This article has been accepted for publication in IEEE Transactions on Audio, Speech and Language Processing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TASLPRO.2025.3646824

Frequency Tracking Features for Domain-Robust and Data-Efficient Deep Siren Identification

Stefano Damiano, Thomas Dietzen, Toon van Waterschoot

***Abstract*—Siren identification systems are crucial for smart vehicles to early detect emergency situations that demand attention and, often, a specific response from the driver. The problem of recognizing the sound of a siren within traffic noise has been largely addressed using deep learning, to tackle the strong and unstructured background noise characterizing the urban soundscape. While convolutional neural networks processing spectrogram features are state-of-the-art, their performance is tied to the availability of large and diverse datasets and is hindered by domain-shift. With the aim of obtaining domain-robust models that do not demand extensive training datasets, that are hardly available in practice, in this work we propose a low-complexity feature extraction method based on frequency tracking. Relying on the harmonic nature of siren signals, we adopt a single-parameter adaptive notch filter to track their periodically evolving fundamental frequency and the ratio between the power of the tracked sinusoid and that of the full input signal. These features are used as input for a small-scale convolutional network, trained entirely on synthetic data. Evaluation results indicate that the proposed system achieves a good generalization to multiple real-world datasets unseen during training, outperforming spectrogram-based state-of-the-art methods when limited data is available while also having a smaller model size.**

***Index Terms*—siren identification, frequency tracking, domain generalization, synthetic training, convolutional neural network**

I. Introduction

THE deployment of smart vehicles with a steadily growing level of automation is backed by extensive research on systems that enhance the safety of passengers by providing assistance to the driver. In particular, advanced driver-assistance systems (ADASs) include a series of devices that support the driver in both normal driving and emergency situations by detecting important events occurring in the vicinity. Examples of such devices are adaptive cruise control, obstacle detection and automatic emergency breaking functionalities, lane changing and vehicle presence warning systems [1]. These ADASs rely on a set of sensors that monitor the surroundings of the vehicle. While vision is the most utilized sensing modality, modern cars are equipped with several additional sensors including radar, LiDAR, infrared and ultrasonic sensors, providing advanced perceptual abilities that enhance the capabilities and the robustness of detection systems [1], [2].

S. Damiano and T. van Waterschoot are with the Department of Electrical Engineering (ESAT), STADIUS Center for Dynamical Systems, Signal Processing and Data Analytics, KU Leuven, Leuven 3001, Belgium (e-mail: stefano.damiano@esat.kuleuven.be, toon.vanwaterschoot@esat.kuleuven.be).

Thomas Dietzen is with the Department of Electrical Engineering (ESAT), Processing Speech and Images, EAVISE, KU Leuven, Sint-Katelijne-Waver 2860, Belgium (e-mail: thomas.dietzen@esat.kuleuven.be).

When driving, visual data constitutes the main source of information not only for autonomous systems, but also for humans. However, the sounds that can be heard in the environment also carry vital information. Most emergency situations are in fact announced by a sound to attract attention: in particular, emergency vehicles (EV) use the siren to signal their presence and ask cars to leave way, while car horns are used to inform other drivers of some impending danger. These sounds have the advantage of being audible from a large distance, in low-visibility conditions and in presence of occlusions, that prevent drivers from seeing the incoming EV or visual sensors from detecting it. Moreover, sound can provide information on the traffic conditions (e.g., presence, density and speed of vehicles, bicycles and pedestrians), the weather and road status, and indicate the failure or malfunctioning of some car component.

For this reason, research on how to integrate acoustic sensing systems into smart vehicles has accelerated in recent years [3], [4]. While audio information retrieval has been studied for several tasks including traffic monitoring [5], [6], fault detection [7] and road surface status analysis [8], its primary application lies in the identification of EVs from the sound of their siren and, in certain multi-microphone setups, their localization [3], [9].

A first class of siren identification methods, based on traditional digital signal processing, exploits specific predictable characteristics of the siren sound to recognize it within traffic noise. For instance, the periodic alternation of two tones characterizing the *two-tone* siren is exploited in [10], via pitch tracking, and in [11], via the longest common subsequence, whereas the periodic modulation of the fundamental frequency characterizing a *yelp* siren is exploited in [12]. These methods feature a low complexity but target specific siren types, offering a limited generalization.

To overcome this limitation, a second class of techniques based on machine and deep learning has been proposed. Some approaches are based on manually extracting features from the audio signals and solving a classification task to identify the presence of sirens, using support vector machines (SVMs) [13], [14], Gaussian mixture model (GMM) classifiers [15], k-nearest neighbor classifiers (k-NN) [16] or part-based models [17]. Most techniques rely instead on end-to-end deep neural networks to solve the identification task. In [18] and [19] a feed-forward network is adopted, taking as input features mel-frequency cepstral coefficients (MFCCs) and the raw audio waveform, respectively. The vast majority of deep learning models, instead, employs convolutional neural

0000–0000/00$00.00 © 2021 IEEE

Authorized licensed use limited to: KU Leuven University Library. Downloaded on January 05,2026 at 10:23:58 UTC from IEEE Xplore. Restrictions apply.
© 2025 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Audio, Speech and Language Processing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TASLPRO.2025.3646824

networks (CNNs), mostly relying on spectral features: [6], [20]–[25] use spectrogram features as input for 2D CNNs, whereas [26] adopts MFCC, gammatone-frequency cepstral coefficients (GFCCs), the constant-Q transform (CQT) and the chromagram features combined. In [27], on the other hand, a processed version of the audio waveform is used as input for a 1D CNN. Ensemble models based on convolutional architectures have also been explored to enhance performance and robustness [28], [29]: these models employ spectral features (mel spectrogram, MFCC) and the raw audio waveform as input for the identification network. Fewer works propose other network types, including long short-term memory (LSTM) combined with CNN blocks [30], prototypical networks [31] and spiking neural networks [32]. While all these models target the siren identification task, a few works employing multi-microphone setups propose systems to identify also its direction of arrival, relying on spectral or correlation features [3], [16], [19], [33], [34]. Finally, multi-modal EV identification networks, involving visual and acoustic data, have been investigated in recent years, exploiting the advances in computer vision and the integration of multiple sensing modalities to achieve an improved robustness [35]–[38].

Deep learning siren identification systems, while robust to the complex and nonstationary acoustic scene characterizing urban environments, face several challenges. First, they usually entail a large computational complexity, which hinders their deployment on resource-constrained edge devices for on-vehicle operation. Second, they demand vast training datasets that capture the diversity of the complex target soundscape: on the one hand, the characteristics of siren sounds strongly vary among different regions and, on the other, the background noise also varies depending on the landscape (urban or rural), the region, the time of the day, etc. In practice, the available data is unlikely to capture this complexity and, therefore, models have limited generalization capabilities. This aspect has been analyzed in [39], where it has been shown that CNN models trained on one dataset and achieving state-of-the-art performance on it struggle to generalize to unseen domains or, in other words, lack domain-robustness. Strategies to minimize the amount of in-domain data needed for training have been investigated. In [31], few-shot learning is adopted to enhance the ability of a pre-trained model to recognize a specific siren, unseen during pre-training. On the other hand, in [39] a training strategy relying entirely on synthetic data is proposed, aiming at enhancing the data diversity, and BCResNet architectures [40], specifically targeting domain generalization, are employed to guarantee good generalization to real-world data.

In this work, we propose a low-complexity, domain-robust siren identification system, relying on frequency tracking features and synthetic training data. Starting from the observation that, as opposed to the unstructured traffic noise characterizing the urban soundscape, siren signals have a well-defined harmonic behavior, characterized by a periodically evolving fundamental frequency, we propose a novel set of features to exploit this signal structure. Using the Kalman filtering-based single-parameter adaptive notch filter (ANF) introduced in [41] (KalmANF), we track the fundamental frequency and compute the ratio between the power of the tracked sinusoid, obtained at the output of the ANF, and that of the input audio. These two compact 1D features are then used as input to a low-complexity CNN. We first show that the proposed method, thanks to the limited model size and to the compactness of the features, can be trained with a limited amount of real-world data (i.e., in a data-efficient setting) and is robust to domain shift. Since in audio signals the domain information is mostly captured along the frequency dimension of the time-frequency spectrum [40], the proposed features, leveraging the instantaneous tracking of a *single* frequency, are less prone to domain overfitting compared to the full spectrogram and thus guarantee an enhanced domain-robustness. Motivated by this, we finally adopt an entirely synthetic dataset for training, and show that the proposed model achieves good generalization on two different real-world datasets, even in absence of real-world training data. Preliminary results of the current work have been presented in our conference paper [42]. In this manuscript, we extend the discussion of [42] with the following contributions:

1) We propose a training strategy that leverages entirely synthetic data, showing that the proposed frequency tracking features guarantee good generalization to real-world data.
2) We largely extend the experimental validation introduced in [42]: we compare the proposed method with several different spectrogram-based CNNs, including BCResNet models, designed for domain generalization, and the CP-Mobile architecture. We show that the proposed method outperforms the baselines both when training data is limited, and when synthetic data is used for training.
3) We discuss the effect of fine-tuning on both the proposed CNN and the traditional spectrogram-based ones, showing that, while traditional models strongly benefit from fine-tuning using real-world data, the proposed method often achieves optimal generalization without necessitating any fine-tuning.
4) We analyze the computational complexity of the proposed method and compare it with two spectrogram-based baselines, showing that the proposed features lead a significantly reduced number of operations in both feature extraction and model inference.

The rest of the paper is organized as follows. In Sec. II we review the architectures of spectrogram-based networks used as baselines to evaluate the proposed method, including a VGG-inspired network [21], BCResNet architectures [39] and CPMobile [43]. In Sec. III we introduce the proposed frequency tracking features, the proposed 1D CNN architectures, and the synthetic data used for training. In Sec. IV we assess the performance of the proposed method by comparing it with the baselines when using a data-efficient training procedure, fully synthetic training, and synthetic pre-training combined with fine-tuning on real-world data. We provide an interpretation of the effect of the proposed features on the representation learned by the network in Sec. V, evaluate the computational complexity of the proposed and baseline models in Sec. VI, and finally draw conclusions in Sec. VII. Accompanying code is provided at [44].

Authorized licensed use limited to: KU Leuven University Library. Downloaded on January 05,2026 at 10:23:58 UTC from IEEE Xplore. Restrictions apply.
© 2025 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Audio, Speech and Language Processing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TASLPRO.2025.3646824

II. Problem Definition and Baseline Systems

The siren identification problem is formulated as a binary classification task, where the classifier receives as input an audio segment with a fixed length (in this work, we use 2 s input segments) and outputs a unique label, either *siren* or *noise*. We use neural networks as classifiers: as our work aims for low-complexity and domain-robustness, in this section we introduce four tiny state-of-the-art architectures that will be used as baselines for the assessment of the proposed method. A typical deep siren identification system is based on three steps. First, the feature extractor takes as input the audio waveform and produces 2D spectral features, typically the mel spectrogram. This spectrogram is then processed by a neural network composed of two cascaded parts: first, a 2D CNN produces embedded features, then a multi-layer perceptron (MLP) composed of a few fully-connected (FC) layers produces the probability that a siren is present in the input audio. If the probability is above a fixed threshold (normally 0.5, also used in this work) the input is assigned to the siren class. We use as baselines the three siren identification networks adopted in [39], and the CPMobile architecture [43], originally proposed for low-complexity acoustic scene classification. In particular, we modify the CPMobile implementation [45] adopted as baseline system for the 2025 DCASE Challenge Task 1 [46] to employ it for our (binary) siren identification problem.

