering results ($k = 2$ to $k = 15$).
The row highlighted in red indicates the selected optimal k . . . 55
- 5.2 Effect of the old-class replay budget under the holdout
protocol (KD gated off). 59

List of acronyms and abbreviations

AoA	Angle of Arrival
BIC	Bayesian Information Criterion
CE	Cross-Entropy
CH	Calinski Harabasz
CNN	Convolutional Neural Network
CSR	Closed-Set Recognition
DB	Davies Bouldin
DBSCAN	Density-Based Spatial Clustering of Applications with Noise
DFT	Discrete Fourier Transform
DNN	Deep Neural Network
EWC	Elastic Weight Consolidation
FCS	Flight Control Signals
FFT	Fast Fourier Transform
FT	Fourier Transform
GAP	Global Average Pooling
GMM	Gaussian Mixture Model
I/Q	In phase/Quadrature
iCaRL	Incremental Classifier and Representation Learning
IL	Incremental Learning
ISM	Industrial Scientific and Medical
KD	Knowledge Distillation
LwF	Learning without Forgetting
MFCC	Mel-Frequency Cepstral Coefficients
MNL	Multi-Nonlinear Layers
OSR	Open-Set Recognition

PCA	Principal Component Analysis
PE	Position Extractor
PSD	Power Spectral Density
RF	Radio Frequency
SA	Self-Attention
SI	Synaptic Intelligence
STFT	Short-Time Fourier Transform
SVDD	Support Vector Data Description
TE	Texture Extractor
TFS	Time Frequency Spectrum
UAV	Unmanned Aerial Vehicle
USRP	Universal Software Radio Peripheral
VTS	Video Transmission Signals
XGBoost	Extreme Gradient Boosting

Chapter 1

Introduction

The rapid proliferation of **Unmanned Aerial Vehicles (UAVs)** in both civilian and military domains has enabled a wide range of applications. However, it also simultaneously raises pressing concerns regarding security, privacy, and safe airspace management [1, 2]. Therefore, it is significant to be able to detect and classify not only known **UAVs** but also unknown **UAVs**. Extensive literature on **Radio Frequency (RF)** based **UAV** detection consistently identifies machine learning as not only the most promising approach but perhaps the only practical path toward fully autonomous **UAVs** detection [3–5]. Although recent studies have proposed various classifiers and tailored **RF** feature sets, neither academia nor industry has yet produced a system refined enough to autonomously detect, cluster and incrementally learn completely new unknown **UAV** signals. In traditional closed-set classification, the model assumes every test sample belongs to one of the known categories. This assumption fails in real-world applications, such as UAV or drone signal recognition, because new or unseen types frequently appear. **Open-Set Recognition (OSR)** allows the system to identify these unfamiliar signals instead of misclassifying them as known ones. This thesis addresses these gaps by developing a unified, self-learning framework that integrates center based discriminative feature learning, **OSR** that recognizes both known and unknown drones, clustering based unknown class discovery, and automated **Incremental Learning (IL)** with replay mechanisms. This project combines theoretical foundations and practical software implementations to achieve the goals.

1.1 Problem

Although supervised models trained on labeled **UAV** datasets can achieve high accuracy within their domains, they suffer from critical limitations in realistic open-world environments. First, conventional classifiers assume a closed set scenario, leading to overconfident misclassifications when encountering unknown **UAV** types. Second, retraining models from scratch to incorporate new **UAV** categories is computationally expensive and time consuming. Finally, existing **IL** methods struggle with catastrophic forgetting, where performance on previously learned **UAV** classes deteriorates when adapting to new ones. Overcoming these challenges requires methods capable of (1) robust feature extraction under real world interference, (2) unsupervised clustering to discover and label unseen **UAV** classes without prior ground truth, and (3) **IL** strategies that integrate new knowledge while retaining prior expertise.

1.2 Goals

The goal of this thesis is to develop a unified framework for RF based **UAV** recognition and classification that can handle **OSR** scenarios, discover and cluster unknown **UAV** classes, and incrementally learn new categories without catastrophic forgetting. Specifically, the framework aims to (1) separate unknown **UAVs** from known ones using a signal semantic based recognition method, (2) automatically cluster and label unknown drones through adaptive clustering techniques, and (3) incorporate **IL** with replay strategies to continuously adapt to newly discovered categories while maintaining knowledge of previously learned **UAVs**. Together, these objectives establish the foundation for a scalable and practical solution to **UAV** recognition in open-world environments.

1.3 Research Methodology

This research leverages the publicly available **RF** spectrogram dataset hosted on IEEE DataPort [6], recorded with a **Universal Software Radio Peripheral (USRP)** device across three **Industrial Scientific and Medical (ISM)** bands under realistic urban conditions. Unlike many controlled or proprietary datasets, DroneRFb includes interference from WiFi, Bluetooth, and other wireless devices, providing a more representative benchmark for open-world

UAV recognition [4].

The software framework is implemented in Python using PyTorch, with modular components for training, classification, clustering, and **IL**. The network architecture employs three parallel **Convolutional Neural Network (CNN)** encoder branches with different dilation rates to capture multi-scale features, concatenated and passed to an **Multi-Nonlinear Layers (MNL)** embedding head. Supervised training adopts a margin-based contrastive loss and runs for 200 epochs. Unknown signal embeddings are detected via Mahalanobis distance thresholds. For unknown class discovery, embeddings of unknown classes are clustered using K-Means and **Gaussian Mixture Model (GMM)**, with composite internal validity metrics such as inertia, silhouette, **Calinski Harabasz (CH)**, **Davies Bouldin (DB)**, and explained variance to select the optimal cluster number automatically. Incremental learning incorporates newly discovered classes into the model through replay-based updates, ensuring minimal storage overhead while preserving performance on known classes.

1.4 Delimitations

This study employs the public DroneRFb spectrogram dataset, which contains 14,460 samples across seven **UAV** brands, recorded in semi controlled indoor and outdoor urban scenarios using a **USRP** device on **ISM** bands. As a result, findings may not generalize to non **ISM** frequencies, autonomous **UAVs** without **RF** emissions. Additionally, the current evaluation focuses on single **UAV** scenarios; simultaneous multi-**UAV** cases, where signals overlap or interfere, are not specifically addressed. Real time processing and computational efficiency are not the focus, instead, this research emphasizes detection and classification accuracy under realistic interference.

1.5 Ethics and Sustainability

The deployment of **RF** monitoring raises privacy concerns related to the potential interception of non **UAV** communications. To mitigate this, data collection is restricted to known **UAV** frequency bands (433MHz–6GHz). All experimental protocols comply with institutional review board (IRB) approvals and national spectrum regulations, ensuring no personal or unauthorized communications are recorded. From a sustainability perspective,

the system is designed to minimize energy consumption by offloading heavy computation to batch processes rather than continuous real time inference. Incremental model updates are scheduled to leverage existing training infrastructure efficiently, reducing the carbon footprint associated with frequent full model retraining.

1.6 Structure of the thesis

The remainder of this thesis is structured as follows:

- Chapter 2 reviews background of the thesis including related current methods for solving problems and the methods that are implemented such as RF signals, machine learning methods for signal classification and clustering, and IL;
- Chapter 3 details system design, data pipelines, and model architectures;
- Chapter 4 presents implementations based on Chapter 3;
- Chapter 5 shows performance of the proposed framework;
- Chapter 6 concludes with key contributions, limitations, and future research directions.

Chapter 2

Background

This chapter provides the necessary background and related work for this thesis. It introduces the concepts of **UAVs**, **RF** signals, **Closed-Set Recognition (CSR)**, **OSR**, clustering, and **IL**. Moreover, it also introduces related work of current **RF** based machine learning techniques for drone detection and recognition, which includes **CSR**, **OSR**, clustering, and **IL**.

2.1 Unmanned Aerial Vehicles

UAVs, also known as drones, are aircraft that operate without an onboard pilot. They can either be remotely piloted by humans or programmed to follow predefined flight paths [7]. The earliest recorded use of **UAV** technology dates back to World War I in 1917, when the United States and the United Kingdom employed pilotless aircraft as aerial targets for gunnery practice [8].

Since then, **UAVs** have become indispensable in both military and civilian domains. Initially, they were primarily used by defense organizations for reconnaissance, surveillance, and combat missions. As communication and control systems evolved, particularly with the emergence of reliable air-to-ground links and 6G enabled connectivity [9–14]. The potential of **UAVs** expanded far beyond military applications. In hazardous environments such as earthquake zones, hurricane-stricken regions, or offshore oil platforms, **UAVs** offer a safe and efficient means to inspect damage and monitor conditions. They have also revolutionized logistics, for instance, Zipline’s autonomous **UAVs** deliver medical supplies to remote areas in Africa without requiring human couriers [8].

Recent advances in AI and reinforcement learning have further enabled intelligent flight control, airspace coordination, and adaptive communication for connected UAVs networks [15–24]. These developments are paving the way for integrated terrestrial–aerial systems and advanced air mobility operations that rely on robust non-terrestrial network support [25–27].

Looking ahead, the integration of AI driven autonomy, edge computing, and 6G communication is expected to enable new services ranging from urban air mobility and last-mile package delivery to space exploration missions [9, 25]. However, the widespread deployment of UAVs also raises legitimate concerns related to privacy, safety, and misuse. UAVs can infringe on personal privacy, pose collision risks to wildlife and manned aircraft, and may be exploited for illicit activities such as contraband transport [28]. Addressing these challenges through effective regulation, reliable air traffic management, and robust cybersecurity frameworks will be crucial to ensuring the safe and sustainable integration of UAVs into modern airspace [8, 19].

2.2 Unmanned Aerial Vehicles

UAVs also known as drones are aircrafts that operate without an onboard pilot. UAVs can be remotely piloted by humans or programmed to follow a predefined flight path [7]. The first recorded use of drone technology dates back to World War I in 1917. At that time, the United States and the United Kingdom employed pilotless aircraft as aerial targets for gunnery practice [8].

Since then, military and defense organizations have relied on drones for reconnaissance, surveillance and combat missions. As drone capabilities continue to advance, these versatile machines are finding ever wider applications in civilian life. In hazardous environments, such as earthquake zones, hurricane stricken regions or offshore oil platforms, drones offer a safer way to inspect damage and monitor conditions. Moreover, they also save time and resources, for example, Zipline’s autonomous drones deliver medical supplies to remote areas of Africa without requiring human couriers [8].

Looking ahead, the integration of artificial intelligence will unlock yet more uses for drones, from urban air mobility and last mile package delivery to exploration beyond Earth’s atmosphere. However, their proliferation raises legitimate concerns. Drones can infringe on personal privacy, pose collision risks to wildlife and manned aircraft, and even be misused for illicit purposes,

such as transporting contraband or weapons [28]. Addressing these challenges through thoughtful regulation, robust air traffic management and security measures will be essential to realizing the full promise of UAV technology while keeping people and property safe [8].

2.3 Radio Frequency Signals

In wireless communication systems, an RF signal can be viewed as a band pass signal modulated onto a high frequency carrier f_c , which is typically from hundreds of MHz to several GHz. Its continuous time form can be written as

$$s_{\text{RF}}(t) = A(t) \cos(2\pi f_c t + \phi(t)),$$

where $A(t)$ is the envelope amplitude and $\phi(t)$ is the instantaneous phase. Information is encoded in the variations of amplitude and phase, and the carrier f_c serves as the high frequency sine wave that “carries” this information [29]. For digital processing, the RF signal is usually down converted to baseband to obtain the *complex envelope*:

$$s_{\text{BB}}(t) = A(t) e^{j\phi(t)},$$

which concentrates the signal’s spectrum around DC while preserving all information of the original RF waveform. This complex baseband signal can then be decomposed into phase (I) and quadrature (Q) components:

$$s_{\text{BB}}(t) = I(t) + j Q(t),$$

where $I(t)$ and $Q(t)$ correspond to the orthogonal projections of the amplitude and phase. In the digital domain, $I(t)$ and $Q(t)$ can be sampled at a sampling rate f_s to yield discrete sequences $\{I[n], Q[n]\}$. These complex samples are commonly referred to as *raw In phase/Quadrature (I/Q) samples* which are the fundamental input for signal processing pipelines, and machine learning models that analyze and classify RF signals [30].

Figure 2.1 shows the time frequency spectrum of the DJI Phantom 4 Pro. The RF signals between the drone and ground controller can be categorized into two streams: **Flight Control Signals (FCS)** and **Video Transmission Signals (VTS)**. FCS carries control commands and flight logs, and VTS carries the live video feed and telemetry data from the onboard camera. These two streams exhibit distinct time–frequency characteristics. FCS follows a specific

frequency hopping pattern in terms of hop sequence, bandwidth and dwell time, while **VTS** manifests as continuous spectral blocks determined by video encoding parameters such as bitrate and GOP structure. By capturing raw **I/Q** samples with an **RF** receiver, high dimensional time-frequency representations of both **FCS** and **VTS** are obtained, which lays the groundwork for subsequent texture and position feature extraction. Because each drone model and manufacturer enforce their own hopping protocol and encoding settings, the fused signal semantics (texture + position) of **FCS** and **VTS** cluster tightly in semantic space for known models. Any signal whose semantics fall outside these learned clusters can thus be flagged as unknown [31].

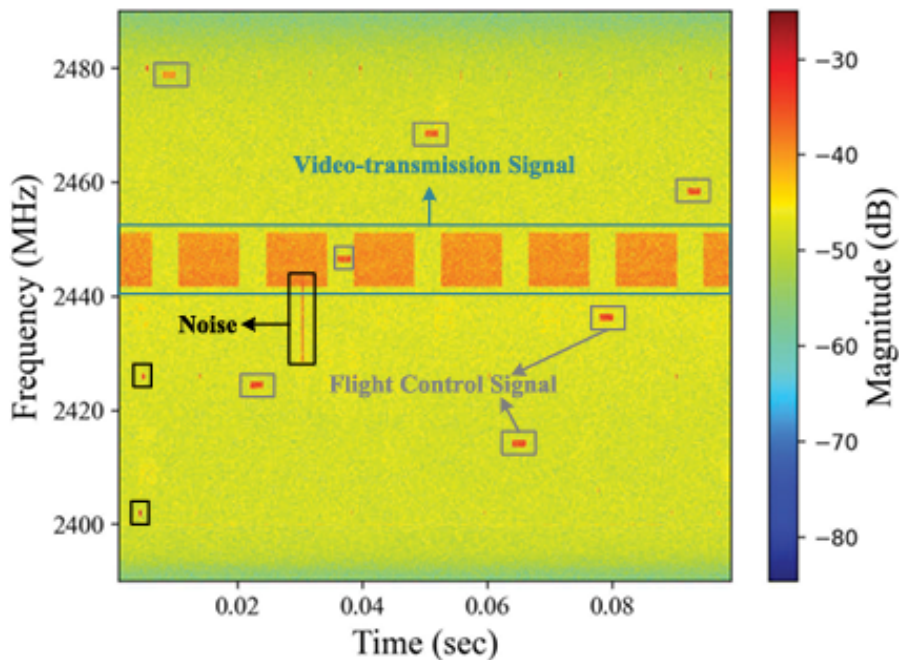


Figure 2.1: Time frequency spectrum for DJI Phantom 4 Pro[31]

UAV communicate with their controllers in the unlicensed 2.4000–2.4835 GHz and 5.470–5.725 GHz bands, which carries uplink control commands and downlink video or telemetry data. For localization and detection, raw **I/Q** samples are split by a spectrum sensor into 3,276 frequency bins, where each bin is ≈ 30.5 kHz wide. Moreover, each frame records both power and **Angle of Arrival (AoA)** per bin at ≈ 300 frames/s, forming a high dimensional time–frequency waterfall representation. Figure 2.2, presents

waterfall diagrams of the **RF** power spectrum recorded while a DJI Inspire 1 was in flight. Time is plotted along the vertical axis and frequency along the horizontal axis. The colors indicates instantaneous **RF** power where white means lowest, and blue means highest. Adjacent to these, we show how the continuous spectrum is discretized into frequency bins at a single time slice. Bins containing the control signals are readily identified by their elevated power levels. At the same time, the corresponding **AoA** can also then be extracted [32].

Captured **I/Q** data can be transformed using various signal processing

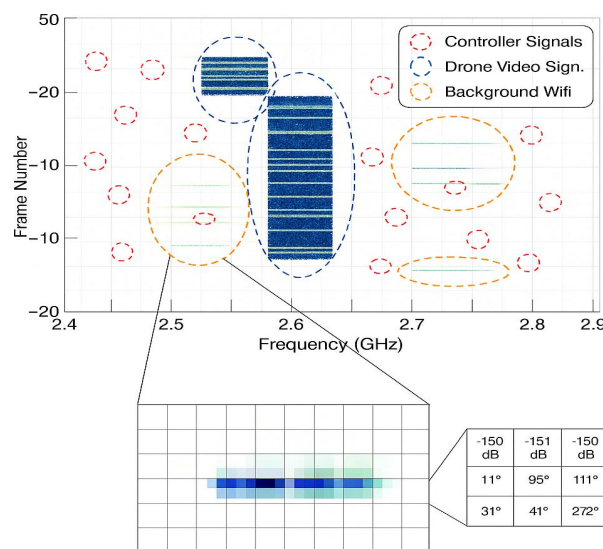


Figure 2.2: Time–frequency waterfall representation of **UAV** control and video signals.[32]

techniques—such as **Fast Fourier Transform (FFT)** [33], **Short-Time Fourier Transform (STFT)** [34], **Mel-Frequency Cepstral Coefficients (MFCC)** [35], wavelet transforms [36], or empirical mode decomposition (EMD) [37] to extract time–frequency features. These features can then be classified through a range of machine learning models, including SVMs [38], fully connected networks [33], CNNs [39], LSTMs [40], or lightweight multi-scale convolutional modules [31].

There are also many other ways utilizing **RF** signals. For example, packet size and inter arrival time features can enable encrypted traffic identification even

when packet payloads are encrypted, the physical layer **RF** transmission still reveals frame durations and spacing. Measurements of energy bursts, packet lengths, and timing intervals are directly derived from the **RF** waveform, making this a **RF** sensing approach [41].

2.4 Machine Learning

This section introduces the core terms in machine learning detection and recognition of drones: Closed-Set Recognition, Open-Set Recognition, Clustering, and Incremental Learning. Moreover, this section also explains their roles and challenges in RF-based drone classification.

2.4.1 Closed-Set Recognition

CSR assumes that all classes encountered during testing have been seen during training [42]. Typical approaches include using a softmax classifier and training with cross-entropy loss. Softmax in a neural network is a commonly used output function. It takes a set of raw scores i.e. logits, and converts them into probabilities. The probabilities means how much chance that a test sample belongs to each class. However, Softmax offers no mechanism to reject unfamiliar inputs and can not reject the test sample and classify it as unknown [42]. Support vector machines(SVM) is another method, it can maximize class margins but may struggle with large datasets problem [43]. Center loss has also been introduced to reduce intra-class variance and increase inter-class separation, in order to produce more compact feature clusters for known classes and further lays the groundwork for downstream **OSR** [44].

2.4.2 Open-Set Recognition

OSR extends **CSR** by allowing the possibility of novel classes while testing with test data. To solve **OSR** problem, there are currently 2 main categories of solutions: discriminative models and generative models. Discriminative models aim to learn to distinguish and differentiate between classes, and furthermore give a boundary in order to classify an unknown class. Generative models aim to learn to understand, simulate and create classes in order to classify an unknown class [42]. Discriminative models include traditional machine learning based methods and **Deep Neural Network (DNN)** based methods. Traditional machine learning based **OSR** methods refer to, for instance, Support Vector Machine (SVM) based **OSR**, nearest neighbor based

OSR and sparse representation based **OSR**. **DNN** based methods refer to OpenMax **OSR** and Deep Open Classification (DOC) **OSR** among others [42].