The first baseline is a VGG-inspired [47] CNN presented in [39] as a slightly modified version of the architecture proposed in [21]. The network, called VGGSiren in the following, is made of a CNN feature extractor of three blocks, each containing two 2D convolutional layers (Conv2D), a batch normalization operation (BN) and a max pooling (MaxPool) performing downsampling by a factor 2 along both the temporal and frequency dimensions. After a global average pooling operation (GlobAvgPool), a classifier head of 2 FC layers is present.

The BCResNet and BCResNorm networks were originally proposed in [40] for acoustic scene classification and successfully adopted in [39] for siren identification with enhanced domain robustness. These networks are based on the same architecture, relying on the BC-ResBlock: in this element, Conv2D layers process the mel spectrogram and Conv1D blocks process frequency-averaged embedded features; the 1D and 2D features are then combined using broadcasting operations and residual connections [40]. The BCResNorm introduces residual normalization operations after each BC-ResBlock to enhance domain generalization [40]; for further details we refer to [39].

Finally, CPMobile was also originally proposed for low-complexity acoustic scene classification. This model relies on three convolutional blocks: a standard block (S), consisting of two normal Conv2D layers interleaved by a third Conv2D layer performing depthwise convolution; a transition block (T), featuring the same structure with an added residual connection; a spatial downsampling block (D), with similar structure, but including depthwise convolutions and average pooling to reduce the spatial dimension [46]. The four architectures are illustrated in Tab. I, Tab. II and Tab. III. In total, VGGSiren has 72.9 k parameters, BCResNet has 45.7 k, BCResNorm has 46.2 k parameters and CPMobile has 60.2 k parameters. A detailed analysis of the complexity of all models is carried out in Sec. VI.

The four baseline networks exploit the mel spectrogram as input feature. While this time-frequency representation is highly effective for several sound analysis and recognition tasks, it constitutes a generic audio descriptor that does not exploit application-specific knowledge. Motivated by the structured nature of siren signals as opposed to background noise, in the next section we propose a novel set of frequency tracking features to solve the identification task. These features have

TABLE I
THE VGGSIREN ARCHITECTURE [39]. ACRONYMS ARE DEFINED IN SEC. II.

Layer	Num. Blocks	Kernel Size	Output channels
Input	-	-	1
Conv2D + BN	2	$(3, 3)$	16
MaxPool	-	$(2, 2)$	-
Conv2D + BN	2	$(3, 3)$	32
MaxPool	-	$(2, 2)$	-
Conv2D + BN	2	$(3, 3)$	64
GlobAvgPool	-	-	-
FC + BN	-	-	10
Output	-	-	1

TABLE II
THE BCRESNET ARCHITECTURE [39]. BCRESNORM HAS THE SAME STRUCTURE, WITH THE INTRODUCTION OF RESIDUAL NORMALIZATION OPERATIONS AFTER EACH BC-RESBLOCK. ACRONYMS ARE DEFINED IN SEC. II.

Layer	Num. Blocks	Kernel Size	Output channels
Input	-	-	1
Conv2D	1	$(5, 5)$	60
BC-ResBlock	2	-	30
MaxPool	-	$(2, 2)$	-
BC-ResBlock	2	-	45
MaxPool	-	$(2, 2)$	-
BC-ResBlock	2	-	60
BC-ResBlock	3	-	75
Conv2D	1	$(1, 1)$	1
GlobAvgPool	-	-	-

TABLE III
THE CPMOBILE ARCHITECTURE [43]. ACRONYMS ARE DEFINED IN SEC. II.

Layer	Kernel Size	Stride	Output channels
Input	-	-	1
Conv2D + BN	$(3, 3)$	$(2, 2)$	8
Conv2D + BN	$(3, 3)$	$(2, 2)$	32
Block S	-	$(1, 1)$	32
Block D	-	$(2, 2)$	32
Block S	-	$(1, 1)$	32
Block T	-	$(2, 1)$	56
Block S	-	$(1, 1)$	56
Block T	-	$(1, 1)$	104
Conv2D + BN	$(1, 1)$	$(1, 1)$	1
GlobAvgPool	-	-	
Output	-	-	1

Authorized licensed use limited to: KU Leuven University Library. Downloaded on January 05,2026 at 10:23:58 UTC from IEEE Xplore. Restrictions apply.
© 2025 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Audio, Speech and Language Processing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TASLPRO.2025.3646824

a significantly reduced size compared to the mel spectrogram: this, on the one hand, promotes the reduction of the model size and, hence, of the computational complexity, and, on the other hand, allows networks to be trained on smaller datasets. Moreover, the proposed features are more robust than the spectrogram to domain shift: as domain information is mostly contained on the frequency dimension [40], using a single tracked frequency as feature results in models less prone to learn domain-specific time-frequency patterns. A major advantage of the proposed method is that it can also leverage entirely synthetic data for training while achieving good generalization performance on multiple real-world datasets.

III. Proposed Method

In this section, we introduce the proposed frequency tracking features and the neural networks adopted to solve the classification problem. We finally illustrate the synthetic dataset adopted for training the proposed model.

A. Frequency Tracking Features

The proposed method relies on the Kalman-based KalmANF frequency tracking algorithm introduced in [41], that we adopt to compute the proposed features. The KalmANF implements a single-parameter ANF, a specific type of notch filter [48] designed to track and suppress a high-energy sinusoidal component whose frequency varies over time while leaving other frequencies relatively unaffected. In order to track the varying frequency, the ANF notch is recursively updated. The KalmANF is designed as a bi-quadratic infinite impulse response (IIR) filter, constrained to have poles and zeros lying on the same radial line and all the zeros being on the unit circle in the complex plane. It can be expressed as

$$H(q^{-1}, n) = \frac{1 - a(n)q^{-1} + q^{-2}}{1 - \rho a(n)q^{-1} + \rho^2 q^{-2}}\,, \tag{1}$$

where n denotes the time index, $\rho < 1$ is a fixed hyperparameter indicating the radius of the complex conjugate pole pair; $a(n)$ is the single filter parameter defined as

$$a(n) = 2\cos[2\pi f(n)/f_s]\,, \tag{2}$$

where $f(n)$ is the notch frequency and f_s the sampling frequency. Finally, the discrete-time shift operator q is defined such that, for an input signal $y(n)$ [49],

$$q^{-k}y(n) = y(n-k)\,. \tag{3}$$

The time-varying filter parameter $a(n)$ represents the state of the KalmANF, recursively estimated to track the strongest frequency in the input signal over time. Consider an N-samples long input signal $y(n)$ and the direct form II of $H(q^{-1}, n-1)$ [50], implemented by means of a joint delay line for the feedforward and feedback paths as depicted in Fig. 1. The delay line signal $s(n)$ and the residual $e(n)$ at the output of the notch filter are defined as

$$s(n) = y(n) + \rho a(n-1)s(n-1) - \rho^2 s(n-2) \tag{4}$$

$$e(n) = s(n) - a(n-1)s(n-1) + s(n-2)\,. \tag{5}$$

In the Kalman formulation adopted by KalmANF, $s(n)$ represents the measurement. We can then define a state-space model from (4) and (5) (the detailed derivation is provided in [41]) as

$$\begin{bmatrix} a(n) \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} a(n-1) \\ 1 \end{bmatrix} + \begin{bmatrix} w(n) \\ 0 \end{bmatrix} \tag{6}$$

$$s(n) = \begin{bmatrix} s(n-1) & -s(n-2) \end{bmatrix} \begin{bmatrix} a(n) \\ 1 \end{bmatrix} + e(n)\,, \tag{7}$$

where we denoted the process noise with $w(n)$. The adaptive parameter $a(n)$ can then be recursively estimated by means of the Kalman filter update equations [51] using scalar operations, a memory of 2 past samples and three hyperparameters, namely the pole radius ρ, the presumed variance σ_e of the residual $e(n)$ and the presumed variance σ_w of the process noise $w(n)$. The state estimation error covariance $\hat{p}(n)$, the Kalman gain $k(n)$, and the parameter estimate $\hat{a}(n)$ are recursively updated. First, given the previous estimate $\hat{a}(n-1)$, $s(n)$ can be computed using (4). The covariance of the state estimation error is then computed as

$$\hat{p}(n|n-1) = \hat{p}(n-1) + \sigma_w \tag{8}$$

and used to calculate the Kalman gain

$$k(n) = \frac{s(n-1)}{s^2(n-1) + \frac{\sigma_e}{\hat{p}(n|n-1)}}\,. \tag{9}$$

The update equation to estimate the current value of the parameter $\hat{a}(n)$ can then be expressed as

$$\hat{a}(n) = \hat{a}(n-1) + k(n)e(n)\,, \tag{10}$$

where, by plugging the past parameter estimate into (5), the residual takes the value

$$e(n) = s(n) - \hat{a}(n-1)s(n-1) + s(n-2)\,. \tag{11}$$

Finally, the covariance of the prediction error is updated by

$$\hat{p}(n) = \left(1 - \frac{s^2(n-1)}{s^2(n-1) + \frac{\sigma_e}{\hat{p}(n|n-1)}}\right)\hat{p}(n|n-1)\,. \tag{12}$$

The recursively estimated parameter $\hat{a}(n)$ contains information on the instantaneous frequency tracked by the ANF, that is retrieved from (2) as

$$\hat{f}(n) = (f_s/2\pi)\arccos\left[\hat{a}(n)/2\right], \tag{13}$$

and constitutes the first feature for our siren identification network. Note that, from the Kalman filter formulation, the tracked frequency is not necessarily the fundamental, but the one with the highest energy. As a consequence, if the fundamental of a siren is missing or hidden in background noise, the higher harmonics can still be identified and tracked by the ANF.