This is an significant problem in real world because there are always new signals or new types of drones coming up. The models need to have the ability to recognize unknown signals and potentially divide them into different groups based on their specialties. In addition, only **CSR** can lead to serious problems if the models are applied in security areas, because the **CSR** models can not reject the unknown signals and further recognize enemy drones as own drones. One of the common strategies involve applying thresholds to softmax outputs or feature space distances to reject samples that do not fit known classes. Moreover, fitting statistical tail models (e.g., Weibull distributions) to feature activations as in the OpenMax algorithm is also a solution [42]. In addition, using classical novelty detectors such as one class SVM [45] or **Support Vector Data Description (SVDD)** [46] is another approach. Each of these techniques offers different trade offs in terms of rejection accuracy, computational overhead, and sensitivity to hyperparameters when applied to high dimensional RF signal representations.

2.4.3 Clustering

The clustering method aims to divide all the recognized unknown classes into different groups in a way that each group has similar points a.k.a. samples. It is employed to discover new classes among the samples rejected during the **OSR**. The common clustering algorithms are K-Means, **GMM**, **Density-Based Spatial Clustering of Applications with Noise (DBSCAN)**, and Hierarchical Clustering [47]. K-Means and **GMM** remain popular for their simplicity and scalability. But the only difficulty is that it requires specifying the number of clusters in advance. Density based methods such as **DBSCAN** can find clusters of arbitrary shape without a predetermined cluster count, but this method is only sensitive to density [48]. Cluster validity indexes such as Inertia [49], BIC [50], Silhouette score [51], Calinski Harabasz index [51], Davies Bouldin Index [51], explained Variance can be used to select the optimal number of clusters.

2.4.4 Incremental Learnin

IL, also known as continual or lifelong learning, refers to the ability of a model to acquire new knowledge from a non-stationary data stream, and

Table 2.1: Comparison of continual learning scenarios

Scenario	Intuitive description	Mapping to learn
Task-incremental learning	Sequentially learn to solve a number of distinct tasks.	$f : \mathcal{X} \times \mathcal{C} \rightarrow \mathcal{Y}$
Domain-incremental learning	Learn to solve the same problem in different contexts.	$f : \mathcal{X} \rightarrow \mathcal{Y}$
Class-incremental learning	Discriminate between incrementally observed classes.	$f : \mathcal{X} \rightarrow \mathcal{C} \times \mathcal{Y}$

\mathcal{X} is the input space, \mathcal{Y} is the within-context output space, and \mathcal{C} is the context space.

at the same time, the model does not catastrophically forget previously learned information. In contrast to the classical batch learning setting, where all training data are available at once, **IL** algorithms must process data in sequential “experiences” or “contexts” gradually. Because the processed data can not be decided in advance and it is from possibly shifting distributions [52, 53].

Recent work has identified three canonical scenarios of **IL**. As shown in Table 2.1 [52], in *task IL*, the model encounters a sequence of distinct tasks, in table, the term context refers to an underlying distribution from which observations are sampled [52]. The context changes over time. In the continual learning literature, the term task is often used in a way analogous to how the term context is used here. Moreover, the model is informed of the task identity while testing with test data, and further allows task specific components or output heads to be employed. In *domain IL*, the underlying classification problem remains the same but the input distribution changes. Task identity is not provided at test time, instead, the model must learn a shape preserving mapping across domains. Finally, in *class IL*, new classes appear over time and the model must both retain existing class knowledge and discriminate among all classes known and newly discovered classes also known as newly added without explicit context labels [52].

Each scenario poses unique challenges. Task **IL** focuses on efficient parameter sharing and positive transfer between tasks. Domain **IL** must mitigate domain shift without task labels. Class **IL** must overcome catastrophic forgetting while simultaneously expanding the classifier to handle an ever growing label space. These distinctions provide a structured foundation for evaluating and

developing continual learning strategies in **RF**-based drone recognition and beyond [52].

There are many methods in class **IL**. Replay based methods retain a small buffer of past samples for joint training. Regularization based methods penalize changes to important parameters (e.g., Elastic Weight Consolidation). Parameter isolation methods allocate dedicated capacity for new classes [52].

Regularization based methods (such as **Elastic Weight Consolidation (EWC)**, **Synaptic Intelligence (SI)** and **Learning without Forgetting (LwF)**) do not save old data like replay based methods, it prevents catastrophic forgetting by limiting the change of the parameters that are important for retaining old knowledge. However, as shown in previous research, regularization based methods such as **EWC** and **LwF** in **IL** are sensitive to feature drift. Features are not updated, and they are sensitive to large domain shifts between incremental tasks [54]. We propose to estimate the drift of features that happens due to the learning of new tasks [55]. The method only limits the parameters, it does not control and limit the change of feature space, which can be deformed while training new classes with pseudo labels even if the parameters do not change much. This is called feature drift, where the feature spaces of the new and old classes overlap and distort, which can lead to the old knowledge being quickly forgotten [52, 55].

Template based classification methods such as **Incremental Classifier and Representation Learning (iCaRL)** uses nearest mean classifier (NCM) instead of softmax classifier. It calculates the distance and assigns it to whoever is closest and cannot backpropagate. NCM means to use a representative template that includes “feature means” from all samples in the category, to classify a test sample. In order to find the class the test sample belongs to, the next step is to calculate the distance between the test sample’s feature vector and “feature means” vectors. After that, choose the nearest feature mean i.e. representative template, and further label the test sample as the class of the nearest feature mean. However, it becomes impossible to be updated through backpropagation and the classifier itself can not jointly optimize the feature extractor + classifier. Therefore, it is not as flexible as the softmax classifier and depends a lot on the quality of the selected exemplar and prototype [52, 55].

2.5 Related Work

Existing research on this thesis can be organized into five areas: [2.5.1 Drone Recognition](#), [2.5.2 Closed-Set Radio Frequency Classification](#), [3.4 Open-Set Radio Frequency Classification](#), [3.5 Clustering](#), [3.6 Incremental Learning](#) [2.5.6 Summary](#).

2.5.1 Drone Recognition

Current methods for classification are through vision, audio, thermal, radar and radio frequency (RF) signals. Vision based machine learning methods such as DNN that works excellent in detecting large objects in videos or pictures, but dataset for drones are considered relatively small. In addition, it is low light visibility and low coverage in some scenarios, which brings also challenges using vision detection and classification [56]. Audio based detection does not work well when there is very large noises in the environment such as airport, but it is impossible to avoid [57–63]. Thermal based detection is easily influenced by weather conditions [56, 64]. Radar based methods detect drones by catching reflected radio frequency wave from drones, but the drones are too small and some of them are plastics that are hard to be detected. Furthermore, radar based methods can take birds as drones. which makes it difficult reach a high accuracy [65]. RF based detection works by collecting communications between drones and controller, where is at channels of frequencies 2.4GHz and 5GHz. Previous researches showed that machine learning methods with RF signals give good performance on classifications [3, 4, 66]. In summary, RF based detection combined with machine learning methods is considered as a promising approach.

2.5.2 Closed-Set Radio Frequency Classification

RF based drone detection and identification has been extensively explored using a variety of signal processing and machine learning techniques. Xie et al. proposed a dual source framework that jointly analyzes FCS and VTS for UAV detection and identification [63]. Al-Sa'd et al. assembled a large open source RF drone database and evaluated several deep feed forward neural networks for drone type classification [62]. This method demonstrates the value of purely data driven models without manual feature design. Xu et al. applied STFT followed by feature reduction methods to extract RF fingerprints for multi-drone detection, which achieves high accuracy on amateur UAV

signals [67]. Kilic et al. used spectral features derived from the **RF** power spectrum to classify multiple drone models under varying channel conditions [68]. Wavelet based feature extraction has also been investigated. Medaiyese et al. employed discrete wavelet transforms and classical classifiers to improve **UAV** identification performance [69]. A dynamic, adaptive **RF**-fingerprint decomposition technique was introduced by Xu et al. to enhance detection of micro **UAVs** via machine learning [70]. Shoufan et al. demonstrated pilot identification by classifying radio control signal patterns, which links signal traces to specific operators [71]. Recent studies have addressed noncooperative and encrypted scenarios. Wang et al. presented an ID-based method for robust identification of heterogeneous drones without prior cooperation [72]. Rajendran et al. proposed an unsupervised spectrum anomaly detection approach with interpretable features for general **RF** surveillance [73]. Cai et al. proposed a lightweight multiscale convolutional backbone for **UAV RF** fingerprinting identification, which significantly reduces model size and computational complexity while maintaining high identification accuracy even at low SNRs [74]. Fanid et al. used machine learning on encrypted WiFi traffic, which exploits packet timing and size for delay aware drone detection and mode classification [75]. Lin et al. demonstrated passive **UAV** detection in urban 5G networks by learning from synchronization signal blocks [76]. This method achieves reliable detection without active probing. While these works advance **RF**-based sensing, they predominantly assume a **CSR** of known drone types, which underscores the need for **OSR** and **IL** frameworks as pursued in this paper. Several machine learning and deep learning approaches on the DroneRF dataset have a good performance. They begin by extracting **Power Spectral Density (PSD)** features from raw **RF** samples, using XGBoost to achieve 100%, 100%, and 99.73% accuracy on 2, 4, and 10 class classification tasks, respectively. To assess feature fusion, they combine **PSD** with additional handcrafted features. This brings slightly lower accuracies of 99.13%, 99.11%, and 93.84%. Exploring deep models, a one dimensional **CNN** attains 100%, 94.31%, and 86.29%. Finally, a hybrid scheme, where a 1D CNN serves as feature extractor and XGBoost as the classifier, produces 100%, 99.82%, and 99.51% accuracy. This shows the benefit of combining deep representation learning with gradient boosting for robust **UAV** identification [3, 4, 66].

2.5.3 Open-Set Radio Frequency Classification

OSR methods aim to detect and reject samples from novel classes not seen during training, yet only a handful have been applied to **RF**-based drone identification. Soltani et al. extend standard neural classifiers by applying a dual-threshold on SoftMax outputs and multi-network agreement to flag unknown **UAV** signals [77]. Bendale and Boulton's OpenMax layer replaces SoftMax with a Weibull-based tail modeling step, which estimates the probability that a feature vector belongs to an unknown class [78]. More recently, Wang et al. introduce an uncertainty inspired **OSR** framework (UIOS) that uses Softplus activations to parameterize Dirichlet distributions over class "evidence," and produce an uncertainty score to separate known from unknown samples [72]. Li et al. propose a multi-modal marginal prototype approach for **RF** modulation recognition, jointly optimizing prototype and reciprocal point learning, and augmenting training with GAN-generated unknown samples to improve robustness [79]. While these methods successfully reject out-of-distribution inputs, they do not assign unknown samples to distinct emerging classes. To address this, Dong et al.'s SR2CNN leverages an autoencoder to embed signals into feature space and then classifies by Mahalanobis distance to class centroids with adaptive thresholds, enabling fine-grained unknown class recognition [80]. Han et al. further refine this strategy by replacing Mahalanobis distance with Euclidean distance to enhance detection of novel jamming patterns [81]. However, both SR2CNN and its variants require access to held-out unknown examples for threshold calibration, which limits their applicability when no unknown data are available [31].

A Signal Semantic based **OSR** (S3R) framework tailored for **RF** based drone recognition are brought forwards. S3R is designed to not only detect unknown drones but also refine their classification into true categories. It begins by applying **STFT** to generate **Time Frequency Spectrum (TFS)**, which effectively decouples drone signals from same band interference. Then, a **Texture Extractor (TE)** based on dilated convolutional layers is employed to capture bandwidth and duration related texture features, while a **Position Extractor (PE)** with position encoding and self attention mechanisms learns frequency points, intervals, and other positional attributes, which enhances the discrimination of same brand drones. These features are fused and optimized in a semantic space using center loss and clustering loss to achieve high intra class compactness and inter class separability. For classification, S3R

determines self adaptive Mahalanobis distance thresholds for each known class through outlier analysis without relying on unknown class data, and subsequently projects unknown samples into an augmented semantic space enriched with intermediate layer features from **TE** to perform clustering, estimate the number of unknown categories, and assign labels accordingly. Compared to conventional **OSR** methods that merely reject unknown samples, S3R enables refined recognition of unknown instances. In this way it significantly improves generalizability and applicability in open world drone monitoring [31].

2.5.4 Clustering

Fourth, unsupervised clustering algorithms and dynamic cluster count selection strategies have been studied for class discovery in high-dimensional data, yet few works integrate these with **RF** sensing. The clustering method aims to divide all the recognized unknown classes into different groups in a way that each group has similar points. The common clustering algorithms are K Means [82–86], **GMM** [86–88], **DBSCAN** [48, 89–91], and Hierarchical Clustering [47, 92–96].

2.5.5 Incremental Learning

IL, also referred to as lifelong or continual learning, has emerged as a crucial paradigm for enabling models to adapt to new tasks or classes without retraining from scratch, while retaining knowledge from previously learned data. This capability is essential in dynamic and resource constrained environments such as wireless communications, IoT, and **UAV** networks.

Early research in wireless device identification leveraged **DNN** to passively recognize devices from their radio frequency (**RF**) signals. However, traditional **IL** approaches in this domain often relied on storing historical data or generating replay samples, which introduces high memory and computational costs, making them unsuitable for IoT applications. To address these challenges, Liu et al. proposed a channel separation enabled **IL** (**CSIL**) strategy that avoids using historical data by separating device fingerprints in latent space, thus mitigating catastrophic forgetting and improving scalability in class incremental scenarios [97].

In **RF** recognition, the emergence of novel wireless emitters requires models

to simultaneously perform few shot learning and **OSR** recognition under continuous incremental updates. Li et al. introduced the Meta **RF** framework, a meta learning based few shot **OSR IL** (FSOSIL) approach [98]. By simulating pseudo incremental tasks and incorporating modules such as **RF** feature augmentation, open loss, and adaptive thresholding, Meta **RF** significantly improves recognition performance for both known and unknown devices in continuous evolution settings.

In **UAV** intrusion detection, maintaining model adaptability is equally important. Farrukh and Khan proposed an autonomous self **IL** architecture that integrates signature based and anomaly based detectors [99]. Their system autonomously updates its classifier with new attack classes detected by the anomaly module. The system eliminates the need for human intervention and enables both known and unknown attack classification.

Beyond single node learning, federated continuous learning has been explored to maintain privacy and efficiency in distributed **UAV** networks. He et al. presented a federated continuous learning framework based on a stacked broad learning system (FCL-SBLS) assisted by digital twin networks [100]. This approach incrementally updates intrusion detection models without catastrophic forgetting, which leverages asynchronous federated aggregation and **UAV** selection optimization. In this way, it improves both accuracy and efficiency in resource constrained **UAV** edge environments.

Overall, recent advances in **IL** span directions such as memoryless **IL** for IoT device fingerprinting, meta learning enhanced few shot **OSR** adaptation, and autonomous or federated **IL** in **UAV** cybersecurity. However, these methods share a common limitation. The first limitation is in real world environments with rapidly changing data distributions and ambiguous class boundaries, the recognition accuracy of basic **IL** models often drops significantly. In particular, under **OSR** and noise interference conditions, models are prone to catastrophic forgetting. This leads to simultaneous degradation in the recognition of both new and old classes. Similar phenomena have been observed in sequential tasks such as continual visual odometry, where the empirical study by Cudrano et al. [101]. It shows that traditional continual learning methods face a clear trade off between task transfer and forgetting, and that basic methods have limited overall accuracy in balancing adaptation to new tasks with the retention of previous knowledge.

Moreover, existing IL methods often overlook the quality of the input feature space before incremental updates. Many frameworks directly train on incoming data without first optimizing class boundaries or handling noisy and ambiguous samples. This can exacerbate feature space overlap, which has been identified as a key factor behind catastrophic forgetting in class incremental settings [97]. In addition, scenarios such as OSR [98] and dynamic UAV network intrusion detection [99, 100] require the system to distinguish between known, novel, and noisy patterns before model updates, yet few studies provide a robust pre incremental structure analysis.

To address this gap, this thesis introduces an improved clustering based pre processing stage prior to IL. Our method evaluates multiple cluster validity metrics such as Inertia [49], BIC [50], Silhouette score [51], Calinski Harabasz index [51], Davies Bouldin Index [51], explained Variance, in order to intelligently select the optimal number of clusters k for various data regimes, including multiclass ($k \geq 2$), single class ($k = 1$), and noise only ($k = 0$) cases. This ensures high inter class separability and intra class compactness before the supervised IL stage. It directly mitigates decision boundary confusion. By explicitly handling edge cases and visualizing the k selection process, our pipeline is well suited for OSR, few shot, and evolving environments, which enables the IL model to adapt rapidly while preserving prior knowledge.

2.5.6 Summary

In summary, existing RF based UAV recognition studies have demonstrated strong performance in CSR classification and, to a lesser extent, in OSR detection and IL. As shown in Table 2.2, the distinctions between the approach proposed in this thesis and prior studies are clearly outlined. Most approaches address these challenges in isolation. For instance, CSR models lack mechanisms for handling unseen classes, OSR methods often stop at rejection without further class discovery, and IL frameworks are prone to catastrophic forgetting when operating in dynamic open environments. Clustering has been explored mainly as a stand alone unsupervised technique rather than as an integrated pre processing stage for OSR or continual learning. These gaps motivate the integrated approach proposed in this thesis, which combines OSR recognition, clustering based unknown class discovery, and IL into a unified pipeline. In this way, it enables robust adaptation to evolving UAV signal environments while preserving knowledge of previously learned classes.

Table 2.2: Comparison of related works in RF-based UAV recognition. Closed: Closed set RF Classification; Open: Open-set RF Classification; IL: Incremental Learning.

Study	Closed	Open	Clustering	IL
Xie et al. [63]	✓	✗	✗	✗
Al-Sa'd et al. [62]	✓	✗	✗	✗
Xu et al. [67]	✓	✗	✗	✗
Kılıç et al. [68]	✓	✗	✗	✗
Medaiyese et al. [69]	✓	✗	✗	✗
Xu et al. [70]	✓	✗	✗	✗
Shoufan et al. [71]	✓	✗	✗	✗
Wang et al. [72]	✓	✓	✗	✗
Rajendran et al. [73]	✓	✗	✗	✗
Cai et al. [74]	✓	✗	✗	✗
Fanid et al. [75]	✓	✗	✗	✗
Deng et al. [102]	✓	✗	✗	✗
Lin et al. [76]	✓	✗	✗	✗
Soltani et al. [77]	✓	✓	✗	✗
Bendale et al. [78]	✓	✓	✗	✗
Wang et al. (UIOS) [103]	✓	✓	✗	✗
Li et al. (ExpertOSR) [79]	✓	✓	✗	✗
Dong et al. (SR2CNN) [80]	✓	✓	✗	✗
Han et al. [81]	✓	✓	✗	✗
S3R [31]	✓	✓	✓	✗
Liu et al. [97]	✓	✗	✗	✓
Li et al. [98]	✓	✗	✗	✓
Farrukh & Khan [99]	✓	✗	✗	✓
He et al. [100]	✓	✗	✗	✓
This thesis	✓	✓	✓	✓

Chapter 3

System Model & Problem Formulation

3.1 Model Selection and Justification

In order to solve the problems that are described in last chapter, different methods have been explored that are explained in details below. Legend of symbols and notations used in this chapter are listed in [Table 3.1](#). As shown in [Figure 3.2](#), this thesis has explored solutions for [CSR](#) problem, [OSR](#) problem, clustering and [IL](#). For [CSR](#) problem i.e., all the used datasets are known samples, feature extraction methods such as [PSD](#), [Discrete Fourier Transform \(DFT\)](#), wavelet, [STFT](#), [CNN](#) and transformer are utilized. The features are obtained are further fed into machine learning classification models such as [Extreme Gradient Boosting \(XGBoost\)](#) and [CNN](#).