Using the tracked frequency together with the input and the residual signal, we introduce a second feature that we name *power ratio*, defined as the ratio between the power of the suppressed sinusoid and that of the input signal. To obtain it we compute, at each time step, the power of the input signal,

Authorized licensed use limited to: KU Leuven University Library. Downloaded on January 05,2026 at 10:23:58 UTC from IEEE Xplore. Restrictions apply.
© 2025 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Audio, Speech and Language Processing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TASLPRO.2025.3646824

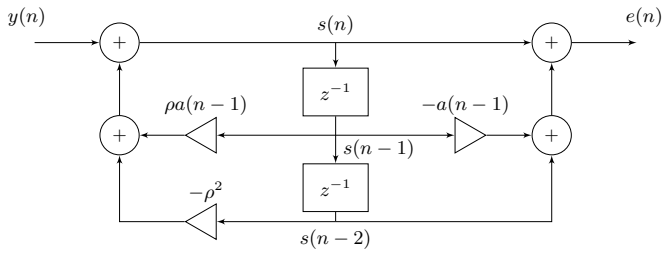


Fig. 1. The ANF biquadratic filter in the direct form II [41].

of the notched frequency component and of the residual (after $\hat{f}$ has been notched) as

$$P_y(n) = \lambda P_y(n-1) + (1-\lambda)y^2(n)\,, \tag{14}$$

$$P_e(n) = \lambda P_e(n-1) + (1-\lambda)e^2(n)\,, \tag{15}$$

$$P_f(n) = P_y(n) - P_e(n)\,, \tag{16}$$

where $\lambda = e^{-1/(\tau f_s)}$ and the time constant parameter τ for the recursive averaging operation constitutes an additional hyperparameter. Finally, the power ratio is defined as

$$P_{\text{ratio}}(n) = P_f(n)/P_y(n)\,. \tag{17}$$

We finally downsample P_{ratio} and $\hat{f}$ by a factor q_{down} (the final filter hyperparameter) to reduce the feature size. The proposed feature extraction method is summarized in Algorithm 1.

To ease the visual understanding of the defined features, in Fig. 2 we show examples with a noise sample and two different siren types (wail and yelp): in the upper plots, $\hat{f}(n)$ is overlaid to the spectrogram, showing the fundamental of the two sirens being effectively tracked by KalmANF. Note, however, that the feature is obtained directly from the time-domain audio waveform using Algorithm 1, without requiring to compute the spectrogram, that is reported only for visualization purposes. These examples underline the clear difference between the features extracted from a structured siren sound and from the unstructured background noise, that does not have a similar harmonic structure.

B. Architectures

To solve the siren identification task using the proposed frequency tracking features, we adopt a tiny CNN that we name ANFNet and illustrate in Tab. IV. The network takes as input the $\hat{f}$ and P_{ratio} features extracted from a single-channel audio sample. While the duration of the sample can be arbitrary, we consider in this work 2 s-long samples. The two features are combined into a 2-channel vector provided as input to ANFNet. The architecture is composed of three 1D convolutional layers (Conv1D) with, respectively, 10, 20 and 40 channels and kernel size 16, 8 and 4. The first two Conv1D layers are both followed by a MaxPool operation to reduce the feature size. The third one is followed by global average pooling (GlobAvgPool) that serves as interface between the convolutional block and the classification head, a MLP containing two FC layers with 40 and 20 neurons and a single-neuron output layer. All layers employ the ReLU activation function except for the output layer, where a sigmoid activation is used to estimate the siren presence probability. Dropout layers with 0.25 drop probability are added after each FC layer. With a 2 s input and $q_{\text{down}} = 5$, the network has 7.7 k parameters (6 to 9 times smaller than the four baselines).

Algorithm 1: The modified KalmANF algorithm

Initialize $s(0), s(1), \hat{a}(1), \hat{p}(1) = 0$
Set the hyperparamters $\sigma_e, \sigma_w, \rho, \tau, q_{\text{down}}$
for $n = 2$ to $N-1$ **do**
  $\hat{p}(n|n-1) = \hat{p}(n-1) + \sigma_w$
  $s(n) = y(n) + \rho\hat{a}(n-1)s(n-1) - \rho^2 s(n-2)$
  $k(n) = \frac{s(n-1)}{s^2(n-1) + \frac{\sigma_e}{\hat{p}(n|n-1)}}$
  $e(n) = s(n) - \hat{a}(n-1)s(n-1) + s(n-2)$
  $\hat{a}(n) = \hat{a}(n-1) + k(n)e(n)$
  $\hat{p}(n) = \left(1 - \frac{s^2(n-1)}{s^2(n-1) + \frac{\sigma_e}{\hat{p}(n|n-1)}}\right)\hat{p}(n|n-1)$
  if $|\hat{a}(n)| > 2$ **then**
    $\hat{a}(n) = 2\text{sgn}(\hat{a}(n))$
  end
  $\hat{f}(n) = (f_s/2\pi)\arccos\left[\hat{a}(n)/2\right]$
  $P_y(n) = \lambda P_y(n-1) + (1-\lambda)y^2(n)$
  $P_e(n) = \lambda P_e(n-1) + (1-\lambda)e^2(n)$
  $P_f(n) = P_y(n) - P_e(n)$
  $P_{\text{ratio}}(n) = P_f(n)/P_y(n)$
end
Downsample $\hat{f}$ and P_{ratio} by factor q_{down}

TABLE IV
THE PROPOSED ANFNET ARCHITECTURE, PROCESSING THE TWO FREQUENCY TRACKING FEATURES.

Layer	Kernel Size	Output channels
Input	-	2
Conv1D	16	10
MaxPool	2	-
Conv1D	8	20
MaxPool	2	-
Conv1D	4	40
GlobAvgPool	-	-
Fully Connected	-	40
Fully Connected	-	20
Output	-	1

To show the effectiveness of the proposed features across different architectures, we also evaluate a convolutional-recurrent neural network, named ANFRec for the experimental analysis: this architecture, summarized in Tab. V, shares the convolutional part with ANFNet, followed by a single gated recurrent unit (GRU) with 64 neurons, one 10-neuron FC layer with batch normalization and a single-neuron output layer. ANFRec exploits the same frequency tracking features as ANFNet: this larger model has 51.4 k parameters, comparable to the size of the spectrogram-based baselines (see Sec. II). A detailed analysis of the complexity of the proposed feature extraction method, compared to the log-mel spectrogram, and of the ANFNet model is carried out in Sec. VI.

C. Synthetic Training Data

As discussed in Sec. I, siren sounds exhibit a vast diversity among regions, countries, emergency vehicle types, and even

Authorized licensed use limited to: KU Leuven University Library. Downloaded on January 05,2026 at 10:23:58 UTC from IEEE Xplore. Restrictions apply.
© 2025 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Audio, Speech and Language Processing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TASLPRO.2025.3646824

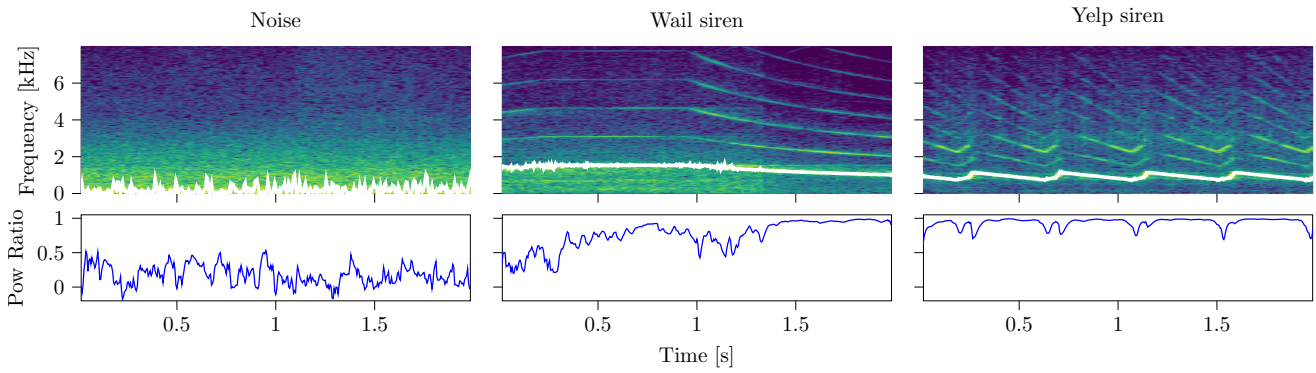


Fig. 2. Proposed features for siren and noise samples: the frequency tracked by the ANF algorithm is highlighted in white and overlaid to the full spectrogram in the upper plot, and the power ratio is depicted in the lower one [42].

TABLE V
THE ANFREC ARCHITECTURE, PROCESSING THE TWO FREQUENCY TRACKING FEATURES.

Layer	Kernel Size	Output channels
Input	-	2
Conv1D	16	10
MaxPool	2	-
Conv1D	8	20
MaxPool	2	-
Conv1D	4	40
GRU	-	64
Fully Connected + BN	-	10
Output	-	1

individual vehicles of the same type. Collecting an extensive dataset that captures such diversity is hence demanding and hardly feasible in a practical setting. Moreover, in [39] it has been shown that spectrogram-based models (VGGSiren, BCResNet and BCResNorm) have limited domain-robustness, exhibiting performance drops when trained on limited (and not diverse enough) data. In the preliminary study [42], we have shown that the proposed frequency tracking features lead to an improved generalization even when models are trained on extremely limited datasets. Motivated by this and to minimize the need for labeled real-world siren data for training, we leverage a fully synthetic dataset for training the proposed model.

To build the synthetic training dataset, we follow the procedure introduced in [39]. We first generate artificial stationary siren samples using a parametric model. A siren sound can in fact be modeled using a periodically modulated fundamental frequency, with period and modulation function depending on the siren type, and a set of higher harmonics. Equations to generate two-tone, wail and yelp sirens are provided in [39]: arbitrary diversity is hence created by randomly varying the siren parameters. We then simulate the motion of the emergency vehicle and the Doppler effect with the open-source pyroadacoustics simulator [52], using randomized smooth trajectories and a maximum speed of $40\,\mathrm{m/s}$. This way, we generate $24\,\mathrm{k}$ siren samples, each containing a different siren configuration, trajectory and speed. We finally add recorded environmental noise to the synthetic siren to create a complete acoustic scene.

For the background noise, we use samples from the SONYC dataset [53], pruned in order to avoid siren sounds being present in the background. In total, the SONYC dataset consists of $50\,\mathrm{h}$ of recording of urban soundscape, collected in different hours, days and locations within the city of New York, US.

The acoustic scene is directly synthesized within the data loader: for each sample, we randomly extract a $2\,\mathrm{s}$ segment from SONYC, and combine it to a randomly picked siren sample from the synthetic dataset with probability 0.5, to balance the siren and noise samples. The siren is added with a random signal-to-noise ratio (SNR) drawn from a uniform distribution defined in an interval optimized for each model, as will be discussed in Sec. IV-C.

This procedure relies entirely on synthetically generated sirens, hence no siren recording is needed. While, on the other hand, recorded background noise is needed for synthesizing the acoustic scene, differently from the scarcity of siren recordings, large-scale datasets containing urban noise are available and, hence, this does not constitute a limitation. Finally, in [39] it was shown that spectrogram-based networks (VGGSiren, BCResNet and BCResNorm) struggle to generalize from synthetic to real sirens and an alternative synthetic dataset based on augmenting a limited amount of recorded siren signals was introduced. Thanks to the improved domain-robustness guaranteed by the proposed frequency tracking features, in Sec. IV we illustrate that the proposed method overcomes this limitation and achieves good performance when trained using the described fully synthetic dataset (i.e., relying solely on synthetic siren sounds).

IV. EXPERIMENTS AND RESULTS

To assess the effectiveness of the frequency tracking features and evaluate how the proposed siren identification method compares with spectrogram-based baselines, we run experiments using the two real-world datasets introduced in Sec. IV-A.

We implement all neural networks (see Sec. II and Sec. III-B) using Pytorch Lightning [54]. We choose a sampling rate $f_s = 16\,\mathrm{kHz}$: for the KalmANF algorithm, we use the manually tuned hyperparameters $\rho = 0.99, \sigma_w =$

Authorized licensed use limited to: KU Leuven University Library. Downloaded on January 05,2026 at 10:23:58 UTC from IEEE Xplore. Restrictions apply.
© 2025 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Audio, Speech and Language Processing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TASLPRO.2025.3646824

$10^{-5}, \sigma_e = 0.66, q_{\text{down}} = 5, \tau = 0.02$, chosen based on the best loss obtained on validation data of the two real-world datasets. As pre-processing step, we normalize $\hat{f}$ to $f_s/2$ (note that P_{ratio} is normalized by definition). For the spectrogram-based baselines, we use a 1024 samples Hann window with 512 samples overlap and 128 mel channels to compute the mel spectrogram; we finally apply peak normalization to the computed mel spectrogram. Data augmentation is applied in all experiments: for the proposed models, we randomly swap the feature channels and apply white Gaussian noise with standard deviation 0.02, with probability 0.5. For the spectrogram-based baselines, we apply feature masking using SpecAugment [55]: in particular, we employ two time masks with a maximum width of 20 frames, and three frequency masks with a maximum width of 10 bins. In all experiments we use a binary cross-entropy loss, the Adam optimizer [56], a learning rate selected between 0.0001 and 0.001, and train the models for 400 epochs with early stopping if the loss computed on the validation set does not decrease for more than 15 epochs. We use the F1-score [57] as evaluation metric. To promote reproducibility, we provide accompanying code at [44].

A. Datasets

For our experimental campaign, we adopt two publicly available real-world datasets that were specifically proposed for siren classification. Both datasets include real recordings, with the presence of background traffic noise, moving sirens and Doppler effect. Three types of sirens are present in the data, namely two-tone, wail and yelp sirens.

First, we use the *sireNNet* dataset [58]: the dataset contains $1\,675$ $3\,\text{s}$-long files, including 421 traffic noise samples that capture the traffic and background noise present in an urban soundscape, whereas the remaining files belong to three types of emergency vehicles (ambulance, fire-truck and police car). Note that the three vehicle types do not correspond to different siren types (for example, samples from the ambulance and fire-truck classes can have the same wail siren), and we hence assign all of them to the siren class. Half of the siren samples are artificially generated for data augmentation purposes: in this work we exclude them and only use the non-augmented files. In total, $1\,039$ files of the sireNNet dataset are used, corresponding to all the 421 noise samples and to 627 siren samples. We divide these samples into training, validation and test sets with ratios $[0.8, 0.1, 0.1]$.

Second, we use the dataset [59], that we call *LSSiren*. The dataset contains $1\,800$ files with variable duration between $3\,\text{s}$ and $15\,\text{s}$, equally divided into siren and noise class samples. The dataset contains data recorded by the authors of [59] and collected from online sources, and has a wider diversity in terms of both sirens and background noise compared to sireNNet. Once again, we split the files into training, validation and test sets with ratios $[0.8, 0.1, 0.1]$.

We convert to mono and re-sample to $16\,\text{kHz}$ all files of both datasets. Moreover, since we use a fixed input length of $2\,\text{s}$, we take only the first two seconds of each sireNNet file, and divide the LSSiren files into non-overlapping $2\,\text{s}$ segments. Note that this operation is performed after splitting the files into training, validation and test sets: hence, all segments extracted from a file belong to the same set, which prevents the risk of data contamination. After this operation, the resulting size of the LSSiren dataset is of $6\,211$ $2\,\text{s}$-long files (thus, LSSiren is around 6 times larger than sireNNet).

TABLE VI
AVERAGE AND STANDARD DEVIATION OF F1-SCORES, OBTAINED BY MODELS TRAINED ON FULL REAL-WORLD DATASETS AND EVALUATED ON IN-DOMAIN AND CROSS-DOMAIN TEST DATA.

Model	In-domain F1 [%]	Cross-domain F1 [%]	# Params.
VGGSiren	98.75 ± 0.37	84.62 ± 15.59	$72.9\,\text{k}$
BCResNet	99.2 ± 1.12	91.07 ± 11.44	$45.7\,\text{k}$
BCResNorm	98.13 ± 0.25	$\mathbf{92.11 \pm 2.54}$	$46.2\,\text{k}$
CPMobile	$\mathbf{99.28 \pm 1.01}$	91.29 ± 7.52	$60.2\,\text{k}$
LogRegr	82.07 ± 4.50	79.44 ± 6.11	$6.4\,\text{k}$
ANFNet	96.07 ± 3.20	$\mathbf{91.67 \pm 5.84}$	$7.7\,\text{k}$
ANFRec	$\mathbf{97.27 \pm 2.67}$	91.32 ± 6.12	$51.4\,\text{k}$

We run three sets of experiments: first, we evaluate the proposed method in a data-efficient setting in Sec. IV-B. Preliminary results of this assessment were introduced in our conference paper [42], that we expand here by evaluating multiple models and datasets. Second, we assess the performance when the network is trained entirely on synthetic data and tested on the two (unseen) real-world datasets in Sec. IV-C. Finally, we assess the effect of pre-training models on synthetic data and fine-tuning them using an increasing amount of real-world data in Sec. IV-D.

B. Data-efficient learning

To perform a data-efficient evaluation, we follow the framework introduced in [60] and divide the training set of both LSSiren and sireNNet into subsets of different size. Specifically, for each dataset we create subsets containing an increasing percentage of the total amount of training data, with ratios $0.25\%, 0.5\%, 1\%, 2\%, 4\%, 8\%, 16\%, 32\%, 64\%$ and 100% (i.e., full training set). The subsets are created such that:

- smaller splits are subsets of larger ones;
- the data distribution of all subsets follows that of the full training set;

The validation and test sets are instead used in their entirety, without any splitting. We run all experiments on 8 different folds, randomly generated for each subset, and compute the mean and standard deviation of the results.

We first run a preliminary experiment to select the best performing architectures for the spectrogram-based baselines and the proposed method. To this end, we train VGGSiren, BCResNet, BCResNorm, CPMobile, ANFNet and ANFRec using the entire training set (i.e., 100% split) of the two datasets and compute the in-domain and cross-domain F1-score. The in-domain F1 is computed by averaging the F1 obtained when training and evaluating on LSSiren and when training and evaluating on sireNNet. Similarly, the cross-domain F1 is computed by averaging the F1 obtained when training on LSSiren and evaluating on sireNNet and vice versa. The average scores and model sizes are reported in Tab. VI for

Authorized licensed use limited to: KU Leuven University Library. Downloaded on January 05,2026 at 10:23:58 UTC from IEEE Xplore. Restrictions apply.
© 2025 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Audio, Speech and Language Processing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TASLPRO.2025.3646824

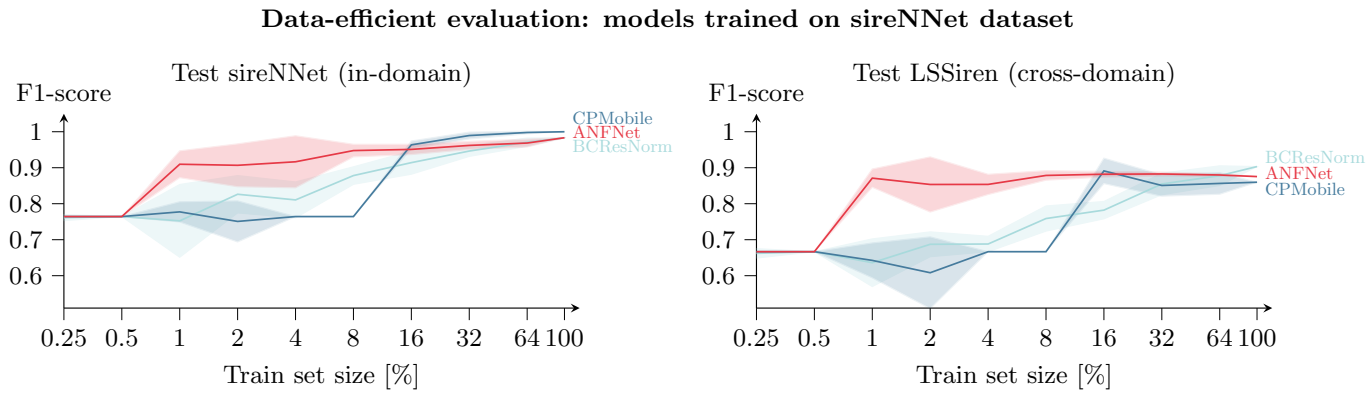


Fig. 3. Average (solid line) and standard deviation (shaded area) of the F1-scores for the BCResNorm and CPMobile baselines and the proposed ANFNet, when trained in a data-efficient setting using the sireNNet dataset. The full training set (100%) contains 839 2 s samples.

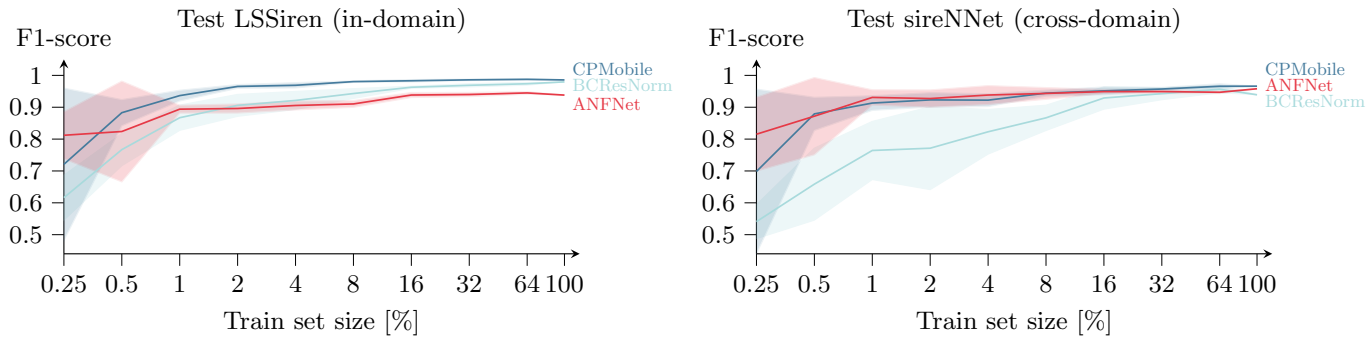


Fig. 4. Average (solid line) and standard deviation (shaded area) of the F1-scores for the BCResNorm and CPMobile baselines and the proposed ANFNet, when trained in a data-efficient setting using the LSSiren dataset. The full training set (100%) contains 4879 2 s samples.

all architectures; we include an additional logistic regression classifier trained with the frequency tracking features, used as additional baseline. The number of parameters of the logistic regression is 6 400 and corresponds to the length of the downsampled features. Two aspects emerge from this preliminary experiment. First, as discussed in [39], BCResNet and BCResNorm achieve significantly better domain generalization performance compared to VGGSiren while reducing the model size, and their performance is comparable to the more modern CPMobile architecture. Second, ANFNet and ANFRec achieve a cross-domain performance comparable to BCResNet, BCResNorm and CPMobile, with ANFNet being significantly smaller. This indicates that the proposed features are effective inputs for multiple architectures of different sizes. As an additional remark, both proposed networks outperform the logistic regression baseline, indicating the importance of extracting higher level features from the proposed frequency tracking ones, rather than directly using them for classification. From this experiment, we select the two spectrogram-based architectures with the most promising cross-domain performance, namely BCResNorm and CPMobile. Similarly, we choose ANFNet for the frequency tracking-based architectures.