[Figure 3.1](#) illustrates the accuracy of classifying drone types while applying different feature extraction methods and classifiers. The approach that performs best is [XGBoost](#) classifier combined with [PSD](#) features, which gives an accuracy of 0.996. The next best performing method is [STFT-CNN](#) Transformer, which means that the raw [RF](#) signals are processed with [STFT](#) and are further processed by [CNN](#) and Transformer to obtain semantic features for the classification [[31](#)]. This method gives accuracy at 0.988, which is close to [CNN](#) Feature Extractor + [XGBoost](#) with accuracy 0.983. Both methods demonstrate strong classification capabilities. In contrast, [CNN](#) + Wavelet only reaches the lowest accuracy at 0.777. These results demonstrate that [PSD](#)-based [XGBoost](#), [STFT-CNN](#) Transformer and [CNN](#) Feature Extractor + [XGBoost](#) are good choices for classifying known drones. However, when

it comes to classify unknown drones, although **XGBoost**-based approaches slightly outperform the **STFT-CNN** Transformer in terms of raw accuracy, the **STFT-CNN** Transformer semantic method is still chosen as the primary model. The reasons are explained in detail below.

Table 3.1: Table of Notations

Symbol	Description
$\mathbf{X} \in \mathbb{R}^{T \times W}$	Time–frequency spectrum (TFS) input.
T, W	Time and frequency resolution.
$x(t)$	Complex baseband I/Q signal.
n, w	STFT frame and frequency bin indices.
R_W	STFT hop size.
$g(t)$	Hamming window.
$X(n, w), \hat{X}(n, w)$	Log power and normalized spectrogram.
F_θ	Feature extractor (TE/PE + fusion).
$\Phi_{\text{TE}}, \Phi_{\text{PE}}$	Texture CNN and positional Transformer.
Φ_{fusion}	Fusion layer for final embedding.
$\mathbf{f}_i \in \mathbb{R}^{d_s}$	Semantic feature of sample i .
$Z_i^{(c)}$	Feature map at layer i (dilation c).
$\mathbf{W}_i^{(c)}, \mathbf{b}_i^{(c)}$	Convolution weights and bias.
z_c, \tilde{z}_a	Pooled vector of branch c ; texture embedding.
SA, MNLS	Self-attention; multi-nonlinear layers.
\tilde{z}_b, \tilde{z}_c	Positional embeddings (time/frequency).
$\mathcal{D}_{\text{train}}, \mathcal{D}_{\text{new}}, \mathcal{D}_{\text{old}}$	Training, new, and replay datasets.
\mathcal{C}_0	Initial known class set.
\mathcal{C}_t	New classes discovered at step t .
$\boldsymbol{\mu}_k$	Center of class k .
ϑ	Margin in clustering loss.
η_1, η_2, η_3	Weights for loss terms.
α	Learning rate.
$\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k^{-1}$	Mean and inverse covariance of class k .
$\text{dist}(\mathbf{z}, \boldsymbol{\mu}_k)$	Mahalanobis distance.
R_k	Decision region for class k .
τ_k	Rejection threshold.
\hat{y}	Predicted label ($-1 = \text{unknown}$).
\mathcal{U}	Unknown sample set.
$\mathbf{F}_{\mathcal{U}}$	Feature matrix of unknown samples.
k, k_{max}, k^*	Cluster number, max, and selected value.
S_k, CH_k, DB_k, V_k	Silhouette, Calinski–Harabasz, Davies–Bouldin, variance.
$\text{Inertia}(k)$	Within-cluster sum of squares.

Table 3.1 (continued)

Symbol	Description
BIC	Bayesian information criterion (GMM only).
Q_k	Composite clustering score.
k_e, k_s	Elbow and score-based k candidates.
h_θ	Student feature extractor (MLP).
$W_{\text{all}}, W_{\text{old}}$	Classifier heads (all / old).
λ	KD loss weight.
T	KD temperature.
α_{min}	KD gating threshold.

First, the **STFT** representation provides a richer description of the signals both in time and frequency, which captures local spectral dynamics that **PSD** or **Fourier Transform (FT)** features may oversimplify [31, 104, 105]. Second, the Transformer architecture effectively models long-range dependencies and semantic patterns, which improves interpretability compared to tree-based ensemble methods such as **XGBoost** [31].

Moreover, tree based methods such as **XGBoost** mainly rely on the feature distributions observed during training, which makes them less effective when encountering unknown samples or samples out of the distribution. This means that they lack the ability to generalize beyond predefined decision boundaries. In addition, **XGBoost** does not natively support incremental learning, which means the entire model often needs to be retrained from scratch when new data or unknown samples appear. Therefore, **XGBoost** is impractical and limited for dynamic, real-world applications [106–110].

XGBoost, as a tree-based ensemble method, does not inherently support incremental learning. Each boosting round constructs new decision trees based on the residuals of the previous trees, and once these trees are built, their structure becomes fixed. This static nature prevents the model from updating existing trees when new data arrives. Although **XGBoost** provides an option to continue training from a saved checkpoint, this approach merely appends additional trees rather than adapting previously learned structures [106–110]. As a result, maintaining optimal performance with continuously evolving datasets often requires retraining the model from scratch, which is inefficient and impractical in dynamic real-world scenarios.

In contrast, deep learning architectures such as **CNNs** and Transformers are parameterized and differentiable, which makes them inherently more suitable for incremental or continual learning. Their parameters can be directly updated through gradient descent when new data becomes available, without rebuilding the model from the ground up. Specifically, **CNNs** are effective in capturing localized spectral patterns, while Transformers excel at modeling long-range dependencies and semantic relationships. This synergy allows **STFT-CNN** Transformer-based frameworks to be fine-tuned efficiently, and it can maintain robustness and scale seamlessly with new data and under distribution shifts. Therefore, compared to tree-based ensemble methods, **STFT-CNN** Transformer-based models offer a more flexible and adaptive solution for real-world applications that demand both incremental learning capability and strong generalization to unknown samples [31]. Consequently, despite the slightly lower accuracy compared to **XGBoost + PSD**, the **STFT-CNN** Transformer offers a more practical and future proof solution for tasks that demand both robust performance and the ability to recognize unknown samples and further incrementally learn the samples.

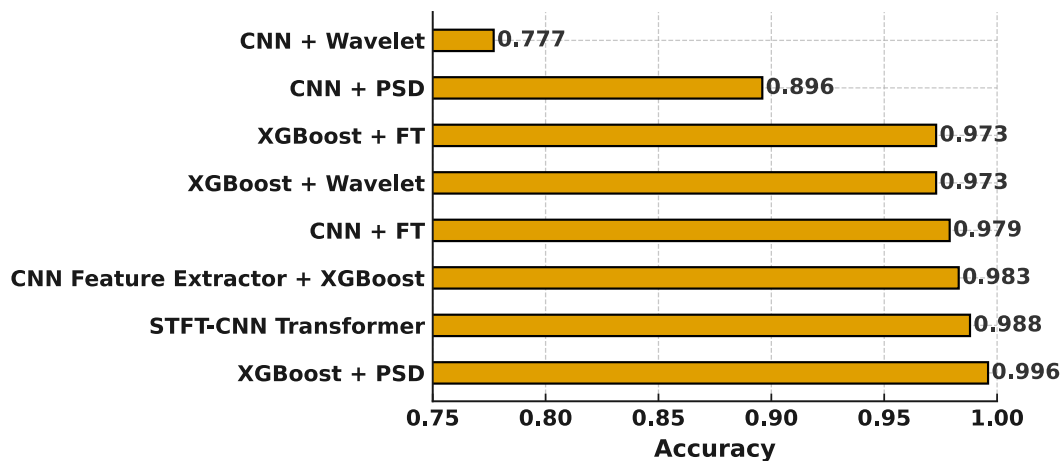


Figure 3.1: Comparison of different models' accuracy.

Therefore, this thesis continues to explore **STFT**, **CNN** and transformer to extract semantic features and further classify unknown samples. Moreover, the classified unknown samples are further clustered into different groups which are then incrementally learned continuously to the model that is used for closed set dataset.

The pipeline of the thesis has five stages: (1) Use **STFT** to construct **TFS**

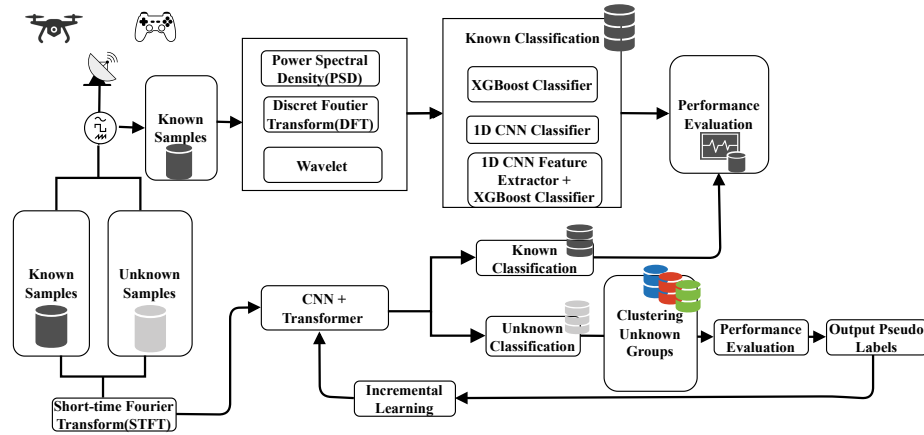


Figure 3.2: An overview of the proposed open-world RF-based UAV recognition framework integrating open-set recognition, novel class discovery, and incremental learning with replay.

from complex **I/Q RF** signals (2) Semantic learning of texture and position features with respectively **TE** and **PE** (3) Open set detection (4) Clustering for unknown class discovery (5) **IL** on the discovered novel classes. More details are explained below.

3.2 Time Frequency Spectrum Feature Construction

In order to distinguish **UAV** signals from background interference, **STFT** is applied to the raw complex **I/Q** signals. The **STFT** operation divides the long signal into equal length sub-segments and performs a **FFT** on each segment, thereby producing a 2 dimensional **TFS** that captures both temporal and spectral characteristics. Since the received power of drones located near the **RF** receiver can be much stronger than that of distant drones, a logarithmic scaling of the power spectrum is used to compress the dynamic range and avoid overwhelming weaker signals. For frame index n and frequency bin w , the log power spectrogram is [31]:

$$X(n, w) = 20 \log_{10} \left| \sum_{t=1}^W x(t + nR_W) g(t) e^{-j2\pi wt/W} \right|, \quad (3.1)$$

where R_W denotes the sliding length of the short-time window, $g(t)$ is the Hamming window function used to reduce spectral leakage, $w \in \mathcal{W} = \{1, 2, \dots, W\}$ indexes the positions within each window of length W , and $n \in \mathcal{N} = \{1, 2, \dots, N\}$ indexes the short-time windows with total number N . Furthermore, to ensure fair comparison across samples, the spectrogram is normalized to the range $[0, 1]$ by subtracting the minimum value and dividing by the range of the spectrum. In this way, the processed **TFS** provides a standardized and robust representation that is well-suited for input into subsequent machine learning models [31]. The normalization to X is denoted:

$$\hat{X}(n, w) = \frac{X(n, w) - \min[X]}{\max[X] - \min[X]}. \quad (3.2)$$

3.3 Semantic Construction and Optimization

To obtain a discriminative semantic representation of the **TFS**, we employ two complementary modules: Texture Extractor (TE) and Position Extractor (PE). Thereafter we fuse their output through multi-nonlinear layers [31].

3.3.1 Texture Extractor (TE)

The TE is designed to capture spectral texture cues such as bandwidth and temporal duration. It consists of three dilated convolutional branches with dilation rates $c \in \{1, 3, 5\}$, each providing a different receptive field. Before entering the TE, the input spectrogram $X(n, w)$ obtained from the short-time Fourier transform is normalized according to (4) to yield $\hat{X}(n, w)$, ensuring consistent amplitude scales across UAV signals. This normalized spectrogram serves as the initial input feature map of the convolutional extractor, i.e., $\mathbf{Z}_0^{(c)} = \hat{X}(n, w)$. For branch r , the i -th feature map is computed as [31]:

$$\mathbf{Z}_i^{(c)} = \text{MaxPool}(\text{ReLU}(W_i^{(c)} * \mathbf{Z}_{i-1}^{(c)} + b_i^{(c)})), \quad i = 1, \dots, I_{\max}, \quad (3.3)$$

where $*$ denotes convolution, $W_i^{(c)}$ and $b_i^{(c)}$ are the learnable convolution weights and biases, $\text{ReLU}(\cdot)$ introduces non-linearity, and $\text{MaxPool}(\cdot)$ downsamples the feature map to retain salient features. The dilation rate c controls the spacing of convolution kernels, thereby adjusting the receptive field: small dilation (e.g., $c = 1$) captures fine-grained details, while larger dilation (e.g., $c = 5$) captures broader contextual patterns. Stacking I_{\max} layers allows the network to learn hierarchical texture patterns.

Global Average Pooling (GAP) aggregates the final feature maps into 1-D vectors:

$$z_c = \text{GlobalAvgPool}(Z_{I_{\max}}^{(c)}), \quad c = 1, 3, 5, \quad (3.4)$$

where $\text{GlobalAvgPool}(\cdot)$ computes the mean value over all spatial positions, yielding a compact representation. The three pooled outputs are concatenated:

$$\tilde{z}_0 = z_1 \oplus z_3 \oplus z_5, \quad (3.5)$$

with \oplus denoting vector concatenation. Finally, the combined vector is passed through **MNLs**:

$$\text{MNLs}(\tilde{z}_0) = \phi_{J_{\max}-1}(\cdots \phi_1(\phi_0(\tilde{z}_0))), \quad (3.6)$$

where each layer is defined as

$$\phi_j(\tilde{z}_j) = \text{ReLU}(w_j * \tilde{z}_j), \quad j = 0, \dots, J_{\max} - 1, \quad (3.7)$$

with w_j the learnable weight of the j -th layer. This design extracts increasingly abstract features and outputs the texture embedding $\tilde{z}_a = \text{MNLs}(\tilde{z}_0)$. In summary, TE captures multi-scale texture patterns in the spectrogram, which are essential to differentiate UAV signals that may share similar global structures but differ in local textures [31].

3.3.2 Position Extractor (PE)

The PE complements TE by encoding positional information, ensuring that the model is aware of where patterns occur in the time–frequency plane. A sinusoidal positional encoding is first applied [31]:

$$P(n, w; X) = \begin{cases} \sin(\omega_w n), & w \bmod 2 = 1, \\ \cos(\omega_w n), & w \bmod 2 = 0, \end{cases} \quad (3.8)$$

where n is the feature index, w is the position index (time or frequency), and $\omega_w = 10000^{-2w/W}$ controls the frequency of the sinusoid. This encoding ensures that each position in the spectrogram is mapped to a unique vector, so that it can preserve absolute and relative positional information. The position-

augmented inputs are then defined as

$$\hat{X}_T(n, w) = X(n, w) + P(n, w; X), \quad (3.9)$$

$$\hat{X}_F(w, n) = X^\top(w, n) + P(w, n; X^\top). \quad (3.10)$$

To model long-range dependencies, we apply a stack of **Self-Attention (SA)** layers [31]:

$$\text{Atten}(Y^0) = \varphi_{L_{\max}-2}(\cdots \varphi_1(\varphi_0(Y^0))), \quad (3.11)$$

where the l -th **SA** block is

$$\varphi_l(Y^l) = \text{softmax}\left(\frac{Y^l Q_l K_l^\top (Y^l)^\top}{\sqrt{\gamma_l}}\right) Y^l V_l. \quad (3.12)$$

$Q_l, K_l, V_l \in \mathbb{R}^{n_l \times \gamma_l}$ are learnable matrices that generate queries, keys, and values, respectively, and γ_l is the scaling dimension to stabilize gradients. The softmax term computes similarity between positions, which allows each position to attend to others with learned importance weights. As a result, the outputs

$$\tilde{X}_T = \text{Atten}(\hat{X}_T), \quad \tilde{X}_F = \text{Atten}(\hat{X}_F) \quad (3.13)$$

capture both local and global positional dependencies. Finally, the outputs are flattened and mapped through **MNLs** to obtain positional embeddings:

$$\tilde{z}_b = \text{MNLs}(\text{Flatten}(\tilde{X}_T)), \quad (3.14)$$

$$\tilde{z}_c = \text{MNLs}(\text{Flatten}(\tilde{X}_F)). \quad (3.15)$$

The rationale is that UAV signals may occupy characteristic positions in the spectrum (e.g., certain frequency bands), so explicitly encoding and modeling positional information enhances discriminability [31].

3.3.3 Fusion

The final semantic vector is obtained by fusing the texture and positional embeddings:

$$z = \text{MNLs}(\tilde{z}_a \oplus \tilde{z}_b \oplus \tilde{z}_c), \quad z \in \mathbb{R}^D, \quad (3.16)$$

where D is the semantic dimension. Fusion allows the network to jointly leverage shape/texture cues and positional cues [31].

3.3.4 Semantic Optimization

To optimize discriminability, margin based contrastive loss is applied where three complementary losses are jointly minimized. First, the center loss encourages samples of the same class to cluster around their class center [31]:

$$L_{\text{cen}}(\theta; M) = \sum_{X_i \in M} \|E(X_i; \theta) - \mu_{y_i}\|_2, \quad (3.17)$$

where $E(X_i; \theta)$ is the embedding of sample X_i , μ_{y_i} is the semantic center of class y_i , and $\|\cdot\|_2$ is the Euclidean norm. This reduces intra-class variance.

Second, the clustering loss enforces inter-class separation by penalizing embeddings that are too close to centers of other classes [31]:

$$L_{\text{clu}}(\theta; M) = \sum_{X_i \in M} \max_{k \neq y_i} \{0, \vartheta - \|E(X_i; \theta) - \mu_k\|_2\}, \quad (3.18)$$

where ϑ is a margin that defines the minimum acceptable distance. This prevents confusion between classes.

Third, the **Cross-Entropy (CE)** loss provides supervised guidance [31]:

$$L_{\text{ce}}(\theta; M) = -\frac{1}{M} \sum_{X_i \in M} y_i \log P(X_i; \theta), \quad (3.19)$$

where $P(X_i; \theta)$ is the predicted probability of X_i belonging to class y_i . This aligns embeddings with class labels.

The final optimization updates parameters by minimizing a weighted sum of all three losses through stochastic gradient descent (SGD) [31]:

$$\theta \leftarrow \theta - \alpha \frac{\partial}{\partial \theta} (\eta_1 L_{\text{cen}} + \eta_2 L_{\text{clu}} + \eta_3 L_{\text{ce}}), \quad (3.20)$$

where η_1, η_2, η_3 are trade-off weights, and α is the learning rate. This joint objective balances compact intra-class clusters, separated inter-class structures, and correct label prediction. As a result, it brings highly discriminative semantic embeddings for downstream open-set recognition, clustering, and incremental learning.

3.4 Open-Set Recognition

For each known class $k \in \mathcal{C}_0$, we estimate its semantic mean $\boldsymbol{\mu}_k$ and inverse covariance $\boldsymbol{\Sigma}_k^{-1}$ from training embeddings, and measure class affinity using the Mahalanobis distance [31]:

$$d_M(\mathbf{z}, \boldsymbol{\mu}_k) = \sqrt{(\mathbf{z} - \boldsymbol{\mu}_k)^\top \boldsymbol{\Sigma}_k^{-1} (\mathbf{z} - \boldsymbol{\mu}_k)}. \quad (3.21)$$

The per-class decision region is [31]:

$$R_k = \{ \mathbf{z} \in \mathbb{R}^D \mid d_M(\mathbf{z}, \boldsymbol{\mu}_k) < \tau_k \}, \quad (3.22)$$

where τ_k is a class-specific rejection threshold.

The distance metric can be written explicitly as [31]:

$$\text{dist}(\mathbf{z}, \boldsymbol{\mu}_k) = \sqrt{(\mathbf{z} - \boldsymbol{\mu}_k)^\top \boldsymbol{\Sigma}_k^{-1} (\mathbf{z} - \boldsymbol{\mu}_k)}, \quad (3.23)$$

which adapts to class covariance, normalizing feature scales and yielding a more reliable affinity measure than the Euclidean distance.

The theoretically optimal threshold τ_k^* is [31]:

$$\tau_k^* = \arg \max_{\tau_k \in (0, +\infty)} \left[\beta \Phi(X_K) + (1 - \beta) \Phi(X_U) \right], \quad (3.24)$$

where X_K and X_U denote known and unknown test sets, $\Phi(\cdot)$ is an evaluation metric (e.g., accuracy), and $\beta \in (0, 1)$ balances their contributions. Since X_U is unavailable in open-set training, we derive thresholds solely from known-class distances. For class k [31]:

$$P_k = \{ \text{dist}(\mathbf{z}_1, \boldsymbol{\mu}_k), \dots, \text{dist}(\mathbf{z}_{N_k}, \boldsymbol{\mu}_k) \}, \quad (3.25)$$

where N_k is the number of training samples in class k . Sorting P_k in descending order yields $\hat{P}_k = \{\varrho_k^1 \geq \dots \geq \varrho_k^{N_k}\}$. Applying the 3σ rule [31]:

$$\tau_k^* = \arg \min_{\varrho_k^{\hat{n}}} \hat{n} \cdot \mathbb{I}(\varrho_k^{\hat{n}} \geq 3\sigma_k), \quad \sigma_k = \text{std}(P_k), \quad (3.26)$$

where σ_k is the standard deviation of within-class distances and $\mathbb{I}(\cdot)$ is the indicator function. This yields a self-adaptive, class-wise threshold without relying on unknown samples.

Given a test embedding \mathbf{z}_t , classification is performed as [31]:

$$\hat{y}_t = \arg \min_{k \in \mathcal{C}_0} \text{dist}(\mathbf{z}_t, \boldsymbol{\mu}_k), \quad (3.27)$$

and \mathbf{z}_t is accepted as class k if it lies within some R_k ; otherwise it is rejected as unknown and routed to clustering.

3.5 Clustering

3.5.1 Clustering Validity Indices

Let \mathcal{U} be the set of samples rejected by the open set detector. This thesis defers feature concatenation, standardization, and dimensionality reduction specifics to Chapter 4. This thesis evaluates candidate cluster numbers $k \in \{0, 1, 2, \dots, k_{\max}\}$ and record quality measures for each. When $k = 0$, no clustering is performed and all samples are kept as a noise pool. When $k = 1$, all samples are treated as a single new class. When $k \geq 2$, this thesis runs both K-Means (with $n_{\text{init}} = 20$) and a **GMM**, and retain the better of the two according to a composite score defined below. For each valid solution, this thesis computes the inertia, the Silhouette coefficient S_k , the **CH** index CH_k , the **DB** index DB_k , the explained variance V_k and **Bayesian Information Criterion (BIC)**. The first main clustering validity is the composite score Q_k of the Silhouette coefficient S_k , the **CH** index CH_k , the **DB** index DB_k and the explained variance V_k . The second main clustering validity is the inertia. If the composite score Q_k and inertia which is also called the elbow method are not fit, **BIC** are computed for completeness but are *not* included in Q_k and **BIC** values are only available for **GMM** fits.

3.5.2 Clustering number k

The selection of k is done after the computation of the scores that are mentioned above. To begin with, the number of unknown samples are checked to adopt a small sample safeguard. Because when the rejected unknown pool is extremely small, it attempts a full model selection sweep over $k \geq 2$, which is statistically brittle and numerically unstable.

First, clustering validity indices used in our composite score Q_k (Silhouette, **CH**, **DB**) become unreliable with very few points. In particular, Silhouette requires at least two points per cluster, and **CH/DB** exhibit high variance

under tiny n . Second, K-Means does not define a likelihood function, so likelihood-based criteria (such as **BIC**) cannot be applied. In contrast, **GMM** are likelihood-based, so **BIC** can be used to compare them. However, for such criteria to be meaningful, the sample size must also be much larger than the number of free parameters, which means that sample size can not be too small. Third, elbow detection from the second-order difference of the inertia curve is ill posed at such small n because the curve is dominated by finite sample noise.

This rule not only avoids unstable multi-cluster partitions in the low sample regime, but also prevents degenerate validity scores, which gives robust handoff to the incremental learner. Beside, once more unknown data accumulate, the standard selection (Q_k maximization vs. elbow) for $k \geq 2$ is resumed.

When $|\mathcal{U}|$ is sufficiently large, the standard selection (Q_k maximization vs. elbow) for $k \geq 2$ is applied. It first detects an elbow candidate k_e from the inertia curve using the second order difference (curvature) of $\text{Inertia}(k)$, and in parallel find the score maximizer $k_s = \arg \max_{k \geq 2} Q_k$. After that, this thesis compares k_s with k_e in order to find the optimal k . Concrete thresholds and fallbacks are summarized in Chapter 4. Moreover, for the choosing k , some evaluations are done such as confusion analysis, per-cluster purity, confidence assignment, and label remapping, which are also described in Chapter 4.

3.6 Incremental Learning

3.6.1 Task Definition

This thesis studies class **IL** for the clustered unknown samples. It defines $\mathcal{Y}_{\text{old}} = \{0, \dots, C_{\text{old}} - 1\}$ as the set of known classes, and defines \mathcal{Y}_{new} as novel classes discovered from the unknown unlabeled data. At inference time, the model predicts over the joint label space \mathcal{Y} which means both known and unknown classes:

$$\mathcal{Y} = \mathcal{Y}_{\text{old}} \cup \mathcal{Y}_{\text{new}} \quad (3.28)$$

The thesis receives a labeled known class set $\mathcal{D}_{\text{old}} = \{(x_i, y_i)\}_{i=1}^{N_{\text{old}}}$ with $y_i \in \mathcal{Y}_{\text{old}}$, and an unlabeled pool $\mathcal{U} = \{u_j\}_{j=1}^{N_u}$ from which novel classes are mined. Base and expanded features are concatenated for each sample,

$$z = [\phi_{\text{base}}(x); \phi_{\text{exp}}(x)] \in \mathbb{R}^D \quad (3.29)$$

The goal is to train a model $f_\theta : \mathbb{R}^D \rightarrow \mathbb{R}^C$ with $C = C_{\text{old}} + C_{\text{new}}$ that maintains performance on \mathcal{Y}_{old} while learning \mathcal{Y}_{new} .

3.6.2 Quality Control Filter

After unsupervised clustering of unknown classes, a family of clusters is obtained:

$$\mathcal{K} = \{K_1, \dots, K_{K_{\text{max}}}\}. \quad (3.30)$$

For each cluster $K \in \mathcal{K}$ the system records cluster size, purity of every class and known class distribution. They can be denoted as:

1. Cluster size: $|K|$, i.e., the number of samples in K .
2. Purity: The percentage of dominant cluster in each clustered group. A higher $q(K)$ means the cluster is more internally consistent. This is a cluster homogeneity measure, not a prediction accuracy, accuracy is evaluated separately on held out data.

To suppress noisy clusters and extreme cluster sizes, only clusters that meet both purity and size constraints are retained:

$$\mathcal{K}_{\text{keep}} = \left\{ K \in \mathcal{K} \mid q(K) \geq \tau_p, |K| \in [s_{\text{min}}, s_{\text{max}}] \right\}. \quad (3.31)$$

τ_p is the minimum percentage of the purity in each cluster, which filters out highly mixed or ambiguous clusters. s_{min} is the minimum size of each cluster, which discards extremely small clusters that are often unstable or noisy. s_{max} is the maximum size of each cluster, which prevents oversized clusters from dominating training and reduces the risk of grouping multiple true classes together. The specified default values for the parameters are described more in Chapter 4. The purity here refers to the dominant label proportion within a cluster.

3.6.3 Test Data Per-Class Holdout

To avoid evaluation bias, the thesis uses per-class holdout split while testing, which means the test dataset is holdout, taken from the dataset before training the model so that it would not be involved in training. Each class first “reserves” a portion of its samples for testing, which is decided by the following equation. For each class c with n_c available samples (old or new),

the test size satisfies:

$$n_c^{\text{test}} \geq \max\left\{m_{\min}, \lceil r_{\text{test}} n_c \rceil\right\}. \quad (3.32)$$

where m_{\min} is a fixed floor and r_{test} is the proportional quota. The specified default values for the parameters are described more in Chapter 4. Only the pre-split training matrices (train_features, train_labels) are used for learning to guarantee no leakage from training data. This also stabilizes evaluation for small classes (which may be entirely held out when $n_c < m_{\min}$) so that small classes are still meaningful. Besides, it prevents large classes from dominating the metrics as well.

3.6.4 Training Data Replay Budget

Known class is called old class and clustered unknown class is called new class from now on. To balance old class rehearsal and new class learning, from the remaining pool, a training replay budget caps per-class training examples via old_max/new_max:

$$n_c^{\text{train}} \leq m_{\max}, \quad (3.33)$$

where m_{\max} is instantiated as old_max for known (old) classes and new_max for clustered unknown (new) classes. This way ensures neither old classes nor new classes cannot monopolize capacity, which yields a small, balanced training set. old_max is varied in experiments to study the trade-off between rehearsal of known classes and learning of clustered unknown classes while reducing compute and overfitting risk. In short, test data per-class holdout for independent, stable evaluation, training data Replay budget for fair, controllable training.

3.6.5 Network Architecture, Loss Functions, and Knowledge Distillation Gating

The student model f_θ consists of a lightweight feature extractor and two linear classification heads. The main head outputs logits over all known and newly discovered classes and is used during both training and inference, while the auxiliary head is only activated during the distillation stage to interface with a frozen teacher model trained on previous classes.

The optional teacher network g provides either (i) logits of old classes or (ii) intermediate feature embeddings. If the output dimensionalities between the

teacher and the student do not match (e.g., when the number of old classes changes), the system automatically switches from logit-level distillation to feature-level distillation to maintain compatibility.

During incremental training, a mini-batch \mathcal{B} from the replay buffer and new class data is used to optimize the total loss:

$$\mathcal{L}(\theta) = \frac{1}{|\mathcal{B}|} \sum_{(z,y) \in \mathcal{B}} \text{CE}(W_{\text{all}}^\top h_\theta(z), y) + \lambda \mathcal{L}_{\text{KD}}(\theta), \quad (3.34)$$

where the first term is the cross-entropy loss over all classes, and the second term represents the **Knowledge Distillation (KD)** regularizer weighted by λ .

Two **KD** variants are implemented. When the teacher and student heads are dimensionally compatible, the model performs *logit-level distillation*, aligning the student's old-class predictions to the teacher's softened outputs:

$$\mathcal{L}_{\text{KD}}^{\text{logits}} = \frac{T^2}{|\mathcal{B}_{\text{old}}|} \sum_{(z,y) \in \mathcal{B}_{\text{old}}} \text{KL}\left(\sigma(W_{\text{old}}^\top h_\theta(z)/T) \parallel \sigma(g(z)/T)\right), \quad (3.35)$$

where T is the temperature and σ is the softmax function.

If the teacher and student heads are incompatible, the system automatically falls back to *feature-level distillation*, which minimizes the feature distance between the student and teacher embeddings:

$$\mathcal{L}_{\text{KD}}^{\text{feat}} = \frac{1}{|\mathcal{B}|} \sum_{(z,y) \in \mathcal{B}} \|h_\theta(z) - h_t(z)\|_2^2. \quad (3.36)$$

A **KD** gate dynamically decides whether to apply distillation. The teacher's performance $\hat{\alpha}$ is evaluated on a held-out old-class validation set. If $\hat{\alpha} \geq \alpha_{\text{min}}$, the gate enables **KD** ($\lambda > 0$); otherwise, **KD** is disabled ($\lambda = 0$) to prevent negative transfer. When **KD** is off, the model is trained solely with the cross-entropy loss. The threshold α_{min} and evaluation procedure are described in Chapter 4.

As illustrated in Fig. 3.3, the left panel shows the student-teacher setup when **KD** is active, whereas the right panel depicts the baseline training when no

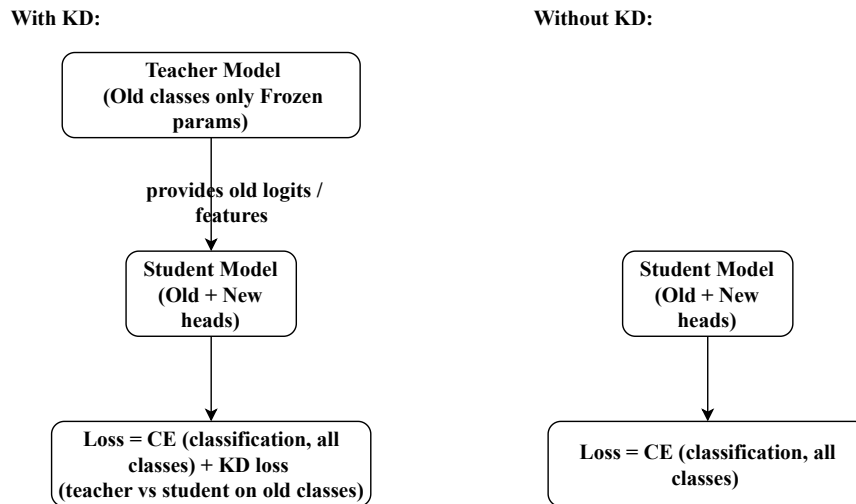


Figure 3.3: **KD** pipeline. Left: with teacher, the student is optimized with cross-entropy on all classes and **KD** alignment on old classes. Right: without teacher, the student is trained only with cross-entropy over all classes.

teacher is used. At inference, predictions are always made via the main head over the full label set, regardless of whether **KD** was used during training.

Chapter 4

Implementation and Parameters

4.1 Environment and Reproducibility

Softwares that are used in this thesis are Python, PyTorch, NumPy, scikit learn, SciPy, Matplotlib. The code sets global random seeds for NumPy and PyTorch to control sources of randomness such as weight initialization, shuffling, and dropout. **Principal Component Analysis (PCA)** and **GMM** use `random_state=42` so that randomized SVD and EM initialization produce repeatable factors and mixture starts. K-Means fixes `n_init=20` to average over multiple centroid initializations and may optionally set `random_state=42` for stricter determinism. These choices reduce run-to-run variance and make reported metrics reproducible. Enabling `cuda_deterministic=True` further constrains GPU kernels to deterministic implementations, which prevents small numeric differences between runs. Logging all seeds and configurations provides an audit trail so that a result can be exactly reproduced or diagnosed later. The current code does not force this setting in order to avoid potential performance slowdowns, but it can be turned on when strict reproducibility is required.

4.2 Data

This thesis evaluates on the a real world **RF** spectrogram dataset collected with a Universal Software Radio Peripheral (USRP) and released on IEEE DataPort as shown in **Table 4.1** [6]. The dataset contains 24 classes captured across two recording scenarios. Ten classes are recorded outdoors with drones operating at standoff distances of roughly 20–150 m from the receiver, while the remaining fourteen classes are recorded indoors where drones hover at

approximately 5 m from the receiver. The 24 classes span seven manufacturers, namely *DJI*, *VBar*, *FrSky*, *Futaba*, *Taranis*, *RadioLink*, and *SkyDroid*.

As shown in [Table 4.1](#), **RF** monitoring covers the three Industrial, Scientific and Medical (ISM) bands at 915 MHz, 2.4 GHz, and 5.8 GHz with a capture bandwidth of 100 MHz. Raw complex I/Q streams are sliced into short segments of 50 ms and transformed into **TFS** with a fixed canvas size of 512×512 for learning and evaluation. This fixed size rendering ensures cross session comparability and matches the input tensor shape expected by our **TE/PE** backbone.

Under folder `experiment_groups/`, there are `known_for_train`, `known_for_test`, and `unknown` which represent respectively training data from first 18 classes, test data from first 18 classes, and test data from last 6 classes considering unknown in the table [Table 4.2](#). Each line has a format of the path to a sample and corresponding label so the goal sample can be easily located and imported. After training the model according to [section 3.3](#) several contents are produced for clustering and incremental learning such as `test_X.npy`, `test_X_expand.npy`, `test_Y.npy`, and `label_hat.npy`. `test_X.npy` includes the base feature matrix (a.k.a. base semantics) for all test samples. `test_X_expand.npy` includes the expanded / multi-scale feature matrix for the same samples, which is aligned row for row with `test_X.npy`. `test_Y.npy` includes the ground truth class labels, which can be used to compute evaluation metrics such as accuracy, confusion metrics etc. `label_hat.npy` includes the prediction decisions obtained from **OSR**. It consists of predicted known class labels for accepted samples, and label with -1 for samples rejected as unknown. It further horizontally concatenates the features as `[test_X | test_X_expand]`, index rows where `label_hat = -1` to get the unknown feature matrix for clustering as described in [section 3.5](#), optionally standardize and reduce (e.g., **PCA**→64), and keep `test_Y` only for evaluation/diagnostics (it's not used to form clusters).

4.3 From Raw Radio Frequency to Fixed Size Time Frequency Spectrum

STFT is applied to the raw complex ?? stream using a Hamming window with **FFT** length $N_f = 2048$ and hop $H = 1024$ corresponding to 50% overlap.

Aspect	Setting (DroneRFb-Spectra)
Acquisition device	USRP (real world over the air capture)
Typical ranges	Outdoor 20–150 m; Indoor \approx 5 m
Manufacturers	DJI, VBar, FrSky, Futaba, Taranis, RadioLink, SkyDroid
ISM bands	915 MHz, 2.4 GHz, 5.8 GHz
Capture bandwidth	100 MHz
Segment length	50 ms (I/Q slice before STFT)
TFS size	512×512 (after rendering / resizing)
Documentation	See IEEE DataPort entry[6]

Table 4.1: Overview of the DroneRFb-Spectra dataset used in this thesis. [6]

Each cropped ?? segment is rendered as a spectrogram with a fixed size, and thereafter saved as a npy file, stored in a location that is described in files under folder `experiment_groups/`. In preprocessing, contiguous ?? streams are split into short segments that are 50 ms and each segment is downsampled to 512×512 . This ensures cross session comparability and a consistent tensor shape for the feature extractor. In order to normalize per sample spectrogram intensities produced by the **STFT**, it uses min-max normalization and maps values to $[0, 1]$ for each spectrogram independently.

$$\widehat{M} = \frac{M - \min(M)}{\max(M) - \min(M)}. \quad (4.1)$$

This per sample scaling removes global amplitude differences across sessions (e.g., gain, distance), yields comparable input ranges, and stabilizes training. The center frequency flag `fC_CH` takes 2440 or 5800 MHz to select the channel.

4.4 Network Implementation

As described in [section 3.3](#), the model (NET) consists of three modules: CNN branch(Texture Extractor), Transformer self attention branch(Position Extractor), Semantic fusion and classification branch.

Idx	Class	Idx	Class
0	Background, including WiFi and Bluetooth	12	DJI Phantom 4 Pro RTK
1	DJI Phantom 3	13	DJI MATRICE 30T
2	DJI Phantom 4 Pro	14	DJI AVATA
3	DJI MATRICE 200	15	DJI DIY
4	DJI MATRICE 100	16	DJI MATRICE 600 Pro
5	DJI Air 2S	17	VBar
6	DJI Mini 3 Pro	18	FrSky X20
7	DJI Inspire 2	19	Futaba T16IZ
8	DJI Mavic Pro	20	Taranis Plus
9	DJI Mini 2	21	RadioLink AT9S
10	DJI Mavic 3	22	Futaba T14SG
11	DJI MATRICE 300	23	Skydroid

Table 4.2: Class index list used in the dataset.

4.4.1 Texture Extractor

As shown in Fig. 4.1, this thesis adopts a multi-scale CNN branch composed of three parallel 2D convolutional streams with dilation factors $d \in \{1, 3, 5\}$ (encoder_d1, encoder_d3, encoder_d5). A dilated convolution slides a small kernel (base size 3×3) over the RF spectrogram while inserting $(d-1)$ holes between kernel taps; the effective receptive field becomes $k_{\text{eff}} = 3 + (3-1)(d-1)$ per axis [111, 112]. Thus $d=1$ yields a standard 3×3 kernel that focuses on local textures and edges, $d=3$ corresponds to an effective 7×7 field that captures larger shapes and mid-range dependencies, and $d=5$ gives an effective 11×11 field for broader context and global contours. Figure 4.2 illustrates the difference between small and dilated receptive fields (example for $d=1$ and $d=2$).*

*The example with $d=2$ is for illustration; this thesis uses $d \in \{1, 3, 5\}$.