Using these selected architectures, we run the data-efficient evaluation campaign and report in Fig. 3 and Fig. 4 the performance of models trained on sireNNet and LSSiren, respectively. In both figures, in-domain and cross-domain performance are reported on the left and right plots, including the average (solid line) and standard deviation (shaded area) of the F1-score across the 8 folds, for each training subset. As expected, all networks suffer from a degraded performance as the training dataset size decreases. However, the results show that ANFNet achieves better performance when using smaller datasets and has a significantly improved domain-robustness compared to BCResNorm. Compared to CPMobile, ANFNet achieves a large performance gain when trained on subsplits of sireNNet data, and achieves comparable performance when trained on LSSiren and tested in a cross-dataset setting, while having a lower complexity (see Sec. VI). The improvement is more prominent when sireNNet data is used for training due to two main reasons. First, the LSSiren dataset contains a wider diversity compared to sireNNet, which guarantees to achieve a stronger performance also when using BCResNorm and CPMobile. Second, the two datasets have a significant difference in size (see Sec. IV-A), which is reflected also on the subsets. Note, for example, that the 4% split of sireNNet contains 34 samples: the subset of LSSiren having the closest absolute size is the 0.5% split, containing 24 samples. As a result, even on smaller subsets of LSSiren, the effective amount of training data is larger, supporting the better performance of the two spectrogram-based baselines. Finally, note that the 0.25% split of sireNNet contains just 2 samples, one per class,

Authorized licensed use limited to: KU Leuven University Library. Downloaded on January 05,2026 at 10:23:58 UTC from IEEE Xplore. Restrictions apply.
© 2025 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Audio, Speech and Language Processing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TASLPRO.2025.3646824

TABLE VII
AVERAGE AND STANDARD DEVIATION OF F1-SCORES, OBTAINED USING MODELS TRAINED ON SYNTHETIC DATA AND EVALUATED ON THE TWO REAL-WORLD DATASETS.

Model	F1-LSSiren [%]	F1-sireNNet [%]	# Params.
VGGSiren	70.31 ± 8.49	50.99 ± 8.19	72.9 k
BCResNet	70.57 ± 15.19	60.07 ± 20.84	45.7 k
BCResNorm	75.46 ± 5.63	67.8 ± 8.37	46.2 k
CPMobile	$\mathbf{79.40 \pm 4.24}$	$\mathbf{69.46 \pm 5.44}$	60.2 k
LogRegr	69.71 ± 4.16	66.08 ± 11.27	6.4 k
ANFNet	$\mathbf{84.42 \pm 4.58}$	94.44 ± 2.80	7.7 k
ANFRec	76.68 ± 2.26	$\mathbf{95.57 \pm 1.76}$	51.4 k

constituting an extremely challenging scenario for all models.

C. Synthetic training

Guided by the domain-robustness of the proposed method analyzed in the previous experiment, we then train models entirely on synthetic data, as discussed in Sec. III-C, and evaluate their ability to generalize to the two real-world datasets, unseen during training.

As the training data is generated in an online fashion by combining randomly selected synthetic sirens and recorded background noise audio segments with a randomly chosen SNR, we arbitrarily set the size of the training and validation sets for a single epoch to 800 and 200 samples, respectively. We manually tune the optimal SNR range used in the synthetic data generation for each model. The chosen ranges are $[-10, 10]\,\mathrm{dB}$ for VGGSiren, $[-5, 5]\,\mathrm{dB}$ for BCResNet, BCResNorm and CPMobile, and $[20, 30]\,\mathrm{dB}$ for the proposed architectures.

As done in the data-efficient experiment, we first select the best performing architectures for the spectrogram-based baselines and the proposed method among the entire architecture pool. Tab. VII reports the F1-scores obtained on the two real-world datasets by all models trained on synthetic data. The two best-performing models are ANFNet and CPMobile, which we employ for the following experimental analysis together with BCResNorm, for consistency with the previous experiments. The table shows that all spectrogram-based models trained on entirely synthetic data poorly generalize to real-world data, confirming the observations of [39]. The proposed frequency tracking features overcome this limitation providing enhanced domain-robustness and always outperforming the baselines: ANFNet achieves an average F1 of 89.43 on the two datasets, against the average 74.43 obtained by CPMobile and 71.63 obtained by BCResNorm. Moreover, note that even the logistic regression baseline exploiting frequency tracking features outperforms both VGGSiren and BCResNet.

We then compare the performance of CPMobile, BCResNorm and ANFNet when trained on synthetic data and on real-world data in the data-efficient setting used in Sec. IV-B. We report the results of the evaluation performed on sireNNet in Fig. 5 and on LSSiren in Fig. 6. In both figures, the models trained on synthetic data are denoted as ANFNet-Synth, CPMobile-Synth BCResNorm-Synth, and average and standard deviation results are shown; the average performance of the models trained on in-domain (left plot) and cross-domain (right plot) real-world data is reported as a dotted line (note that these lines correspond to the results shown in Fig. 3 and Fig. 4). The figures show the robustness of the proposed method to synthetic training. When considering the sireNNet dataset (Fig. 5), training ANFNet on synthetic data yields results comparable to training using the entire real-world datasets. This indicates that the network is able to learn a good representation from synthetic data, not requiring any real-world data in the training phase. For CPMobile and BCResNorm, on the other hand, training on real-world data is important and even small subsets of recorded data, either taken from sireNNet (in-domain) or LSSiren (cross-domain), produce a performance improvement on sireNNet compared to fully synthetic training. On the LSSiren dataset (Fig. 6), ANFNet achieves comparable performance when trained on synthetic data or on cross-domain recorded data, even in larger subsets. This confirms the good ability of ANFNet to generalize from synthetic to real-world data. Moreover, ANFNet-Synth outperforms CPMobile and BCResNorm trained with up to 16% of real-world data, and achieves comparable performance in larger splits. Using in-domain LSSiren data is instead beneficial for all architectures, that outperform the counterparts trained on synthetic data on all splits larger than 0.5%.

These results confirm the robustness provided by the proposed features against domain-shift, showing that not only in presence of limited datasets, but also in absence of any real-world training data the proposed features yield models with strong performance on multiple real datasets.

D. Fine-tuning

We finally evaluate the effect of fine-tuning. For ANFNet, CPMobile and BCResNorm, we select the best model pre-trained on synthetic data (i.e., from the 8 folds of the previous experiment, we choose the one yielding the highest F1-score). We then perform fine-tuning using an increasing number of real-world data, adopting the same data-efficient subsets discussed in Sec. IV-B, and evaluate the models on in-domain data (i.e., from the same dataset used for fine-tuning) and cross-domain data (i.e., from the other dataset, completely unseen during both pre-training and fine-tuning).

First, we compare the F1-scores obtained by the fine-tuned models with that obtained by the models trained entirely on synthetic data, and report the results on sireNNet in Fig. 7 and on LSSiren in Fig. 8. Fig. 7 indicates that fine-tuning on sireNNet does not improve the average performance of ANFNet. However, it has the positive effect of reducing the standard deviation, even when few samples are used for fine-tuning: the improvement is significant particularly when in-domain data is available (even with just the 2 samples of the 0.25% split, the standard deviation is almost reduced to zero). Fine-tuning, on the other hand, strongly helps CPMobile and BCResNorm, that with 16% of the available data achieve a performance comparable to ANFNet (that, however, achieves this performance with only synthetic training). On LSSiren, Fig. 8 indicates that all the three models benefit from in-domain fine-tuning. If only cross-domain recorded data are

Authorized licensed use limited to: KU Leuven University Library. Downloaded on January 05,2026 at 10:23:58 UTC from IEEE Xplore. Restrictions apply.
© 2025 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Audio, Speech and Language Processing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TASLPRO.2025.3646824

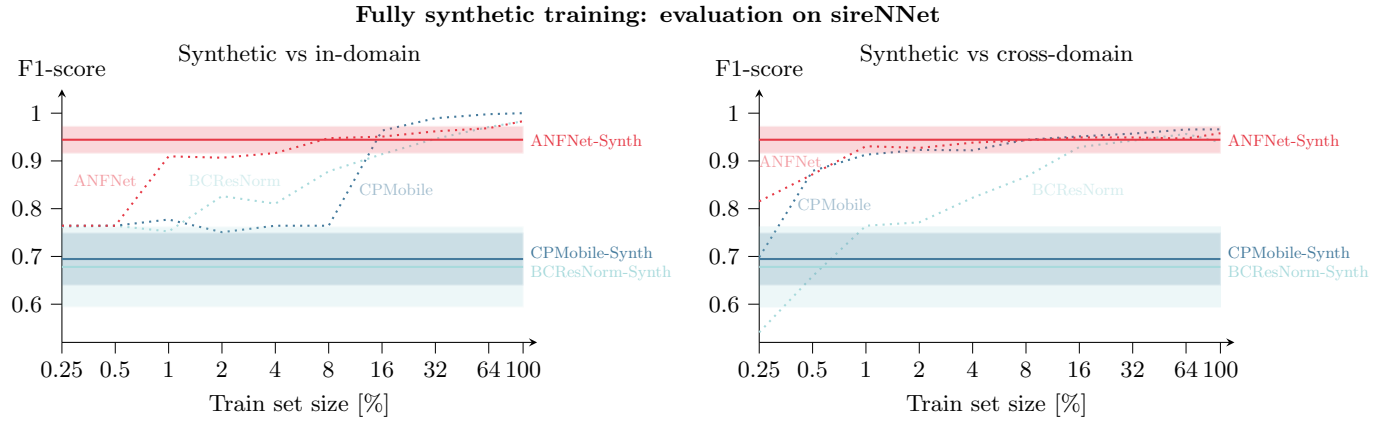


Fig. 5. Average (solid line) and standard deviation (shaded area) of the F1-score obtained on sireNNet using models trained on synthetic data (ANFNet-Synth, CPMobile-Synth, BCResNorm-Synth) and compared with the counterparts (ANFNet, CPMobile, BCResNorm) trained on in-domain and cross-domain real-world data (dotted lines), in a data-efficient setting. ANFNet-Synth outperforms CPMobile-Synth, BCResNorm-Synth, CPMobile and BCResNorm on smaller splits. Moreover, its performance is comparable with that of ANFNet.