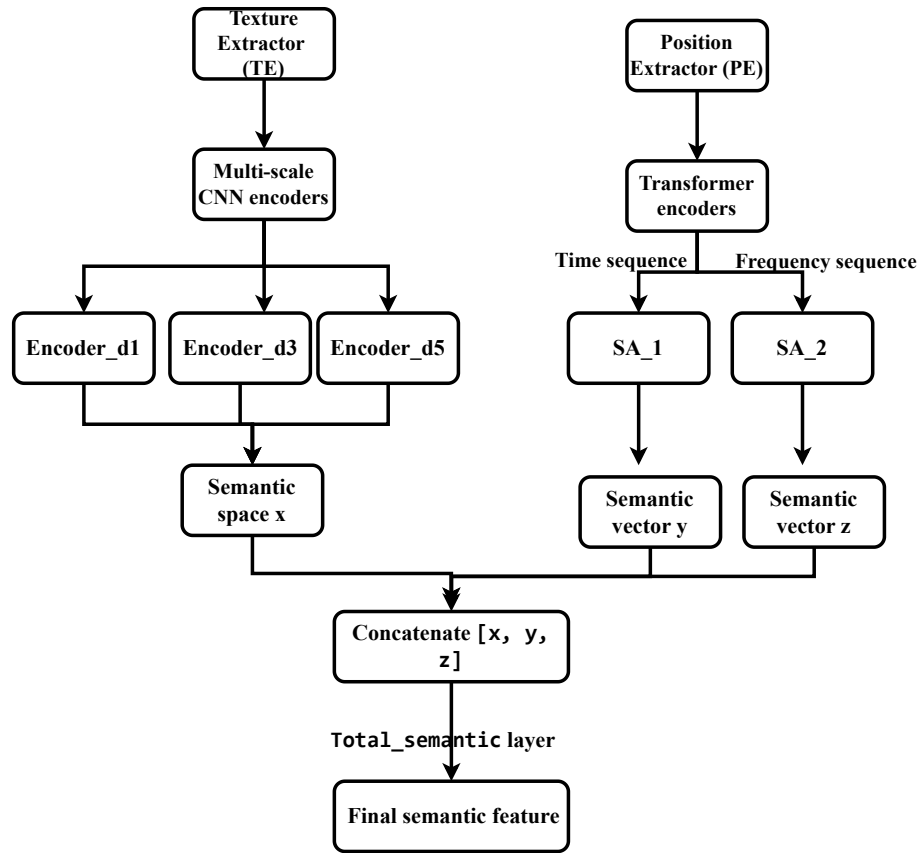


Figure 4.1: Overview of TE/PE + fusion

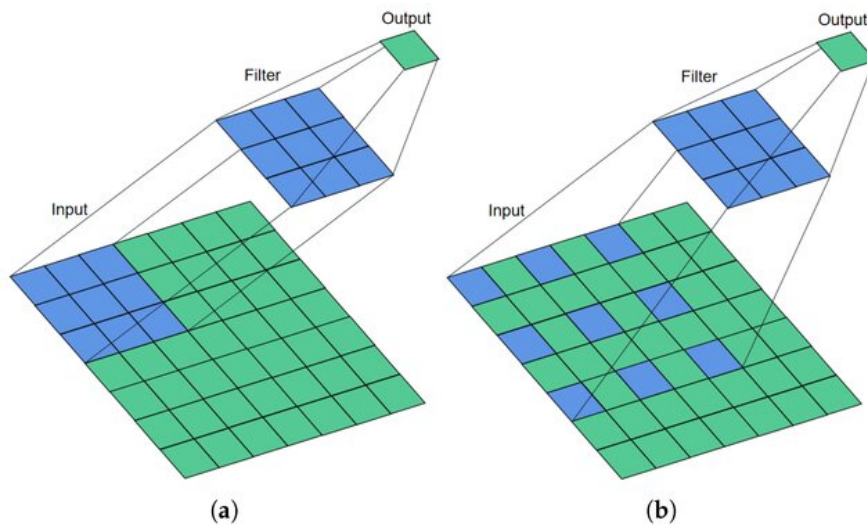


Figure 4.2: Dilated convolution with dilation rate $d=1$ and $d=2$ [112].

Each **TE** stream stacks Conv2d–BN–ReLU blocks interleaved with MaxPool2d (kernel 2, stride 2), and expands channels as

$$\text{out channels: } 4 \rightarrow 8 \rightarrow 16 \rightarrow 32 \rightarrow 64 \rightarrow 128.$$

Consequently, every stream produces a spatial feature map with 128 channels; the three maps are concatenated along the channel dimension, forming a $3 \times 128 = 384$ -channel multi-scale tensor that retains information from local detail to global structure. A final AvgPool2d followed by global average pooling (GAP) compresses each channel to its mean activation so that the output becomes a fixed-length vector suitable for a fully connected projection. The module `encoder_to_semantic` then maps this pooled multi-scale vector into the shared semantic space, which is later fused with the **PE** branch.

In summary, there are three parallel dilated convolution streams with dilation rates 1, 3, 5: `encoder_d1`, `encoder_d3`, `encoder_d5`. Each stream stacks Conv2d \rightarrow BatchNorm \rightarrow ReLU \rightarrow MaxPool2d (stride=2) blocks and increases channels as $4 \rightarrow 8 \rightarrow 16 \rightarrow 32 \rightarrow 64 \rightarrow 128$. The three feature maps are concatenated along the channel dimension before a global adaptive average pooling to 1×1 . The pooled vector is then projected to the semantic space by `encoder_to_semantic`. Because different dilations give different receptive fields (local to global textures on TFS), max pooling halves the spatial size to control compute, concatenation preserves multi-scale evidence, and global average pooling turns a variable spatial map into a fixed length descriptor for the classifier and for later fusion.

4.4.2 Position Extractor and Fusion

As shown in Fig. 4.1, this thesis uses two Transformer encoder stacks that operate separately along the time and frequency axes to capture long range dependencies that are complementary to the local, translation invariant cues learned by the dilated CNN in **TE**. 2 three-layer transformer encoders are applied separately to the time view Y and the frequency view Z are called SA_1 and SA_2. Before attention, `encoding_to_sa1` maps each time token from $512 \rightarrow 128 \rightarrow 64$ (so $d_{\text{model}}=64$), and `encoding_to_sa2` maps each frequency token from $512 \rightarrow 128 \rightarrow 64$. Concretely, the time token view $Y \in \mathbb{R}^{T \times W}$ and the frequency token view $Z \in \mathbb{R}^{W \times T}$ are first projected to 64 dimensional tokens by two linear MLPs with ReLU, then augmented with sinusoidal positional encodings on the corresponding axis. It means that sine and cosine positional codes are added via

`position_coding`. The outputs are flattened and projected to the semantic dimension by `sa1_to_semantic` and `sa2_to_semantic`, yielding two vectors, denoted \mathbf{y} (temporal semantics) and \mathbf{z} (spectral semantics).

The reason to split into time and frequency axes is to let attention model along one axis at a time, either temporal patterns or spectral patterns. Therefore to avoid the cost of full 2D attention. The tokenizer fixes token dimensionality to 64 so it matches the Transformer’s d_{model} . Moreover, positional coding provides order information that plain linear tokens would lack.

\mathbf{y} (temporal semantics) and \mathbf{z} (spectral semantics) are concatenated with the **TE** semantic vector \mathbf{x} to form $[\mathbf{x}; \mathbf{y}; \mathbf{z}] \in \mathbb{R}^{3d_s}$, which is passed through a two layer fusion MLP `total_semantic` ($3d_s \rightarrow d_s \rightarrow d_s$ with BatchNorm and ReLU). The first layer learns coarse cross branch weightings and interdependencies while reducing dimensionality to d_s , the second layer refines the fused embedding, introduces non linear feature interactions, suppresses residual noise and emphasizes highly discriminative components. This hierarchical fusion captures higher order correlations between the convolutional and Transformer branches while maintaining computational efficiency and training stability, thereby mitigating overfitting.

Besides the fused semantic, the model also keeps a pooled multi-scale vector `expand_x` (obtained by pooling the concatenated CNN feature map). This becomes the “expanded” feature in the saved arrays and is later concatenated with the base semantic in the clustering stage.

4.5 Open-Set Recognition

As shown in Fig. 4.3 and described in section 3.4, Open set decision uses Mahalanobis distance with per-class thresholds. During testing the code exports `label_hat`, the base semantic features for each test sample after fusion, which have values -1 or a known class index, and further simultaneously writes `test_X`, `test_X_expand`, `test_Y` for the clustering stage. `test_X` are the base semantic features for each test sample after fusion. `test_X_expand` are the expanded multi-scale CNN features (the pooled vector often called `expand_x`), and `test_Y` are the ground-truth labels for those test samples.

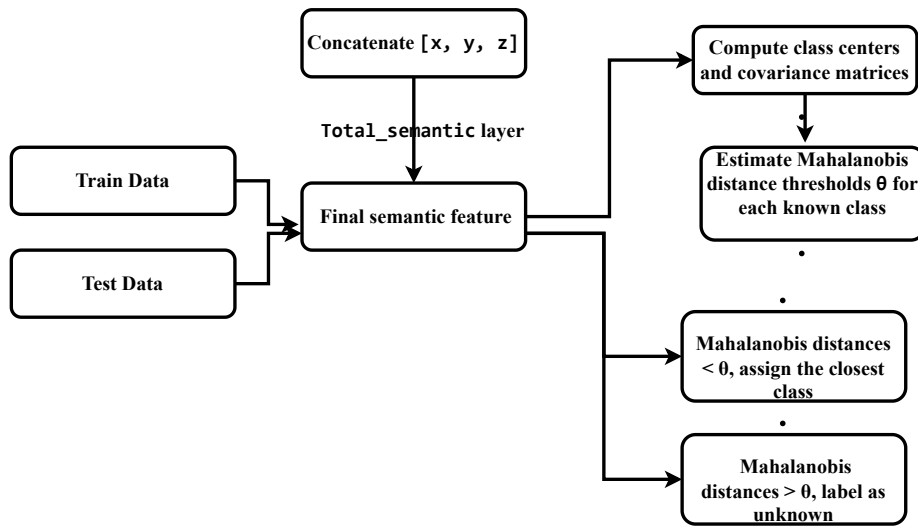


Figure 4.3: Overview of open set recognition

4.6 Clustering

After open-set rejection, it is left with a pool of unknown samples. As shown in Algorithm 1, the clustering engine decides whether these unknowns should (i) be ignored as noise, (ii) form one new class, or (iii) split into multiple new classes. How this is implemented is described more in details below.

Algorithm 1 Clustering for Unknown Samples ($k \in \{0, 1, \dots, k_{\max}\}$)

Input: Base features $\text{test_X} \in \mathbb{R}^{N \times d_1}$, expanded features $\text{test_X_expand} \in \mathbb{R}^{N \times d_2}$, labels test_Y , open-set tags label_hat ($-1 = \text{unknown}$), k_{\max} .

Output: Chosen k ; labels ℓ for unknowns; diagnostics (elbow, silhouette, composite, BIC).

- 1: **Step 1: Preprocessing & Unknown Extraction**
 Concatenate $Z \leftarrow [\text{test_X} \mid \text{test_X_expand}]$
 Unknown index $U \leftarrow \{i \mid \text{label_hat}_i = -1\}$; $Z_u \leftarrow Z[U, :]$
 $y_u \leftarrow \text{test_Y}[U]$; $Z_s \leftarrow \text{Standardize}(Z_u)$;
 $Z_p \leftarrow \begin{cases} \text{PCA}(Z_s, 64), & \dim(Z_s) > 64 \\ Z_s, & \text{otherwise} \end{cases}$
- 2: **Step 2: Score all k for $k = 0$ to k_{\max} do**
- 3: **if $k = 0$ then**
- 4: | No clustering: $\ell = -1$, inertia = 0, other scores \emptyset .
- 5: **else if $k = 1$ then**
- 6: | Single cluster: $\ell = 0$, global inertia, approximate BIC, other scores \emptyset .
- 7: **else**
- 8: **foreach $m \in \{\text{K - Means, GMM}\}$ do**
- 9: | $\ell_m \leftarrow m.\text{fit_predict}(Z)$;
- 10: | **if $\text{nUnique}(\ell_m) < k$ then**
- 11: | **continue;**
- 11: | $I \leftarrow \text{WCSS}(Z, \ell_m)$;
- 11: | $S \leftarrow \text{Silhouette}(Z, \ell_m)$;
- 11: | $CH \leftarrow \text{Calinski-Harabasz}(Z, \ell_m)$;
- 11: | $DB \leftarrow \text{Davies-Bouldin}(Z, \ell_m)$;
- 11: | $V \leftarrow 1 - I/\text{WCSS}_{\text{global}}(Z)$;
- 11: | $Q_k \leftarrow 0.4S + 0.3(CH/1000) + 0.2/(1+DB) + 0.1V$;
- 11: | keep labels/scores if Q_k is larger;
- 12: | Record inertia[k] $\leftarrow I$
- 13: **Step 3: Select k if $|Z_p| < 10$ then**
- 14: | $k \leftarrow (1 \text{ if } |Z_p| > 1 \text{ else } 0)$; **goto** Step 4.
- 15: Elbow candidate: use inertias for $k \geq 1$. **if fewer than 3 valid points then**
- 16: | $k_{\text{elbow}} = 1$
- 17: **else**
- 18: | $k_{\text{elbow}} = \text{find_elbow_point}$
- 19: Composite candidate: $k_{\text{score}} = \arg \max_{k \geq 2} Q_k$. **if no valid $k \geq 2$ then**
- 20: | $k = 1$ if $\text{BIC}(1) < \text{BIC}(0)$ else 0; **goto** Step 4.
- 21: **if $k_{\text{score}} \leq 2$ then**
- 22: | $k = k_{\text{score}}$
- 23: **else**
- 24: | **if $Q_{k_{\text{elbow}}} \geq 0.9 Q_{k_{\text{score}}}$ then**
- 25: | $k = k_{\text{elbow}}$
- 26: | **else**
- 27: | $k = k_{\text{score}}$
- 28: **Step 4: Output** $\ell = \text{results}[k].\text{labels}$; if $k=0$ set all -1 ; if $k=1$ set all 0.

Base semantics and multi-scale expanded features are concatenated column wise and an unknown mask selects rows whose predicted label is -1 to form \mathbf{F}_U . Features are standardized to zero mean and unit variance. If the dimensionality exceeds 64, **PCA** reduces to 64 components to improve stability and efficiency. As described in [section 3.5](#), there are several scoring and validity indices. For a fixed k define the within cluster sum of squares and total variance as

$$\text{Inertia}(k) = \sum_{c=1}^k \sum_{z \in C_c} \|z - \bar{z}_c\|_2^2, \quad \text{TV} = \sum_{z \in \mathbf{F}_U} \|z - \bar{z}\|_2^2 \quad (4.2)$$

and the explained variance $V_k = 1 - \text{Inertia}(k)/\text{TV}$. Moreover given a fitted model with maximized log-likelihood ℓ , number of parameters p , and sample size n , the two criteria are defined as:

$$\text{BIC} = -2\ell + p \log(n). \quad (4.3)$$

Besides, as described in [section 3.5](#), internal validation uses three widely adopted measures: the silhouette coefficient [[113–115](#)], the **CH** index [[114](#)], and the **DB** index [[115](#)], all computed with the scikit-learn implementation. The silhouette coefficient assesses how similar samples are to their assigned cluster compared to others, with larger values indicating better separation. The **CH** index is the ratio of between cluster dispersion to within cluster dispersion, where higher scores imply more distinct clusters. The **DB** index measures average cluster similarity, where lower values indicate superior clustering. These are fused into a composite score:

$$Q_k = 0.4 S_k + 0.3 \frac{CH_k}{1000} + 0.2 \frac{1}{1 + DB_k} + 0.1 V_k. \quad (4.4)$$

This combination is robust across data regimes. S_k emphasizes cohesion and separation. CH_k measures the ratio of between to within dispersion and is numerically scaled. DB_k is transformed as $1/(1 + DB_k)$ so that the larger DB_k is the better the model is. V_k captures variance explained.

The next step is to select the optimal k . When the number of unknown samples is less than 10, a small sample safeguard considers only $k \leq 1$. For $k \geq 2$ the code runs K-Means and **GMM**. K-Means using `n_init=20`, this means when running K-Means, the algorithm is started 20 times with different random initial centroids, obtain a clustering result each time, and then select

the one with the lowest inertia as the final result. Because K-Means is very sensitive to initial centroids, bad initial points can lead to poor local optima. Using multiple initializations can significantly improve stability and quality. Thereafter, this thesis compares Q_k obtained by K-Means and **GMM**, the better model by Q_k is kept. Inertia values are recorded for elbow analysis, and **BIC** are recorded for diagnostics. Let $k_s = \arg \max_{k \geq 2} Q_k$ and let k_e be the elbow candidate. If $Q_{k_e} \geq 0.9Q_{k_s}$ choose k_e to avoid overfitting, otherwise choose k_s . If all $k \geq 2$ solutions fail, fall back to comparing $k = 0$ and $k = 1$ with the single-Gaussian **GMM BIC** as a reference.

The next step is to do a quality analysis and packaging. When $k = 0$ the pool is kept as noise. When $k = 1$ the next available label is assigned to a single new class and each sample is given confidence 1.0. When $k \geq 2$ more than 1 clusters are obtained and in this thesis, the ground truth labels are given, therefore confusion matrix can be computed. Besides, for each cluster the purity, dominant class, and sample count are also computed and reported. Clusters are remapped to consecutive new class IDs $\{K, \dots, K + k - 1\}$. Each sample receives a confidence equal to the purity of its cluster. The model writes and saves two artifacts. The first one is a npy file that is a list of per-sample dictionaries with keys feature (standardized feature vector), original label, assigned label (the new class ID after cluster class remapping), cluster id (index of the discovered cluster), and confidence (the cluster's purity). The second one is also a npy file that is a dictionary that maps each cluster's dominant ground truth label to its corresponding new class ID, providing a reproducible cluster to class mapping for the incremental learner.

4.7 Quality Control Filter

As described in **subsection 3.6.2**, this thesis conducts a filter to control the qualities of new clustered groups so that incremental learning gets good quality input. The implementation uses the following defaults. The purity threshold $\tau_p = 0.6$ requires at least sixty percent of samples to agree with the dominant pseudo label which removes mixed and ambiguous clusters. The minimum size $s_{\min} = 10$ removes very small and unstable clusters. The maximum size $s_{\max} = 1000$ prevents very large clusters from dominating training. The program logs kept and removed counts and sample retention rates and then deterministically remaps kept clusters to new class labels.

4.8 Incremental Learning: Test Holdout and Replay

As described in [section 3.6](#) per-class split for test holdout. A test holdout policy is adopted for both new and old classes. For each class c with n_c samples the holdout size satisfies

$$n_c^{\text{test}} \geq \max\left\{m_{\min}, \lceil r_{\text{test}} n_c \rceil\right\}, \quad (4.5)$$

with defaults $m_{\min} = 20$ and $r_{\text{test}} = 0.3$. The fixed floor m_{\min} prevents small classes from having too few test points, while the proportional term r_{test} yields a familiar 70/30 train–test split for medium and large classes so that the test set remains representative.

The reasons of choosing $m_{\min}=20$ and $r_{\text{test}}=0.3$ are explained below. Evaluation is done per class, therefore if a class contributes only a handful of test samples, its measured accuracy will fluctuate heavily. The lower bound $m_{\min} = 20$ is guided by the sampling variability of a binomial proportion. Testing can be considered as estimating the head and tail probability of a coin. With very few tosses the estimate is noisy, with more tosses it stabilizes. Formally, the standard error of a test accuracy \hat{p} based on n independent trials is:

$$\text{SE}(\hat{p}) = \sqrt{\hat{p}(1 - \hat{p})/n} \quad (4.6)$$

As described in [\[116\]](#), it is maximized at $\hat{p} \approx 0.5$, giving $\text{SE}_{\max} = \sqrt{0.25/n} = 0.5/\sqrt{n}$. For $n=5$ this is ≈ 0.223 (a 95% interval of roughly $\pm 44\%$), for $n=10$ it is ≈ 0.158 ($\pm 31\%$), and for $n=20$ it is ≈ 0.111 ($\pm 22\%$). Although not “tight,” $n=20$ already avoids purely luck-driven evaluations while still leaving enough samples for training. Hence a per–class floor $m_{\min}=20$ ensures every class retains a minimally reliable, independent test set. A fixed floor alone is not sufficient for large classes: taking only 20 points from a large class would under-represent it. Therefore a proportional reservation $r_{\text{test}}=0.3$ is added, which corresponds to the commonly used 70/30 train–test split and lets the test set scale with class size. In short, the floor protects small classes from unstable estimates, and the 30% quota keeps medium and large classes representative [\[116, 117\]](#).

Typical examples on different amount of samples are explained following. Class A has 200 samples and reserves at least 60 for test. Class B has 30

samples and reserves at least 20 for test. Class C has 8 samples, the rule requests 20 but only 8 exist, so all 8 go to test and none to train. Only the pre-split `train_features` and `train_labels` are used for learning in the process of training, which confirms there is no leakage from the training data while testing.

After reserving the holdout and removing the test holdout, the remaining samples are sent to training data but limited by a training replay budget per class in case the remaining amount of samples is too large:

$$n_c^{\text{train}} \leq m_{\text{max}}, \quad (4.7)$$

where m_{max} is instantiated as `old_max` for known classes and `new_max` for novel classes. This prevents overrepresented classes from monopolizing capacity, produces a small and balanced training set, and provides a clear knob to study the trade-off between rehearsal of known classes and learning of newly discovered classes. In the experiments `old_max` is set to 0, 5, or 10 to study the rehearsal versus new class learning trade-off and `new_max` is fixed at 60. This produces a smaller and more balanced training set and reduces overfitting and computation, which delivers independent and stable evaluation across all classes and fair, controllable training dynamics for incremental learning.

4.9 Student Teacher KD and Gate

The student model `ExpandedNetwork` is a three-layer MLP with hidden sizes 256 and 128 and dropout 0.3. The joint head $W_{\text{all}} \in \mathbb{R}^{128 \times C}$ outputs logits over the full label space. The old-class head $W_{\text{old}} \in \mathbb{R}^{128 \times C_{\text{old}}}$ is used only for distillation. Input features are aligned to the student input dimension by zero padding or truncation and in this run the student input dimension is $D = 512$. When a teacher is provided the model checks head dimensions and switches to feature distillation if heads are incompatible, while the teacher is kept frozen. Distillation is gated and enabled only if the teacher accuracy on the old-class holdout reaches $\hat{\alpha} \geq \alpha_{\text{min}}$ with default $\alpha_{\text{min}} = 0.7$. If the gate fails KD is disabled and the total loss reduces to cross entropy with $\lambda = 0$. The total loss is CE plus $\lambda \mathcal{L}_{\text{KD}}$. Logits distillation uses temperature softened KL on old-class samples with a factor T^2 . Feature distillation uses mean squared error with dimension alignment as needed.

4.10 Summary of Default Hyperparameters

Component	Default
Cluster filter	purity $\tau_p=0.6$; size range $[s_{\min}, s_{\max}] = [10, 1000]$
Holdout split	test_ratio= 0.3; min_test= 20 per class
Replay caps	old_max $\in \{0, 5, 10\}$, new_max= 60

Table 4.3: Default hyperparameters and training setup.

Chapter 5

Results

This chapter presents the experimental results that evaluate the effectiveness and robustness of the proposed method. The analysis of the results is organized into three parts: accuracy of known/unknown classification, accuracy of clustering of new categories, and accuracy of incremental learning, which corresponds to step 4 “Recognition of known and unknown classes”, 5 “Clustering to discover new categories” and 6 “Incremental learning of newly discovered classes” in Figure 5.1.

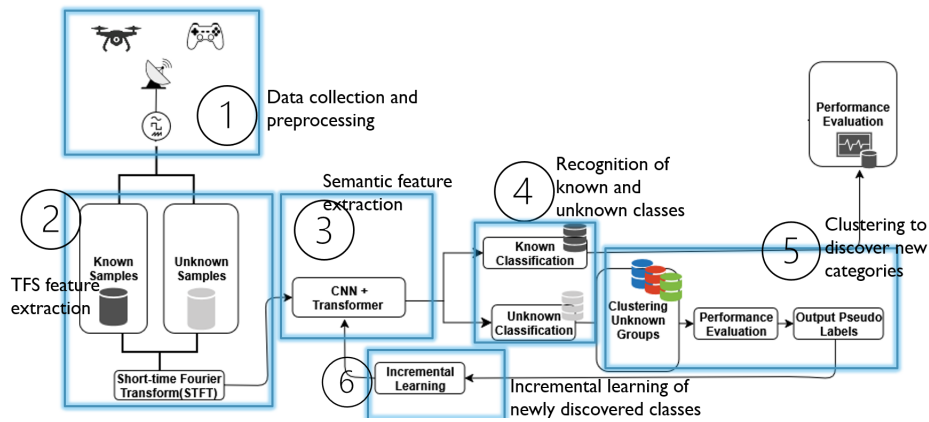


Figure 5.1: Overall system model for drone classification and incremental learning.

First, this work assesses step 4 in Figure 5.1, the method under the *OSR* scenario, where unknown classes appear during testing. The embedding space is visualized using t-SNE, and the classification quality is analyzed through confusion matrices and prediction type distributions. Second, this study reports the clustering results, which shows how the optimal number of clusters

is determined and providing both quantitative and qualitative evaluations of cluster purity and consistency with ground truth labels. Finally, the step 4 in Figure 5.1, **IL** performance is examined, focusing on the balance between retaining knowledge of old classes and acquiring new ones under different replay budgets. Together, these results provide a comprehensive understanding of the method’s performance in challenging recognition, clustering, and incremental learning settings.

5.1 Open-Set Recognition Result

This section is to evaluate the performance of the step 4 “Recognition of known and unknown classes” in Figure 5.1, which is also the result under **OSR** scenario. We have obtained good accuracies over known samples a.k.a. from class 0 to class 17. Moreover, we have successfully recognized most of the unknown samples a.k.a. from class 18 to class 23. More details are described below.

We first evaluate the classification quality using the normalized confusion matrix (Figure 5.2) and the prediction type distribution (Figure 5.3). The confusion matrix shows strong diagonal dominance, while the prediction type analysis demonstrates that most samples are correctly identified under the **CSR** scenario.

Overall, these visualizations confirm that the proposed approach not only achieves high clustering quality in the closed set setting but also generalizes effectively to the **OSR** scenario, where unknown classes are accurately identified and isolated.

5.2 Clustering Result

This section is to evaluate the performance of the step 5 “Clustering to discover new categories” in Figure 5.1. There are 7 new categories that have been successfully discovered. More details are described below. To determine the optimal number of clusters, this thesis first compares the Elbow Method and the Composite Score Method, as shown in Figure 5.4. The Elbow Method suggested $k = 4$ based on diminishing returns in inertia, while the Composite Score Method identified the global maximum at $k = 7$. In order to further choose the optimal k , a comparison is done according to implementation of clustering 4.6 and Algorithm 1. If $Q_{k_e} \geq 0.9Q_{k_s}$ choose k_e to avoid

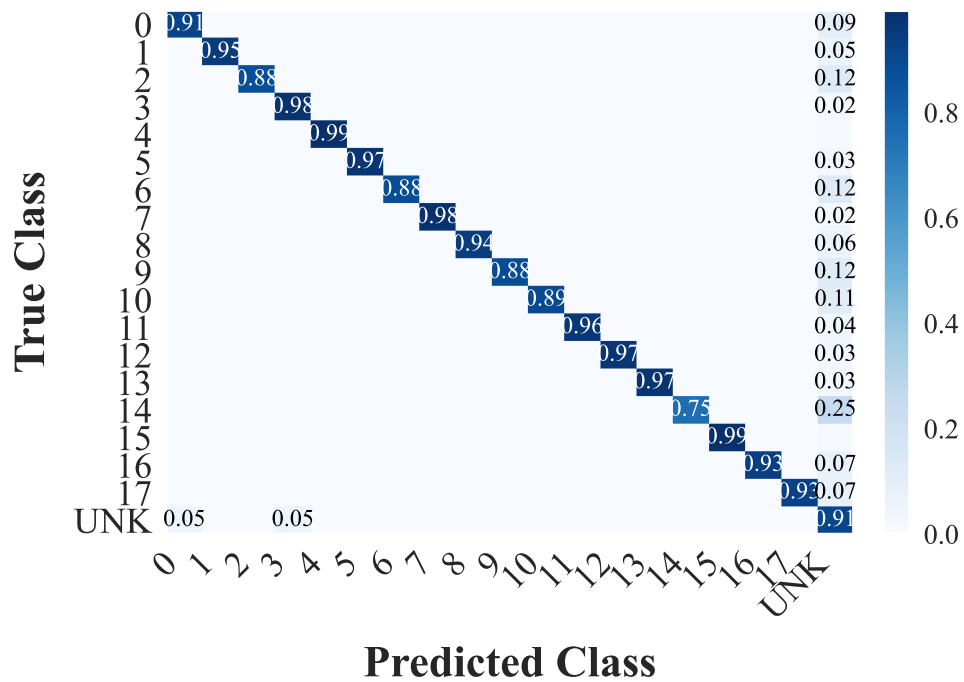


Figure 5.2: Normalized confusion matrix under open-set testing.

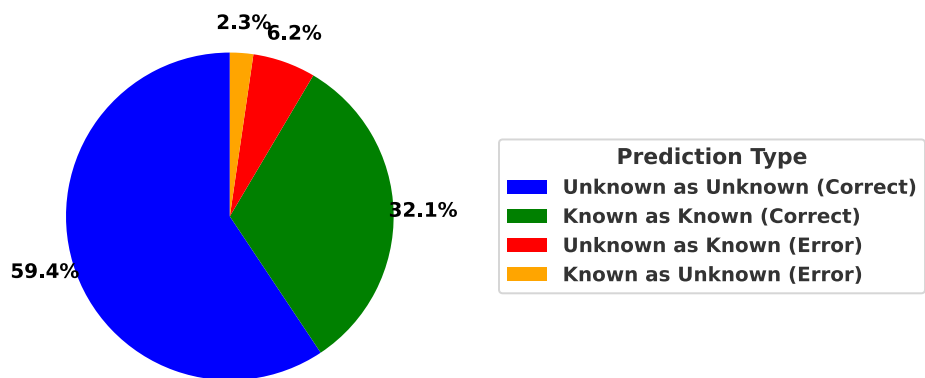


Figure 5.3: Prediction type distribution in **OSR**. The majority of samples are correctly identified as either known or unknown, while errors remain limited.

overfitting, otherwise choose k_s . As shown in Table 5.1 this thesis compares the composite scores $Q_{\text{elbow}} = 0.5876$ and $Q_{\text{score}} = 0.7685$. Since

$$Q_{\text{elbow}} = 0.5876 < 0.9 \times Q_{\text{score}} = 0.69165,$$

the condition for adopting k_{elbow} is not satisfied. Therefore, the final number of clusters is determined as

$$K = k_{\text{score}} = 7.$$

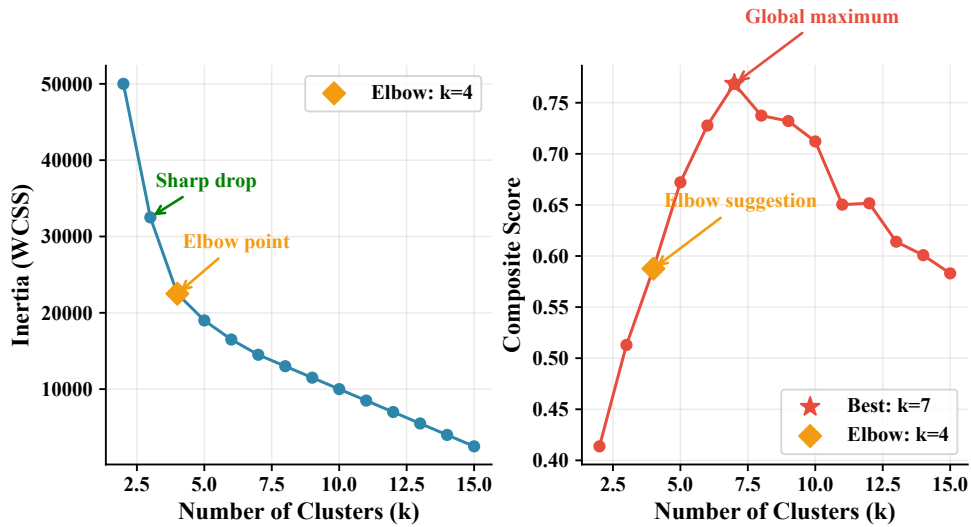


Figure 5.4: Comparison between the Elbow Method and the Composite Score Method for determining the optimal number of clusters. The Elbow Method suggests $k = 4$ based on diminishing returns in inertia, while the Composite Score Method identifies the global maximum at $k = 7$, representing the optimal choice.

Given the amount of clusters are 7, this thesis further analyzes the resulting seven clusters in detail. Figure 5.5 presents per cluster purity distributions along with qualitative ratings. Cluster 2 and Cluster 6 achieve purities of 99.45% and 100%, respectively, rated as *Excellent*. Cluster 1, Cluster 3, Cluster 4, and Cluster 5 all exceed 90% purity and are rated as *High*, while Cluster 0 shows only 57.86% purity and is considered *Moderate*. These results indicate that most clusters exhibit strong intra class consistency, with only one cluster showing significant mixing.

Figure 5.6 (a) shows the clustering results of unknown samples, where colors indicate the assigned clusters. In contrast, Figure 5.6 (b) presents the same

Table 5.1: Detailed comparison of clustering results ($k = 2$ to $k = 15$). The row highlighted in red indicates the selected optimal k .

k Value	Silhouette	Composite Score	Method	Rank
2	0.2356	0.4137	K-Means	14
3	0.2844	0.5129	GMM	13
4	0.3523	0.5876	K-Means	11
5	0.4092	0.6721	K-Means	6
6	0.4447	0.7277	K-Means	4
7	0.4440	0.7685	K-Means	1
8	0.4176	0.7374	K-Means	2
9	0.4202	0.7321	K-Means	3
10	0.4115	0.7121	K-Means	5
11	0.3633	0.6503	K-Means	8
12	0.3770	0.6516	K-Means	7
13	0.3218	0.6140	K-Means	9
14	0.3212	0.5896	K-Means	10
15	0.3209	0.5830	GMM	12

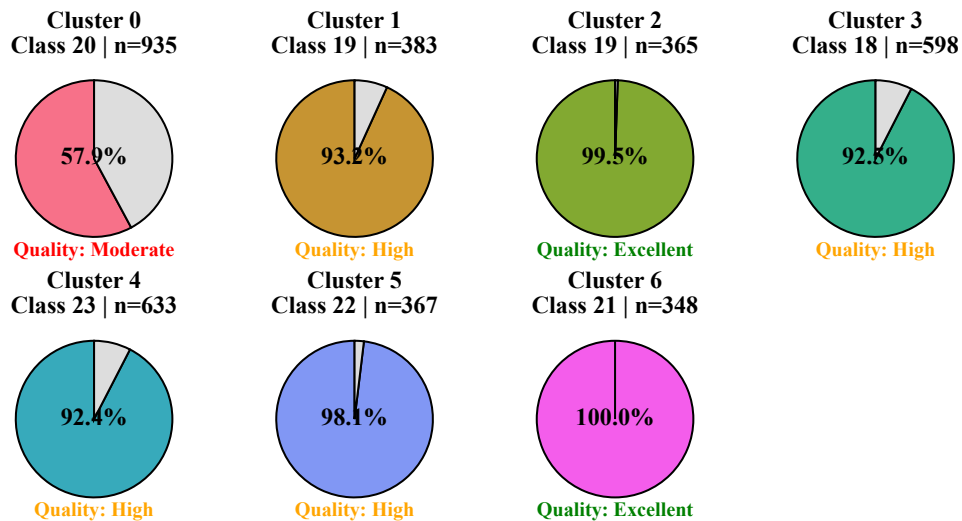


Figure 5.5: Per cluster purity pies with sample counts and qualitative ratings.

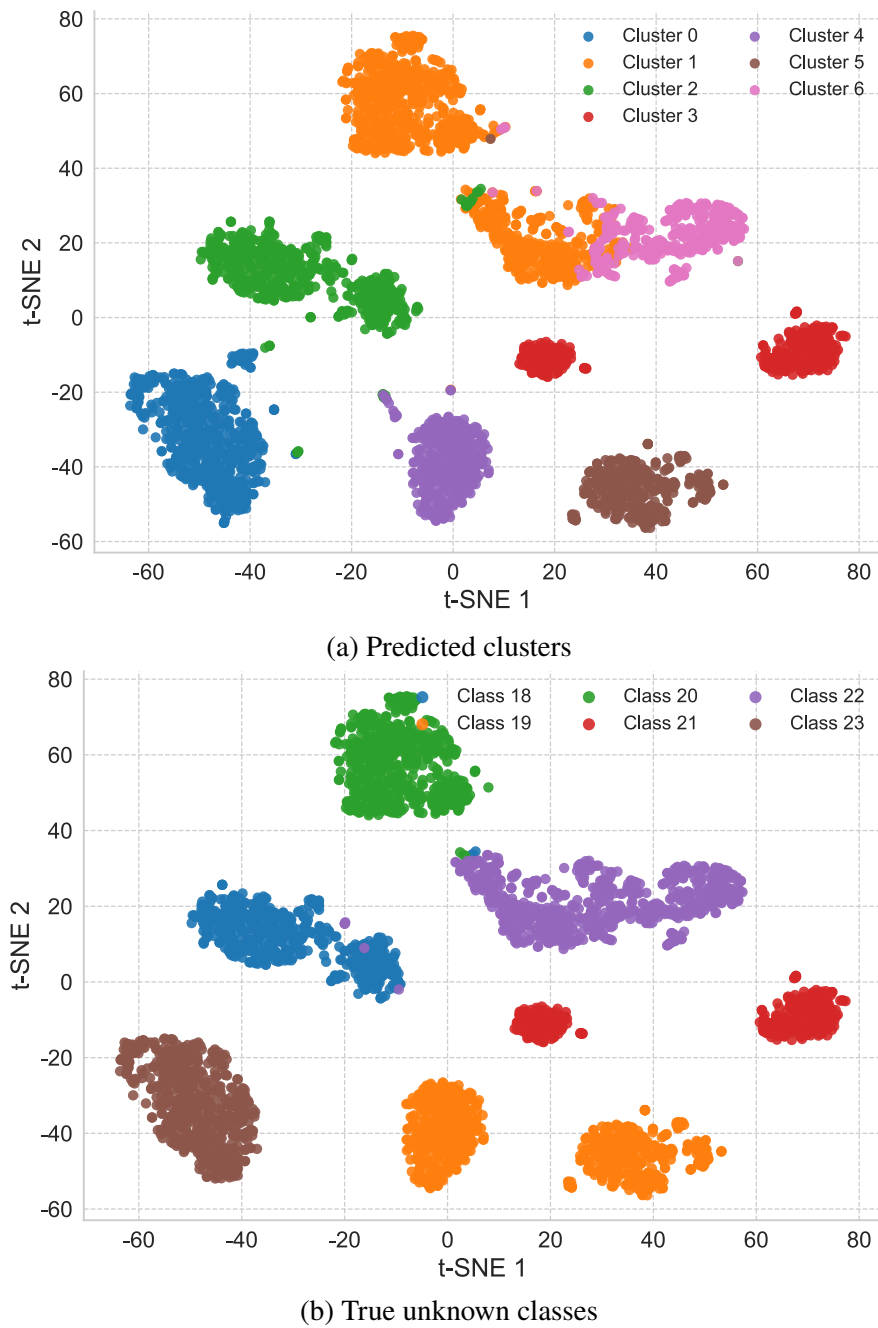


Figure 5.6: t-SNE visualization of (a) predicted clusters and (b) true unknown classes. Strong correspondence validates semantic consistency of clustering.

unknown samples but colored by their true classes. The strong alignment between predicted clusters and true labels demonstrates that the clustering not

only separates unknown samples but also preserves their semantic consistency.