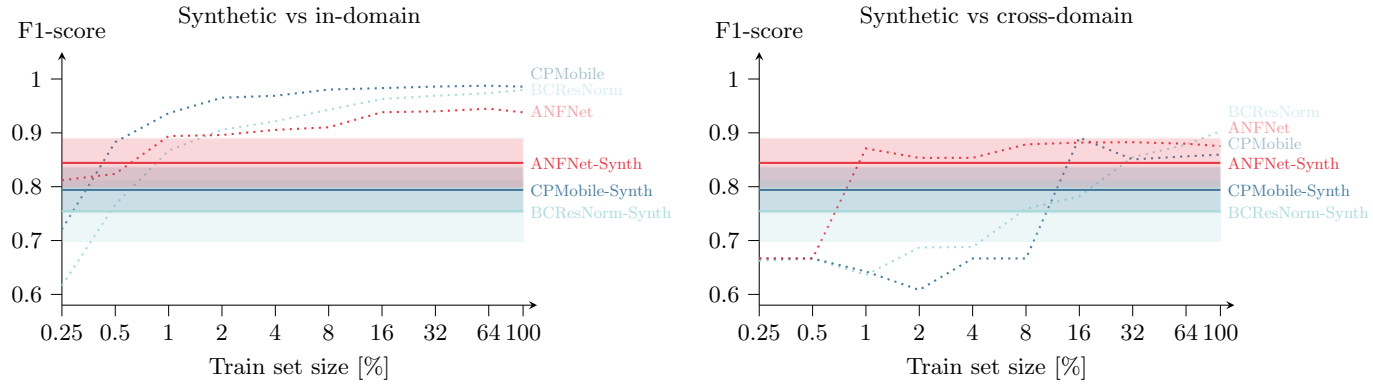


Fig. 6. Average (solid line) and standard deviation (shaded area) of the F1-score obtained on LS using models trained on synthetic data (ANFNet-Synth, CPMobile-Synth, BCResNorm-Synth) and compared with the counterparts (ANFNet, CPMobile, BCResNorm) trained on in-domain and cross-domain real-world data (dotted lines), in a data-efficient setting. While using larger sets of in-domain real-world data is the best choice, ANFNet-Synth outperforms CPMobile and BCResNorm on smaller splits. Moreover, when using cross-domain data, it outperforms CPMobile and BCResNorm, and achieves comparable performance to ANFNet on most splits.

available, instead, ANFNet does not significantly benefit from fine-tuning, and the performance of the fine-tuned CPMobile and of the ANFNet-Synth model trained entirely on synthetic data are comparable, underlining once again the domain-robustness of the proposed method.

We finally analyze the impact of fine-tuning on the three architectures by comparing the performance of: i) models trained from scratch using real-world data in a data-efficient setting (as in Sec. IV-B); ii) models trained entirely on synthetic data (as in Sec. IV-C); iii) models pre-trained on synthetic data and fine-tuned as discussed in the current section. In the left part of Fig. 9, Fig. 10 and Fig. 11 we compare the results for in-domain training and fine-tuning for ANFNet, BCResNorm, and CPMobile, respectively. In other words, in the left part of the figures, we trained/fine-tuned and evaluated the models using the same dataset (LSSiren to LSSiren, sireNNet to sireNNet) and averaged the results. In the right part of the figures, the cross-domain training and fine-tuning results are reported (in other words, the average between models trained/tuned on LSSiren and evaluated on sireNNet and vice versa). For ANFNet, the performance of models trained entirely on synthetic data is optimal and fine-tuning has a marginal impact, with the important effect of shrinking the standard deviation of the results (or, in other words, enhancing the stability of the models). For BCResNorm and CPMobile, training on synthetic data yields poor performance, and fine-tuning is clearly beneficial. In particular, compared to training from scratch on real-world data, pre-training helps when small training datasets are available, strongly boosting the achieved performance.

V. Discussion

In the previous section, we have shown quantitative results highlighting how the proposed method has an improved domain robustness compared to state-of-the-art spectrogram-based techniques, and can successfully be trained using an entirely synthetic dataset. To further investigate how the proposed frequency tracking features enhance domain robustness, we hereby qualitatively analyze the embedded features when projected onto a two-dimensional space using the t-SNE plot [61].

Authorized licensed use limited to: KU Leuven University Library. Downloaded on January 05,2026 at 10:23:58 UTC from IEEE Xplore. Restrictions apply.
© 2025 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Audio, Speech and Language Processing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TASLPRO.2025.3646824

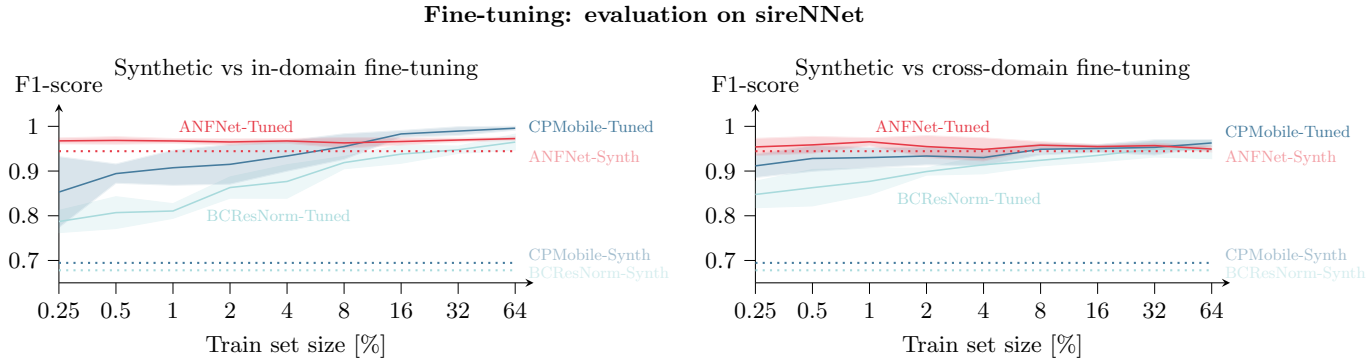


Fig. 7. Average (solid line) and standard deviation (shaded area) of the F1-score obtained on sireNNet by models pre-trained on synthetic data and fine-tuned on real-world in-domain (left) and cross-domain (right) data, in a data-efficient setting (ANFNet-Tuned, CPMobile-Tuned, BCResNorm-Tuned). The performance of the models trained on synthetic data without fine-tuning (ANFNet-Synth, CPMobile-Synth, BCResNorm-Synth) is reported for comparison using dotted lines.

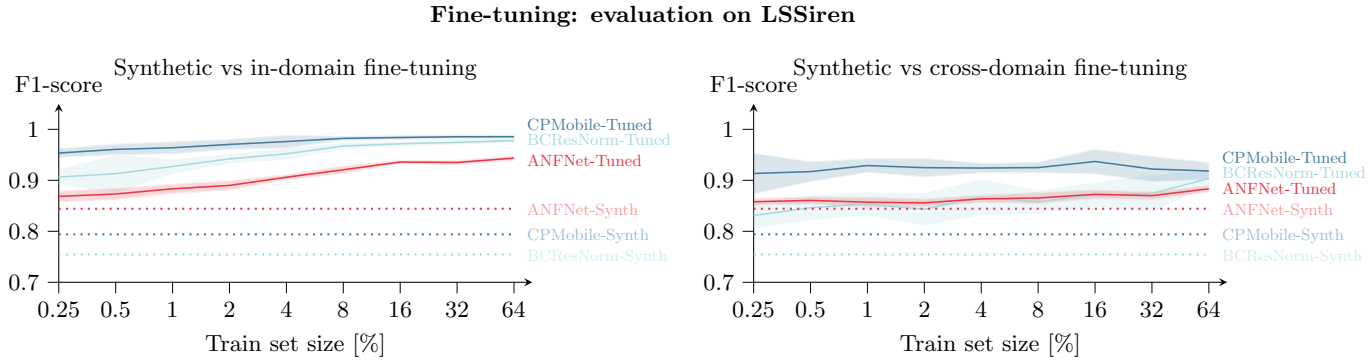


Fig. 8. Average (solid line) and standard deviation (shaded area) of the F1-score obtained on LSSiren by models pre-trained on synthetic data and fine-tuned on real-world in-domain (left) and cross-domain (right) data, in a data-efficient setting (ANFNet-Tuned, CPMobile-Tuned, BCResNorm-Tuned). The performance of the models trained on synthetic data without fine-tuning (ANFNet-Synth, CPMobile-Synth, BCResNorm-Synth) is reported for comparison using dotted lines.

For this purpose, we consider the best ANFNet, BCResNorm and CPMobile models trained on synthetic data out of the 8 training folds (i.e., the same ones used in Sec. IV-D) and extract embedded features after the GlobAvgPool of ANFNet, after the last convolutional layer of BCResNorm, and before the last convolutiona layer of CPMobile. We then compute t-SNE on the test sets of sireNNet and LSSiren, and on 800 synthetic samples for comparison, and show the plots for the three models in Fig. 12, where the three datasets are indicated by the different markers and the class is encoded in the color. While two clear clusters appear in the features extracted using all three models, corresponding to the siren and noise classes, the embedded features computed using ANFNet exhibit a more prominent separation. In the BCResNorm feature projection plot, several samples fall in the region between the two clusters, smearing the separation. Moreover, in ANFNet almost only LSSiren samples fall within the wrong cluster, while for BCResNorm this happens for samples of both datasets. In the CPMobile feature projection, it can be observed that the synthetic and real clusters appear more separated, indicating that the decision boundary learned on synthetic data might not be valid on the real-world datasets. This analysis suggests that the proposed frequency tracking features are less affected by domain-specific characteristics, and hence the ANFNet learns a domain-invariant representation of the two classes that yields the improved classification performance under domain-shift.

VI. Complexity Analysis

In this section, we compare the computational complexity of the proposed ANFNet and the two baselines BCResNorm and CPMobile to provide insights on the resources needed to run these models in real time. The analysis is focused on the inference complexity and is based on the number of Floating Point OPerations (FLOPs) needed to produce a prediction, based on a 2 s audio input.

We first analyze the feature extraction complexity. Both BCResNorm and CPMobile take as input log-mel spectrogram features. In the following, we consider that the Short-time Fourier Transform (STFT) of the input signal is computed using the Cooley-Tukey algorithm [62] and contains T time frames and N_{FFT} frequency bins, and we use an M-channel mel filter. To compute the log-mel spectrogram, four operations are needed, involving the following number of operations:

1) Short-time Fourier Transform (STFT) computation [62]: $T \cdot 5N_{\text{FFT}} \log_2 N_{\text{FFT}}$;
2) Power spectrum computation: $3T\left(\frac{N_{\text{FFT}}}{2}+1\right)$;
3) Mel filtering: $TM \cdot 2\left(\frac{N_{\text{FFT}}}{2}+1\right)$;
4) dB scaling: $8 \cdot TM$.