The alignment between clusters and their dominant true classes is further illustrated in Figure 5.7, where each cluster maps cleanly to a single class with varying sample sizes. Cluster 1 and 2 have both class 19 as dominant true class, which potentially because class 19 are over clustered since it has different operational modes such as hovering, recording etc.

Figure 5.8 provides a complementary view, plotting purity against sample count. All clusters except Cluster 0 concentrate in the high purity region.

Finally, Figure 5.9 summarizes the purity distribution using bar plots and threshold lines. Six out of seven clusters exceed the high quality threshold ($\geq 90\%$), and two of them reach or approach the excellent threshold ($\geq 99\%$). These findings confirm that the chosen clustering configuration ($k = 7$) not only achieves high quantitative performance but also aligns well with ground truth class structure, demonstrating the robustness of the clustering method.

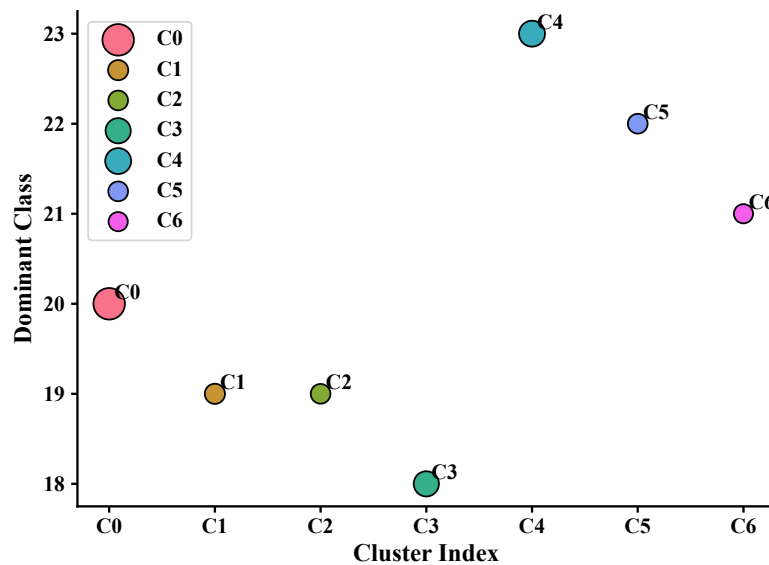


Figure 5.7: Cluster \rightarrow dominant *true* class mapping with sample counts.

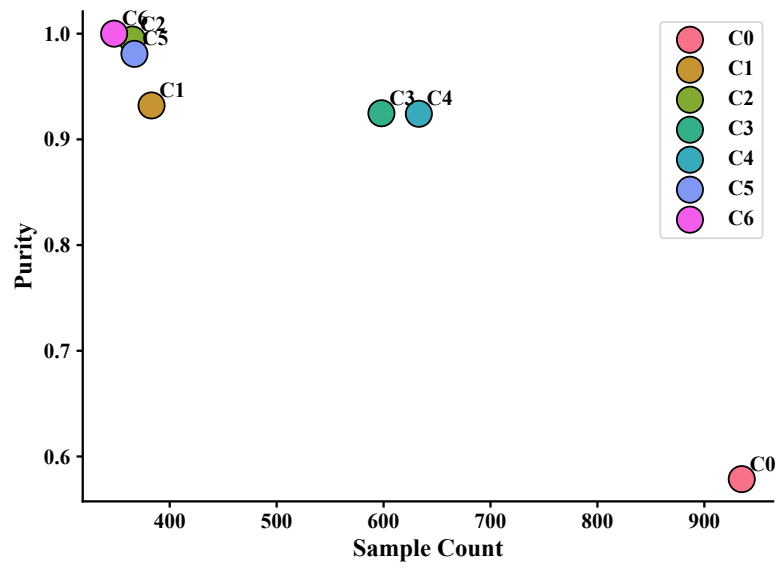


Figure 5.8: Purity against sample count.

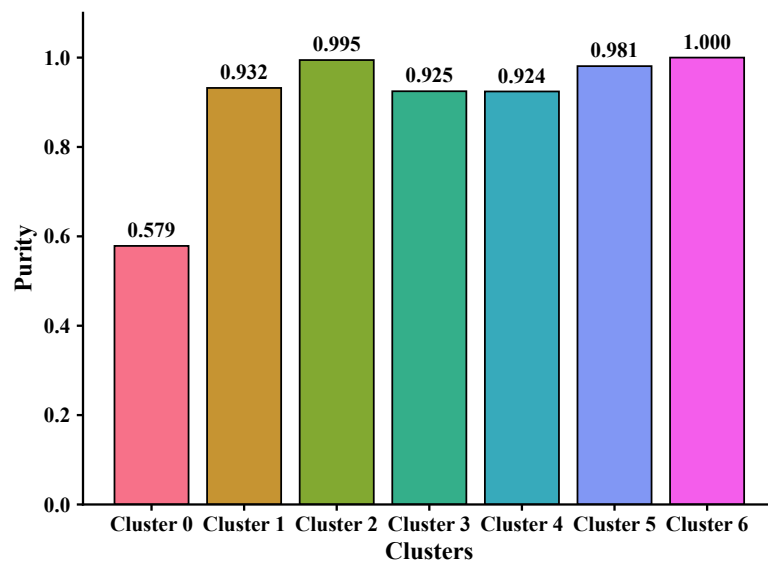


Figure 5.9: Cluster purity bars.

5.3 Incremental Learning Result

This section is to evaluate the performance of the step 6 “Incremental learning of newly discovered classes” in Figure 5.1. The new discovered classes have been successfully incrementally learned by the model and does not forget the old classes at the same time by replaying a small amount of old classes. More details are described below. Table 5.2 summarizes the holdout accuracies on old and new classes, and Figure 5.10 plots the same values against the replay budget. As shown in the table and figure, the results are obvious:

Replay cap	Train size	Old (HOLDOUT)	New (HOLDOUT)
<code>old_max= 0</code>	(360, 512)	0.00%	98.66%
<code>old_max= 5</code>	(450, 512)	100.00%	98.10%
<code>old_max= 10</code>	(540, 512)	100.00%	98.49%

Table 5.2: Effect of the old-class replay budget under the holdout protocol (KD gated off).

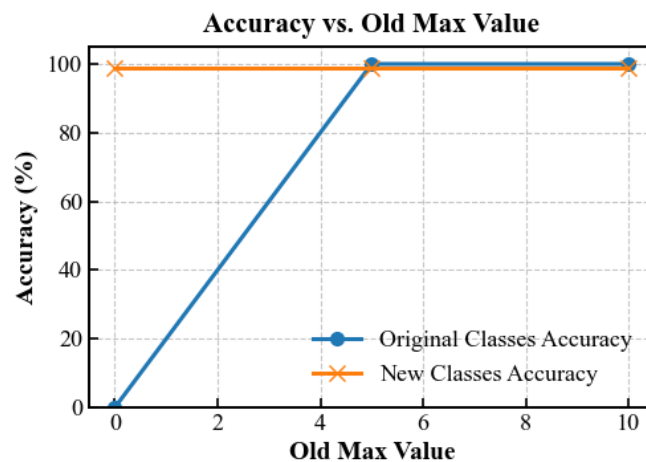


Figure 5.10: Accuracy on holdout sets vs. old class replay cap (`old_max`). Old class accuracy exhibits catastrophic forgetting at 0 and recovers fully with small replay, new class accuracy remains $\approx 98\%$ across budgets.

- **Catastrophic forgetting without replay.** With `old_max= 0`, the student achieves 0% on old classes while maintaining high new class

accuracy (98.66%). This confirms that, in the absence of **KD** and replay, fitting only the novel classes erases knowledge of the previous ones.

- **Small replay fully restores the old classes.** Increasing replay to `old_max= 5` suffices to recover *perfect* holdout accuracy on the old classes (100%) while keeping new-class accuracy essentially unchanged (98.10%).
- **Diminishing returns beyond 5 samples/class.** Raising the replay budget to `old_max= 10` maintains 100% on old classes and slightly changes new class accuracy (98.49%). The curve is flat, indicating limited interference from modest replay.

Two implementation choices appear crucial for the stability observed at `old_max ∈ {5, 10}`: (i) *feature alignment*. Pad or truncate features prevents distributional drift between old and new features at the student input and (ii) *balanced mini batching* with a strict per class limitation avoids biasing the shared head toward classes with larger pools. Together these choices allow a small replay budget to anchor the old decision boundaries while the head expands to the novel classes.

Because the teacher underperforms the accuracy gate on old classes and the single student model with replay performs well, **KD** is deactivated by design. Moreover, the ablation also isolates the effect of replay alone. This behavior illustrates the intended safety mechanism: when a teacher is unreliable, the system avoids negative transfer and relies on replay to stabilize old knowledge.

In this thesis, a replay first, **KD** optional strategy is sufficient: a budget of 5–10 old samples per class restores old performance on holdout with negligible impact on novel classes. When a reliable teacher is available (passing the gate), **KD** can be re-enabled to further contribute in the model.

Chapter 6

Conclusion

This work proposes a unified framework for clustering, **OSR**, and **IL**, addressing key challenges in handling unknown classes and adapting to evolving data distributions. The method is extensively evaluated through three perspectives.

First, under the **OSR** scenario, the approach demonstrates its ability to separate unseen classes while maintaining semantic consistency. Visualizations based on t-SNE projections, confusion matrices, and prediction type distributions confirm that unknown samples are effectively identified, rejected and isolated while known samples are effectively recognized.

Second, clustering analysis shows that the proposed method achieves high quality partitions of the data. The combination of composite score method and elbow method determine the optimal number of clusters, and both purity metrics and qualitative evaluations confirmed strong alignment with ground truth classes. Most clusters exceeded the high purity threshold, validating the method's capability to capture intrinsic class structure.

Third, in the **IL** setting, the approach effectively mitigates catastrophic forgetting. Experiments reveal that even with a small replay budget, old class knowledge could be fully preserved while new class learning remained stable. This demonstrates that the replay first strategy, combined with careful feature alignment and balanced mini batching, provides a practical solution for continual adaptation.

Overall, this study shows that the proposed framework not only performs

competitively in **CSR** but also generalizes reliably to **OSR**, clustering and **IL** scenarios. These findings suggest that the method provides a robust foundation for real world applications where data evolves over time and previously unseen categories are common. Future directions include extending the approach to large scale datasets, integrating knowledge distillation under reliable teachers, and exploring its use in multimodal learning environments.

Bibliography

- [1] S. Hu, W. Ni, X. Wang, A. Jamalipour, and D. Ta, “Joint optimization of trajectory, propulsion, and thrust powers for covert uav-on-uav video tracking and surveillance”, *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 1959–1972, 2020.
- [2] X. Shi, C. Yang, W. Xie, C. Liang, Z. Shi, and J. Chen, “Anti-drone system with multiple surveillance technologies: architecture, implementation, and challenges”, *IEEE Communications Magazine*, vol. 56, no. 4, pp. 68–74, 2018.
- [3] Y. Zhang, “Rf-based drone detection using machine learning”, in *2021 2nd International Conference on Computing and Data Science (CDS)*, IEEE, 2021, pp. 425–428.
- [4] O. O. Medaiyese, A. Syed, and A. P. Lauf, “Machine learning framework for rf-based drone detection and identification system”, in *2021 2nd International Conference On Smart Cities, Automation & Intelligent Computing Systems (ICON-SONICS)*, IEEE, 2021, pp. 58–64.
- [5] B. Taha and A. Shoufan, “Machine learning-based drone detection and classification: state-of-the-art in research”, *IEEE access*, vol. 7, pp. 138 669–138 682, 2019.
- [6] N. Yu, *Dronerfb-spectra: a rf spectrogram dataset for drone recognition*, 2024. DOI: [10.21227/wv7h-sv64](https://doi.org/10.21227/wv7h-sv64). [Online]. Available: <https://dx.doi.org/10.21227/wv7h-sv64>.
- [7] E. Vinogradov, H. Sallouha, S. De Bast, M. M. Azari, and S. Pollin, “Tutorial on uavs: a blue sky view on wireless communication”, *Journal of Mobile Multimedia*, vol. 14, no. 4, pp. 395–468, 2018.
- [8] S. Islam, “Drones on the rise: exploring the current and future potential of uavs”, *arXiv preprint arXiv:2304.13702*, 2023.

- [9] M. Ozger *et al.*, “6g for connected sky: a vision for integrating terrestrial and non-terrestrial networks”, in *2023 Joint European Conference on Networks and Communications 6G Summit (EuCNC/6G Summit)*, 2023, pp. 711–716.
- [10] A. E. Garcia *et al.*, “Direct air-to-ground communications for flying vehicles: measurement and scaling study for 5g”, in *2019 IEEE 2nd 5G World Forum (5GWF)*, 2019, pp. 310–315.
- [11] M. Ozger, M. Vondra, and C. Cavdar, “Towards beyond visual line of sight piloting of uavs with ultra reliable low latency communication”, in *2018 IEEE Global Communications Conference (GLOBECOM)*, 2018, pp. 1–6.
- [12] P. Wang, M. Ozger, C. Cavdar, and M. Petrova, “Beyond visual line of sight piloting of uavs using millimeter-wave cellular networks”, in *2019 IEEE 30th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, 2019, pp. 1–7.
- [13] F. Salehi, M. Ozger, N. Neda, and C. Cavdar, “Ultra-reliable low-latency communication for aerial vehicles via multi-connectivity”, in *2022 Joint European Conference on Networks and Communications 6G Summit (EuCNC/6G Summit)*, 2022, pp. 166–171.
- [14] F. Salehi, M. Ozger, and C. Cavdar, “Reliability and delay analysis of 3-dimensional networks with multi-connectivity: satellite, haps, and cellular communications”, *IEEE Transactions on Network and Service Management*, vol. 21, no. 1, pp. 437–450, 2024.
- [15] I. A. Meer, M. Ozger, M. Lundmark, K. W. Sung, and C. Cavdar, “Ground based sense and avoid system for air traffic management”, in *2019 IEEE 30th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, 2019, pp. 1–6.
- [16] Y. Deng, I. A. Meer, S. Zhang, M. Ozger, and C. Cavdar, “D3qn-based trajectory and handover management for uavs co-existing with terrestrial users”, in *2023 21st International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt)*, 2023, pp. 103–110.
- [17] Y. Deng, S. Zhang, I. A. Meer, M. Ozger, and C. Cavdar, “Joint trajectory and handover management for uavs co-existing with terrestrial users: a multi-agent drl approach”, *IEEE Transactions on Cognitive Communications and Networking*, 2025.

- [18] I. A. Meer, K.-L. Besser, M. Ozger, D. Schupke, H. V. Poor, and C. Cavdar, “Learning-based dynamic cluster reconfiguration for uav mobility management with 3d beamforming”, in *2024 IEEE International Conference on Machine Learning for Communication and Networking (ICMLCN)*, 2024, pp. 486–491.
- [19] I. A. Meer, M. Ozger, D. A. Schupke, and C. Cavdar, “Mobility management for cellular-connected uavs: model-based versus learning-based approaches for service availability”, *IEEE Transactions on Network and Service Management*, vol. 21, no. 2, pp. 2125–2139, 2024.
- [20] I. A. Meer, K.-L. Besser, M. Ozger, H. V. Poor, and C. Cavdar, “Reinforcement learning-based dynamic power control for uav mobility management”, in *2023 57th Asilomar Conference on Signals, Systems, and Computers*, 2023, pp. 724–728.
- [21] I. A. Meer *et al.*, “Explainable ai for uav mobility management: a deep q-network approach for handover minimization”, *arXiv preprint arXiv:2504.18371*, 2025.
- [22] I. A. Meer, K.-L. Besser, M. Ozger, D. Schupke, H. V. Poor, and C. Cavdar, “Hierarchical multi-agent drl-based dynamic cluster reconfiguration for uav mobility management”, *arXiv preprint arXiv:2412.16167*, 2024.
- [23] J. Chen, M. Ozger, and C. Cavdar, “Nash soft actor-critic leo satellite handover management algorithm for flying vehicles”, in *2024 IEEE International Conference on Machine Learning for Communication and Networking (ICMLCN)*, 2024.
- [24] V. Megas, S. Hoppe, M. Ozger, D. Schupke, and C. Cavdar, “A combined topology formation and rate allocation algorithm for aeronautical ad hoc networks”, *IEEE Transactions on Mobile Computing*, vol. 23, no. 1, pp. 12–28, 2024.
- [25] A. Manzoor, M. Ozger, D. Schupke, and C. Cavdar, “Combined airspace and non-terrestrial 6g networks for advanced air mobility”, in *2024 20th International Conference on the Design of Reliable Communication Networks (DRCN)*, 2024, pp. 47–53.
- [26] I. A. Meer, “Ai-assisted mobility management for cellular connected uavs”, Ph.D. dissertation, KTH Royal Institute of Technology, 2025.

- [27] S. Hofmann, V. Megas, M. Ozger, D. Schupke, F. H. P. Fitzek, and C. Cavdar, “Combined optimal topology formation and rate allocation for aircraft-to-aircraft communications”, in *ICC 2019 - IEEE International Conference on Communications (ICC)*, 2019, pp. 1–6.
- [28] G. K. Pandey, D. S. Gurjar, H. H. Nguyen, and S. Yadav, “Security threats and mitigation techniques in uav communications: a comprehensive survey”, *IEEE Access*, vol. 10, pp. 112 858–112 897, 2022.
- [29] U. Madhow, *Introduction to communication systems*. Cambridge University Press, 2014.
- [30] GNU Radio Project, *IQ Complex Tutorial*, https://wiki.gnuradio.org/index.php/IQ_Complex_Tutorial, Online; accessed 2025-08-03, 2025.
- [31] N. Yu, J. Wu, C. Zhou, Z. Shi, and J. Chen, “Open set learning for rf-based drone recognition via signal semantics”, *IEEE Transactions on Information Forensics and Security*, 2024.
- [32] D. Shorten, A. Williamson, S. Srivastava, and J. C. Murray, “Localisation of drone controllers from rf signals using a deep learning approach”, in *Proceedings of the international conference on pattern recognition and artificial intelligence*, 2018, pp. 89–97.
- [33] M. F. Al-Sa’d, A. Al-Ali, A. Mohamed, T. Khattab, and A. Erbad, “Rf-based drone detection and identification using deep learning approaches: an initiative towards a large open source drone database”, *Future Generation Computer Systems*, vol. 100, pp. 86–97, 2019.
- [34] C. Xu, B. Chen, Y. Liu, F. He, and H. Song, “Rf fingerprint measurement for detecting multiple amateur drones based on stft and feature reduction”, in *2020 Integrated Communications Navigation and Surveillance Conference (ICNS)*, IEEE, 2020, 4G1–1.
- [35] R. Kılıç, N. Kumbasar, E. A. Oral, and I. Y. Ozbek, “Drone classification using rf signal based spectral features”, *Engineering Science and Technology, an International Journal*, vol. 28, p. 101 028, 2022.
- [36] O. O. Medaiyese, M. Ezuma, A. P. Lauf, and I. Guvenc, “Wavelet transform analytics for rf-based uav detection and identification system using machine learning”, *Pervasive and Mobile Computing*, vol. 82, p. 101 569, 2022.

- [37] C. Xu, F. He, B. Chen, Y. Jiang, and H. Song, “Adaptive rf fingerprint decomposition in micro uav detection based on machine learning”, in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2021, pp. 7968–7972.
- [38] A. Shoufan, H. M. Al-Angari, M. F. A. Sheikh, and E. Damiani, “Drone pilot identification by classifying radio-control signals”, *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 10, pp. 2439–2447, 2018.
- [39] Q. Wang, L. Wang, L. Yu, J. Wang, and X. Zhang, “An id-based robust identification approach toward multitype noncooperative drones”, *IEEE Sensors Journal*, vol. 23, no. 9, pp. 10 179–10 192, 2023.
- [40] S. Rajendran, W. Meert, V. Lenders, and S. Pollin, “Unsupervised wireless spectrum anomaly detection with interpretable features”, *IEEE transactions on cognitive communications and networking*, vol. 5, no. 3, pp. 637–647, 2019.
- [41] A. Alipour-Fanid, M. Dabaghchian, N. Wang, P. Wang, L. Zhao, and K. Zeng, “Machine learning-based delay-aware uav detection and operation mode identification over encrypted wi-fi traffic”, *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 2346–2360, 2019.
- [42] C. Geng, S.-j. Huang, and S. Chen, “Recent advances in open set recognition: a survey”, *IEEE transactions on pattern analysis and machine intelligence*, vol. 43, no. 10, pp. 3614–3631, 2020.
- [43] W.-Y. Yan and Q. He, “Multi-class fuzzy support vector machine based on dismissing margin”, in *2009 International Conference on Machine Learning and Cybernetics*, vol. 2, 2009, pp. 1139–1144. doi: [10.1109/ICMLC.2009.5212368](https://doi.org/10.1109/ICMLC.2009.5212368).
- [44] Y. Wen, K. Zhang, Z. Li, and Y. Qiao, “A discriminative feature learning approach for deep face recognition”, in *European conference on computer vision*, Springer, 2016, pp. 499–515.
- [45] J. Muñoz-Marí, F. Bovolo, L. Gómez-Chova, L. Bruzzone, and G. Camp-Valls, “Semisupervised one-class support vector machines for classification of remote sensing data”, *IEEE Transactions on Geoscience and Remote Sensing*, vol. 48, no. 8, pp. 3188–3197, 2010. doi: [10.1109/TGRS.2010.2045764](https://doi.org/10.1109/TGRS.2010.2045764).

- [46] S. Zhu, G. Shuai, G. Sun, and J. Zhang, “Change detection for mapping paddy using support vector data description”, in *2012 First International Conference on Agro- Geoinformatics (Agro-Geoinformatics)*, 2012, pp. 1–4. doi: [10.1109/Agro-Geoinformatics.2012.6311649](https://doi.org/10.1109/Agro-Geoinformatics.2012.6311649).
- [47] A. A. Wani, “Comprehensive analysis of clustering algorithms: exploring limitations and innovative solutions”, *PeerJ Computer Science*, vol. 10, e2286, 2024.
- [48] D. Deng, “DbSCAN clustering algorithm based on density”, in *2020 7th international forum on electrical engineering and automation (IFEEA)*, IEEE, 2020, pp. 949–953.
- [49] A. Rykov, R. C. De Amorim, V. Makarenkov, and B. Mirkin, “Inertia-based indices to determine the number of clusters in k-means: an experimental evaluation”, *Ieee Access*, vol. 12, pp. 11 761–11 773, 2024.
- [50] J. Kuha, “Aic and bic: comparisons of assumptions and performance”, *Sociological methods & research*, vol. 33, no. 2, pp. 188–229, 2004.
- [51] O. Arbelaitz, I. Gurrutxaga, J. Muguerza, J. M. Pérez, and I. Perona, “An extensive comparative study of cluster validity indices”, *Pattern Recognition*, vol. 46, no. 1, pp. 243–256, 2013, ISSN: 0031-3203. doi: <https://doi.org/10.1016/j.patcog.2012.07.021>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S003132031200338X>.
- [52] G. M. Van de Ven, T. Tuytelaars, and A. S. Tolias, “Three types of incremental learning”, *Nature Machine Intelligence*, vol. 4, no. 12, pp. 1185–1197, 2022.
- [53] Y. Wu *et al.*, “Large scale incremental learning”, in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 374–382.
- [54] G. Petit, A. Popescu, H. Schindler, D. Picard, and B. Delezoide, “Fetрил: feature translation for exemplar-free class-incremental learning”, in *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, 2023, pp. 3911–3920.
- [55] L. Yu *et al.*, “Semantic drift compensation for class-incremental learning”, in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 6982–6991.

- [56] F. Svanström, C. Englund, and F. Alonso-Fernandez, “Real-time drone detection and tracking with visible, thermal and acoustic sensors”, in *2020 25th International Conference on Pattern Recognition (ICPR)*, IEEE, 2021, pp. 7265–7272.
- [57] A. Bernardini, F. Mangiatordi, E. Pallotti, and L. Capodiferro, “Drone detection by acoustic signature identification”, in *Proc. IS&T Int. Symp. Electron. Imag. Sci. Technol.*, Jan. 2017, pp. 60–64.
- [58] E. Matson, B. Yang, A. Smith, E. Dietz, and J. Gallagher, “UAV detection system with multiple acoustic nodes using machine learning models”, in *Proc. 3rd IEEE Int. Conf. Robot. Comput. (IRC)*, Institute of Electrical and Electronics Engineers, Piscataway, NJ, USA, Mar. 2019, pp. 493–498.
- [59] S. Jeon, J.-W. Shin, Y.-J. Lee, W.-H. Kim, Y. Kwon, and H.-Y. Yang, “Empirical study of drone sound detection in real-life environment with deep neural networks”, in *Proc. 25th Eur. Signal Process. Conf. (EUSIPCO)*, Institute of Electrical and Electronics Engineers, Piscataway, NJ, USA, Aug. 2017, pp. 1858–1862.
- [60] X. Yue, Y. Liu, J. Wang, H. Song, and H. Cao, “Software defined radio and wireless acoustic networking for amateur drone surveillance”, *IEEE Communications Magazine*, vol. 56, no. 4, pp. 90–97, Apr. 2018.
- [61] Z. Shi, X. Chang, C. Yang, Z. Wu, and J. Wu, “An acoustic-based surveillance system for amateur drones detection and localization”, *IEEE Transactions on Vehicular Technology*, vol. 69, no. 3, pp. 2731–2739, Mar. 2020.
- [62] M. F. Al-Sa’id, A. Al-Ali, A. Mohamed, T. Khattab, and A. Erbad, “Rf-based drone detection and identification using deep learning approaches: an initiative towards a large open source drone database”, *Future Generation Computer Systems*, vol. 100, pp. 86–97, 2019.
- [63] Y. Xie, P. Jiang, Y. Gu, and X. Xiao, “Dual-source detection and identification system based on uav radio frequency signal”, *IEEE Transactions on Instrumentation and Measurement*, vol. 70, pp. 1–15, 2021.
- [64] P. Andrašić, T. Radišić, M. Muštra, and J. Ivošević, “Night-time detection of uavs using thermal infrared camera”, *Transportation research procedia*, vol. 28, pp. 183–190, 2017.

- [65] I. Guvenc, F. Koohifar, S. Singh, M. L. Sichitiu, and D. Matolak, “Detection, tracking, and interdiction for amateur drones”, *IEEE communications magazine*, vol. 56, no. 4, pp. 75–81, 2018.
- [66] K. N. Inani, K. Sangwan, *et al.*, “Machine learning based framework for drone detection and identification using rf signals”, in *2023 4th International Conference on Innovative Trends in Information Technology (ICITIIT)*, IEEE, 2023, pp. 1–8.
- [67] C. Xu, B. Chen, Y. Liu, F. He, and H. Song, “Rf fingerprint measurement for detecting multiple amateur drones based on stft and feature reduction”, in *Proc. Integr. Commun. Navigat. Surveill. Conf. (ICNS)*, 2020, 4G1.
- [68] R. Kılıç, N. Kumbasar, E. A. Oral, and I. Y. Ozbek, “Drone classification using rf signal based spectral features”, *Engineering Science and Technology, an International Journal*, vol. 28, p. 101 028, 2022.
- [69] O. O. Medaiyese, M. Ezuma, A. P. Lauf, and I. Guvenc, “Wavelet transform analytics for rf-based uav detection and identification system using machine learning”, *Pervasive and Mobile Computing*, vol. 82, p. 101 569, 2022.
- [70] C. Xu, F. He, B. Chen, Y. Jiang, and H. Song, “Adaptive rf fingerprint decomposition in micro uav detection based on machine learning”, in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2021, pp. 7968–7972.
- [71] A. Shoufan, H. M. Al-Angari, M. F. A. Sheikh, and E. Damiani, “Drone pilot identification by classifying radio-control signals”, *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 10, pp. 2439–2447, 2018.
- [72] Q. Wang, L. Wang, L. Yu, J. Wang, and X. Zhang, “An id-based robust identification approach toward multitype noncooperative drones”, *IEEE Sensors Journal*, vol. 23, no. 9, pp. 10 179–10 192, 2023.
- [73] S. Rajendran, W. Meert, V. Lenders, and S. Pollin, “Unsupervised wireless spectrum anomaly detection with interpretable features”, *IEEE Transactions on Cognitive Communications and Networking*, vol. 5, no. 3, pp. 637–647, 2019.

- [74] Z. Cai, Y. Wang, Q. Jian, G. Gui, and J. Sha, "Toward intelligent lightweight and efficient uav identification with rf fingerprinting", *IEEE Internet of Things Journal*, 2024. DOI: [10.1109/JIOT.2024.3395466](https://doi.org/10.1109/JIOT.2024.3395466).
- [75] A. A. Fanid, M. Dabaghchian, N. Wang, P. Wang, L. Zhao, and K. Zeng, "Machine learning-based delay-aware uav detection and operation mode identification over encrypted wi-fi traffic", *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 2346–2360, 2020.
- [76] L. Lin, N. Yu, Y. Wang, and Z. Shi, "5g spectrum learning-based passive uav detection in urban scenario", in *Proc. IEEE/CIC International Conference on Communications in China (ICCC)*, 2023, pp. 1–5.
- [77] N. Soltani, G. Reus-Muns, B. Salehi, J. Dy, S. Ioannidis, and K. M. Chowdhury, "RF fingerprinting unmanned aerial vehicles with non-standard transmitter waveforms", *IEEE Transactions on Vehicular Technology*, vol. 69, no. 12, pp. 15 518–15 531, Dec. 2020. DOI: [10.1109/TVT.2020.3037865](https://doi.org/10.1109/TVT.2020.3037865).
- [78] A. Bendale and T. E. Boult, "Towards open set deep networks", in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2016, pp. 1563–1572. DOI: [10.1109/CVPR.2016.173](https://doi.org/10.1109/CVPR.2016.173).
- [79] T. Li, Q. Zhou, X. Wang, J. Li, H. Chen, and W. Sun, "The importance of expert knowledge for automatic modulation open set recognition", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 11, pp. 13 730–13 748, Nov. 2023. DOI: [10.1109/TPAMI.2023.3256784](https://doi.org/10.1109/TPAMI.2023.3256784).
- [80] Y. Dong, X. Jiang, H. Zhou, Y. Lin, and Q. Shi, "Sr2cnn: zero-shot learning for signal recognition", *IEEE Transactions on Signal Processing*, vol. 69, pp. 2316–2329, 2021. DOI: [10.1109/TSP.2021.3075689](https://doi.org/10.1109/TSP.2021.3075689).
- [81] H. Han, W. Li, Z. Feng, G. Fang, Y. Xu, and Y. Xu, "Proceed from known to unknown: jamming pattern recognition under open-set setting", *IEEE Wireless Communications Letters*, vol. 11, no. 4, pp. 693–697, Apr. 2022. DOI: [10.1109/LWC.2022.3165678](https://doi.org/10.1109/LWC.2022.3165678).

- [82] C. J. Swinney and J. C. Woods, “K-means clustering approach to uas classification via graphical signal representation of radio frequency signals for air traffic early warning”, *IEEE transactions on intelligent transportation systems*, vol. 23, no. 12, pp. 24 957–24 965, 2022.
- [83] T. M. Hoang, N. M. Nguyen, and T. Q. Duong, “Detection of eavesdropping attack in uav-aided wireless systems: unsupervised learning with one-class svm and k-means clustering”, *IEEE Wireless Communications Letters*, vol. 9, no. 2, pp. 139–142, 2019.
- [84] T.-N. Tran, T.-L. Nguyen, V. T. Hoang, and M. Voznak, “Sensor clustering using a k-means algorithm in combination with optimized unmanned aerial vehicle trajectory in wireless sensor networks”, *Sensors*, vol. 23, no. 4, p. 2345, 2023.
- [85] J. C. Park, K.-M. Kang, and J. Choi, “K-means clustering-aided power control for uav-enabled ofdm networks”, *IEEE Access*, vol. 12, pp. 15 549–15 560, 2024.
- [86] E. Patel and D. S. Kushwaha, “Clustering cloud workloads: k-means vs gaussian mixture model”, *Procedia computer science*, vol. 171, pp. 158–167, 2020.
- [87] X. He, D. Cai, Y. Shao, H. Bao, and J. Han, “Laplacian regularized gaussian mixture model for data clustering”, *IEEE transactions on knowledge and data engineering*, vol. 23, no. 9, pp. 1406–1418, 2010.
- [88] Y. Zhang *et al.*, “Gaussian mixture model clustering with incomplete data”, *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, vol. 17, no. 1s, pp. 1–14, 2021.
- [89] J. Shen, X. Hao, Z. Liang, Y. Liu, W. Wang, and L. Shao, “Real-time superpixel segmentation by dbscan clustering algorithm”, *IEEE transactions on image processing*, vol. 25, no. 12, pp. 5933–5942, 2016.
- [90] M. Hahsler, M. Piekenbrock, and D. Doran, “Dbscan: fast density-based clustering with r”, *Journal of Statistical Software*, vol. 91, pp. 1–30, 2019.
- [91] K. Khan, S. U. Rehman, K. Aziz, S. Fong, and S. Sarasvady, “Dbscan: past, present and future”, in *The fifth international conference on the applications of digital information and web technologies (ICADIWT 2014)*, IEEE, 2014, pp. 232–238.

- [92] C. K. Reddy and B. Vinzamuri, “A survey of partitional and hierarchical clustering algorithms”, in *Data clustering*, Chapman and Hall/CRC, 2018, pp. 87–110.
- [93] X. Ran, Y. Xi, Y. Lu, X. Wang, and Z. Lu, “Comprehensive survey on hierarchical clustering algorithms and the recent developments”, *Artificial Intelligence Review*, vol. 56, no. 8, pp. 8219–8264, 2023.
- [94] F. Nielsen, “Hierarchical clustering”, in *Introduction to HPC with MPI for Data Science*, Springer, 2016, pp. 195–211.
- [95] F. Murtagh and P. Contreras, “Algorithms for hierarchical clustering: an overview, ii”, *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 7, no. 6, e1219, 2017.
- [96] F. Murtagh and P. Contreras, “Algorithms for hierarchical clustering: an overview”, *Wiley interdisciplinary reviews: data mining and knowledge discovery*, vol. 2, no. 1, pp. 86–97, 2012.
- [97] Y. Liu, J. Wang, J. Li, S. Niu, and H. Song, “Class-incremental learning for wireless device identification in iot”, *IEEE Internet of Things Journal*, vol. 8, no. 23, pp. 17 227–17 235, 2021.
- [98] T. Li *et al.*, “Meta-rff: meta-task adaptive based few-shot open-set incremental learning for rf fingerprint recognition”, *IEEE Transactions on Cognitive Communications and Networking*, 2025.
- [99] Y. A. Farrukh and I. Khan, “An autonomous self-incremental learning approach for detection of cyber attacks on unmanned aerial vehicles (uavs)”, *arXiv preprint arXiv:2112.11219*, 2021.
- [100] X. He *et al.*, “Federated continuous learning based on stacked broad learning system assisted by digital twin networks: an incremental learning approach for intrusion detection in uav networks”, *IEEE Internet of Things Journal*, vol. 10, no. 22, pp. 19 825–19 838, 2023.
- [101] P. Cudrano, X. Luo, and M. Matteucci, “The empirical impact of forgetting and transfer in continual visual odometry”, *arXiv preprint arXiv:2406.01797*, 2024.
- [102] J. Deng, X. Ji, B. Wang, B. Wang, and W. Xu, “Dr. defender: proactive detection of autopilot drones based on csi”, *IEEE Transactions on Information Forensics and Security*, vol. 19, pp. 194–206, 2024.

- [103] M. Wang, X. Li, R. Chen, H. Zhao, and Y. Zhang, “Uncertainty-inspired open set learning for retinal anomaly identification”, *Nature Communications*, vol. 14, no. 1, p. 6757, Oct. 2023. DOI: [10.1038/s41467-023-41778-3](https://doi.org/10.1038/s41467-023-41778-3).
- [104] S. Chikkerur, A. N. Cartwright, and V. Govindaraju, “Fingerprint enhancement using stft analysis”, *Pattern recognition*, vol. 40, no. 1, pp. 198–211, 2007.
- [105] M. K. Kıymık, İ. Güler, A. Dizibüyük, and M. Akın, “Comparison of stft and wavelet transform methods in determining epileptic seizure activity in eeg signals for real-time application”, *Computers in biology and medicine*, vol. 35, no. 7, pp. 603–616, 2005.
- [106] T. Chen *et al.*, “Xgboost: extreme gradient boosting”, *R package version 0.4-2*, vol. 1, no. 4, pp. 1–4, 2015.
- [107] D. Nielsen, “Tree boosting with xgboost-why does xgboost win" every" machine learning competition?”, M.S. thesis, NTNU, 2016.
- [108] Y. Qiu, J. Zhou, M. Khandelwal, H. Yang, P. Yang, and C. Li, “Performance evaluation of hybrid woa-xgboost, gwo-xgboost and bo-xgboost models to predict blast-induced ground vibration”, *Engineering with Computers*, vol. 38, no. Suppl 5, pp. 4145–4162, 2022.
- [109] T. Chen and C. Guestrin, “Xgboost: a scalable tree boosting system”, in *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, 2016, pp. 785–794.
- [110] A. Ogunleye and Q.-G. Wang, “Xgboost model for chronic kidney disease diagnosis”, *IEEE/ACM transactions on computational biology and bioinformatics*, vol. 17, no. 6, pp. 2131–2140, 2019.
- [111] X. Lei, H. Pan, and X. Huang, “A dilated cnn model for image classification”, *IEEE access*, vol. 7, pp. 124 087–124 095, 2019.
- [112] J. He, P. Wu, Y. Tong, X. Zhang, M. Lei, and J. Gao, “Bearing fault diagnosis via improved one-dimensional multi-scale dilated cnn”, *Sensors*, vol. 21, no. 21, p. 7319, 2021.
- [113] D.-T. Dinh, T. Fujinami, and V.-N. Huynh, “Estimating the optimal number of clusters in categorical data clustering by silhouette coefficient”, in *International Symposium on Knowledge and Systems Sciences*, Springer, 2019, pp. 1–17.

- [114] X. Wang and Y. Xu, “An improved index for clustering validation based on silhouette index and calinski-harabasz index”, in *IOP Conference Series: Materials Science and Engineering*, IOP Publishing, vol. 569, 2019, p. 052 024.
- [115] S. Petrovic, “A comparison between the silhouette index and the davies-bouldin index in labelling ids clusters”, in *Proceedings of the 11th Nordic workshop of secure IT systems*, Citeseer, vol. 2006, 2006, pp. 53–64.
- [116] S. Raschka, “Model evaluation, model selection, and algorithm selection in machine learning”, *arXiv preprint arXiv:1811.12808*, 2018.
- [117] Y. Yao, Z. Xiao, B. Wang, B. Viswanath, H. Zheng, and B. Y. Zhao, “Complexity vs. performance: empirical analysis of machine learning as a service”, in *Proceedings of the 2017 Internet Measurement Conference*, 2017, pp. 384–397.

TRITA-EECS-EX-2025:943
Stockholm, Sweden 2025

www.kth.se