Authorized licensed use limited to: KU Leuven University Library. Downloaded on January 05,2026 at 10:23:58 UTC from IEEE Xplore. Restrictions apply.
© 2025 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Audio, Speech and Language Processing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TASLPRO.2025.3646824

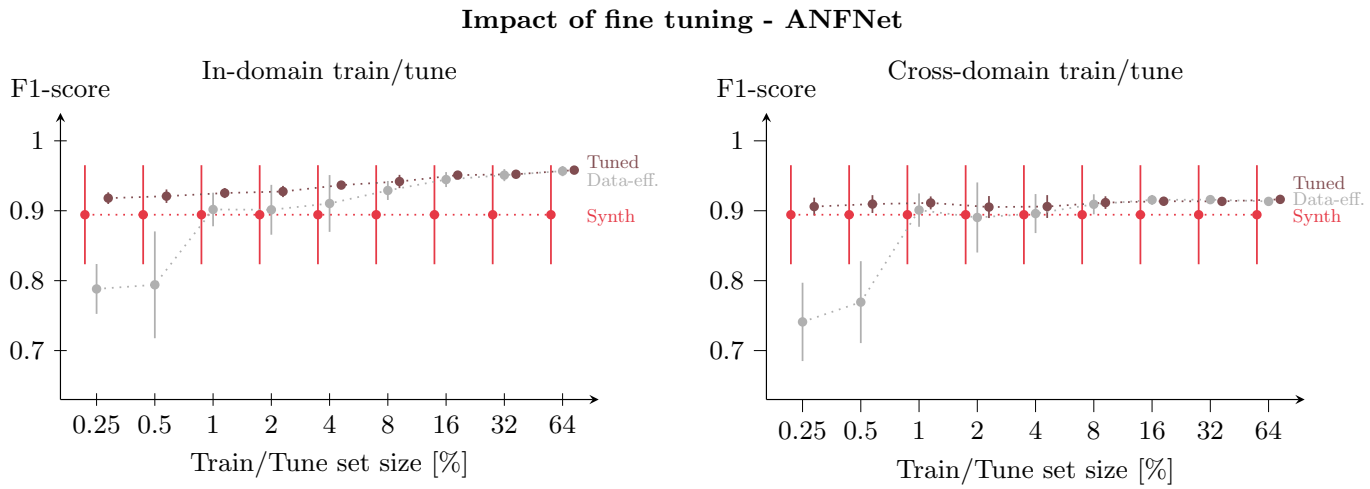


Fig. 9. Effect of fine-tuning using in-domain and cross-domain data on ANFNet. For in-domain training/tuning, models are tuned and evaluated on the same dataset for both LSSiren and sireNNet, and results are averaged. For cross-domain training/tuning, models are tuned on one dataset and evaluated on the other, and results are averaged.

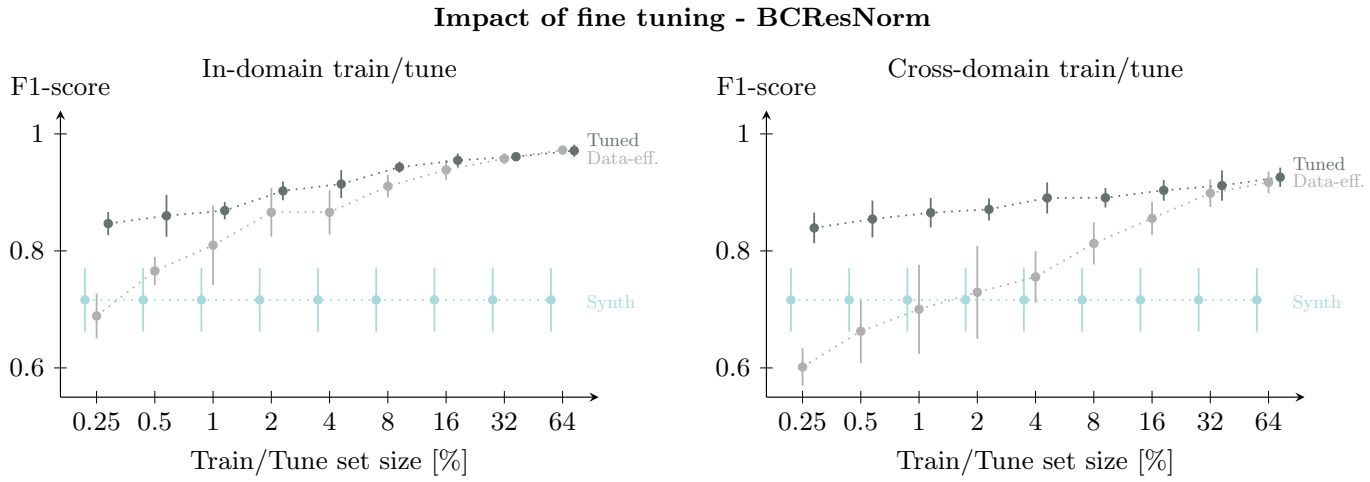


Fig. 10. Effect of fine-tuning using in-domain and cross-domain data on BCResNorm. For in-domain training/tuning, models are tuned and evaluated on the same dataset for both LSSiren and sireNNet, and results are averaged. For cross-domain training/tuning, models are tuned on one dataset and evaluated on the other, and results are averaged.

Substituting our parameters $N_{\text{FFT}} = 1024, M = 128, T = 64$, we obtain a total of $11.84\,\text{MFLOPs}$.

The proposed features are extracted using Algorithm 1:

1) The features are first extracted using the KalmANF algorithm, that involves a total of 13 additions, 18 multiplications, 5 divisions and 1 $\arccos$ computation per input sample. Assuming that divisions require 4 FLOPs and the $\arccos$ requires 8 [63], a total of 59 FLOPs are needed for each *input* sample.
2) The extracted features are downsampled using polyphase filtering, using a filter with 101 taps and a downsampling factor $q_{\text{down}} = 5$. In total, this operation requires 43 FLOPs per *output* sample [62].

Given a $2\,\text{s}$ audio input, a sampling frequency $f_s = 16\,\text{kHz}$ and $q_{\text{down}} = 5$, in total $2.16\,\text{MFLOPs}$ are needed to extract the features.

We finally evaluate the complexity of the neural networks by first computing their number of Multiply-Accumulate (MAC) operations using the *torchinfo* package, and then counting one FLOP operation as two MACs:

1) BCResNorm: $9.44\,\text{MFLOPs}$;
2) CPMobile: $13.36\,\text{MFLOPs}$;
3) ANFNet: $2.98\,\text{MFLOPs}$.

The total complexity is summarized in Tab. VIII. This analysis underlines how the proposed system has a significantly reduced computational complexity both in the feature extraction and in the model inference, making the proposed model attractive for real-time operation and deployment on resource-constrained devices.

VII. Conclusions

In this work, we presented a siren identification system based on frequency tracking features. Considering the harmonic nature of siren signals, in contrast to the unstructured urban background noise, we adopted the low-complexity

Authorized licensed use limited to: KU Leuven University Library. Downloaded on January 05,2026 at 10:23:58 UTC from IEEE Xplore. Restrictions apply.
© 2025 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Audio, Speech and Language Processing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TASLPRO.2025.3646824

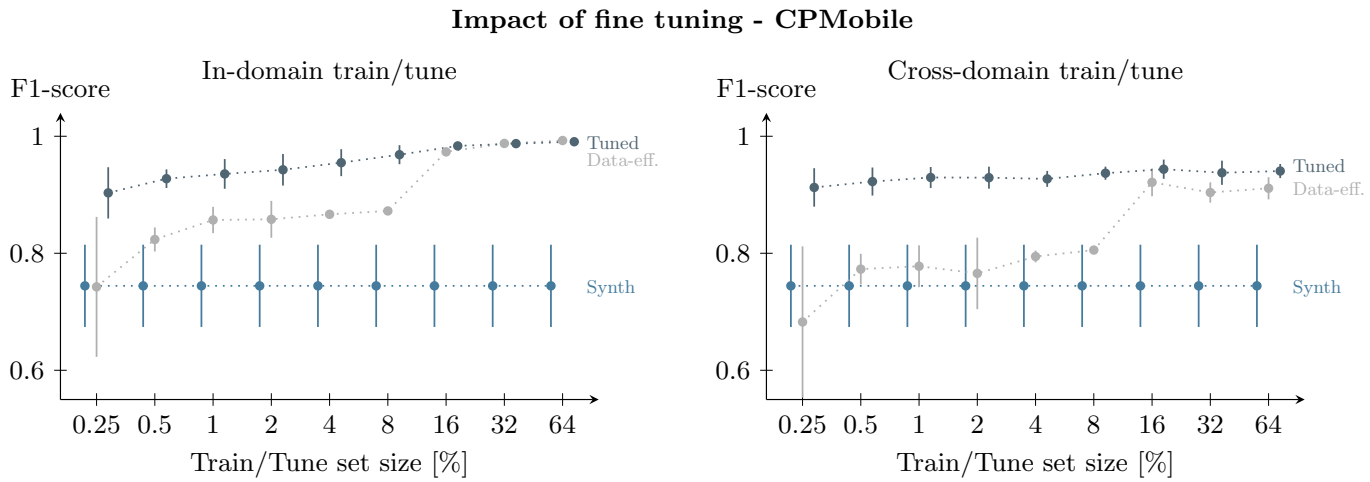


Fig. 11. Effect of fine-tuning using in-domain and cross-domain data on CPMobile. For in-domain training/tuning, models are tuned and evaluated on the same dataset for both LSSiren and sireNNet, and results are averaged. For cross-domain training/tuning, models are tuned on one dataset and evaluated on the other, and results are averaged.

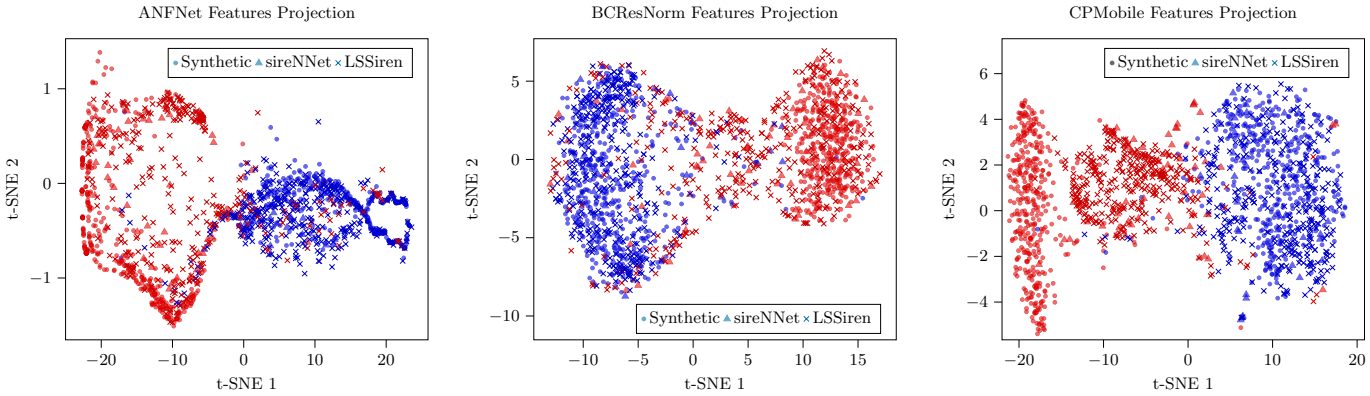


Fig. 12. t-SNE projection of the features computed by ANFNet, BCResNorm and CPMobile when trained entirely using synthetic data. The features of sireNNet, LSSiren and synthetic data are reported, with the two classes indicated by the color (red for siren, blue for noise).

TABLE VIII
COMPUTATIONAL COMPLEXITY OF THE PROPOSED ANFNET MODEL AND OF THE TWO BASELINES BCRESNORM AND CPMOBILE IN TERMS OF FLOPS.

Model	Feature Extraction	Neural Network	Total
BCResNorm	11.84 M	9.44 M	21.28 M
CPMobile	11.84 M	13.36 M	25.2 M
ANFNet	**2.16 M**	**2.98 M**	**5.14 M**

KalmANF algorithm to track the highest-energy frequency, usually corresponding to the fundamental when a siren sound is active, and the ratio between the power of the tracked frequency and of the input signal. We used these features as input to a small-scale 1D convolutional neural network to solve the identification task, and showed that, thanks to the characteristics of the proposed features, the model achieves a strong robustness to domain shift. Thanks to this advantage, we leveraged a fully synthetic dataset to train the proposed method and achieved good generalization to two real-world evaluation datasets, unseen during training. The proposed system outperforms two spectrogram-based state-of-the-art method both when limited data is available for training and when only the synthetic training data is used. Finally, the proposed method has the advantage of a significantly reduced model complexity, making it promising for deployment in the automotive sector.

ACKNOWLEDGMENTS

This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No. 956962, from KU Leuven internal funds C3/23/056, and from FWO Research Project G0A0424N. This paper reflects only the authors' views and the Union is not liable for any use that may be made of the contained information. The resources and services used in this work were provided by the VSC (Flemish Supercomputer Center), funded by the Research Foundation - Flanders (FWO) and the Flemish Government.

REFERENCES

[1] F. Arena, G. Pau, and A. Severino, "An Overview on the Current Status and Future Perspectives of Smart Cars," *Infrastructures*, vol. 5, no. 7, p. 53, Jun. 2020.

Authorized licensed use limited to: KU Leuven University Library. Downloaded on January 05,2026 at 10:23:58 UTC from IEEE Xplore. Restrictions apply.
© 2025 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Audio, Speech and Language Processing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TASLPRO.2025.3646824

[2] R. Hussain and S. Zeadally, "Autonomous Cars: Research Results, Issues, and Future Challenges," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 2, pp. 1275–1313, 2019.

[3] L. Marchegiani and X. Fafoutis, "How Well Can Driverless Vehicles Hear? A Gentle Introduction to Auditory Perception for Autonomous and Smart Vehicles," *IEEE Intell. Transp. Syst. Mag.*, pp. 92–105, 2022.

[4] J. Yin, S. Damiano, M. Verhelst, T. van Waterschoot, and A. Guntoro, "Real-Time Acoustic Perception for Automotive Applications," in *Proc. 2023 Des. Autom. Test Europe Conf. Exhib. (DATE)*, Antwerp, Belgium, Apr. 2023.

[5] S. Djukanović, Y. Patel, J. Matas, and T. Virtanen, "Neural network-based acoustic vehicle counting," in *Proc. 29th Europ. Signal Process. Conf. (EUSIPCO)*, Dublin, Ireland, Sep. 2021, pp. 561–565.

[6] S. Damiano, L. Bondi, S. Ghaffarzadegan, A. Guntoro, and T. van Waterschoot, "Can Synthetic Data Boost the Training of Deep Acoustic Vehicle Counting Networks?" in *Proc. 2024 IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, Seoul, Korea, Republic of, Apr. 2024, pp. 631–635.

[7] S. N. Dandare and S. V. Dudul, "Support vector machine based multiple fault detection in an automobile engine using sound signal," *J. Electron. Elect. Eng.*, vol. 3, no. 1, pp. 59–63, Mar. 2012.

[8] J. Libby and A. J. Stentz, "Using sound to classify vehicle-terrain interactions in outdoor environments," in *Proc. 2012 IEEE Int. Conf. Robot. Autom. (ICRA)*, St Paul, USA, May 2012, pp. 3559–3566.

[9] K. Choudhury and D. Nandi, "Review of Emergency Vehicle Detection Techniques by Acoustic Signals," *Trans. Indian National Acad. Eng.*, vol. 8, no. 4, pp. 535–550, Dec. 2023.

[10] F. Meucci, L. Pierucci, E. Del Re, L. Lastrucci, and P. Desii, "A real-time siren detector to improve safety of guide in traffic environment," in *Proc. 16th Europ. Signal Process. Conf. (EUSIPCO)*, Lausanne, Switzerland, Aug. 2008, pp. 1–5.

[11] J.-J. Liaw, W.-S. Wang, H.-C. Chu, M.-S. Huang, and C.-P. Lu, "Recognition of the Ambulance Siren Sound in Taiwan by the Longest Common Subsequence," in *Proc. 2013 IEEE Int. Conf. Syst. Man Cybernetics*, Manchester, UK, Oct. 2013, pp. 3825–3828.

[12] R.-A. Dobre and E.-V. Dumitrascu, "High-performance, low complexity yelp siren detection system," *Alexandria Engineering Journal*, vol. 109, pp. 669–684, Dec. 2024.

[13] D. Carmel, A. Yeshurun, and Y. Moshe, "Detection of alarm sounds in noisy environments," in *Proc. 25th Europ. Signal Process. Conf. (EUSIPCO)*, Kos, Greece, Sep. 2017, pp. 1839–1843.

[14] D. C. Chinvar, M. Rajat, R. L. Bellubbi, S. Sampath, and K. Guddad, "Ambulance Siren Detection using an MFCC based Support Vector Machine," in *Proc. 2021 IEEE Int. Conf. Mobile Networks Wireless Commun. (ICMNWC)*, Tumkur, Karnataka, India, Dec. 2021.

[15] M. K. Nandwana and T. Hasan, "Towards Smart-Cars That Can Listen: Abnormal Acoustic Event Detection on the Road," in *Proc. Interspeech 2016*, San Francisco, USA, Sep. 2016, pp. 2968–2971.

[16] L. Marchegiani and I. Posner, "Leveraging the urban soundscape: Auditory perception for smart vehicles," in *Proc. 2017 IEEE Int. Conf. Robot. Autom. (ICRA)*, Singapore, Singapore, May 2017, pp. 6547–6554.

[17] J. Schröder, S. Goetze, V. Grützmacher, and J. Anemüller, "Automatic acoustic siren detection in traffic noise by part-based models," in *Proc. 2013 IEEE Int. Conf. Acoust. Speech Signal Process.*, Vancouver, Canada, May 2013, pp. 493–497.

[18] F. Beritelli, S. Casale, A. Russo, and S. Serrano, "An automatic emergency signal recognition system for the hearing impaired," in *Proc. 12th IEEE Digital Signal Process. Workshop & 4th IEEE Signal Process. Educ. Workshop*, Teton National Park, USA, Sep. 2006, pp. 179–182.

[19] Y. Furletov, V. Willert, and J. Adamy, "Auditory Scene Understanding for Autonomous Driving," in *Proc. 2021 IEEE Intell. Veh. Symp. (IV)*, Nagoya, Japan, Jul. 2021, pp. 697–702.

[20] D. Pramanick, H. Ansar, H. Kumar, S. Pranav, R. Tengshe, and B. Fatimah, "Deep learning based urban sound classification and ambulance siren detector using spectrogram," in *Proc. 12th Int. Conf. Comput. Commun. Networking Technol. (ICCCNT)*, Kharagpur, India, Jul. 2021.

[21] M. Cantarini, A. Brocanelli, L. Gabrielli, and S. Squartini, "Acoustic Features for Deep Learning-Based Models for Emergency Siren Detection: An Evaluation Study," in *Proc. 12th Int. Symp. Image Signal Process. Anal. (ISPA)*, Zagreb, Croatia, Sep. 2021, pp. 47–53.

[22] D. F. Caetano Dos Santos and L. Boccato, "A Study of Emergency Siren Recognition on Resource-Constrained Devices," in *Intelligent Systems*. Cham: Springer International Publishing, 2022, vol. 13654, pp. 154–168.

[23] L. Marchegiani and P. Newman, "Listening for Sirens: Locating and Classifying Acoustic Alarms in City Scenes," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 10, pp. 17 087–17 096, 2022.

[24] F. Walden, S. Dasgupta, M. Rahman, and M. Islam, "Improving the Environmental Perception of Autonomous Vehicles using Deep Learning-based Audio Classification," *arXiv:2209.04075*, Sep. 2022.

[25] F. Bossio, P. Bestagini, L. Menescardi, F. Maver, D. Foscarin, M. Buccoli, and S. Pecorino, "A small-footprint deep-learning solution for real-time in-car emergency vehicle detection," in *Proc. 33rd Europ. Signal Process. Conf. (EUSIPCO)*, Isola delle Femmine, Italy, Sep. 2025.

[26] J. Sharma, O.-C. Granmo, and M. Goodwin, "Emergency Detection with Environment Sound Using Deep Convolutional Neural Networks," in *Proc. of 5th Int. Congr. Inf. Commun. Technol.*, 2021, pp. 144–154.

[27] H. Parineh, M. Sarvi, and S. A. Bagloee, "Detecting emergency vehicles With 1D-CNN using fourier processed audio signals," *Measurement*, vol. 223, p. 113784, Dec. 2023.

[28] V.-T. Tran and W.-H. Tsai, "Acoustic-Based Emergency Vehicle Detection Using Convolutional Neural Networks," *IEEE Access*, vol. 8, pp. 75 702–75 713, 2020.

[29] U. Mittal and P. Chawla, "Acoustic Based Emergency Vehicle Detection Using Ensemble of deep Learning Models," *Procedia Computer Science*, vol. 218, pp. 227–234, 2023.

[30] S.-M. Tsai, M.-L. Chuang, and W.-C. Wang, "Emergency Vehicle Siren Direction Detection Using Deep Learning," in *Proc. 2025 IEEE Int. Conf. Consum. Electron. (ICCE)*, Las Vegas, USA, Jan. 2025.

[31] M. Cantarini, L. Gabrielli, and S. Squartini, "Few-Shot Emergency Siren Detection," *Sensors*, vol. 22, no. 12, p. 4338, Jun. 2022.

[32] S. Kshirasagar, A. Guntoro, and C. Mayr, "Impact of Sliding Window Variation and Neuronal Time Constants on Acoustic Anomaly Detection Using Recurrent Spiking Neural Networks in Automotive Environment," *Algorithms*, vol. 17, no. 10, p. 440, Oct. 2024.

[33] H. Sun, X. Liu, K. Xu, J. Miao, and Q. Luo, "Emergency Vehicles Audio Detection and Localization in Autonomous Driving," *arXiv:2109.14797*, Sep. 2021.

[34] L. Banchero, F. Vacalebri-Lloret, J. M. Mossi, and J. J. Lopez, "Enhancing Road Safety with AI-Powered System for Effective Detection and Localization of Emergency Vehicles by Sound," *Sensors*, vol. 25, no. 3, p. 793, Jan. 2025.

[35] V.-T. Tran and W.-H. Tsai, "Audio-Vision Emergency Vehicle Detection," *IEEE Sensors Journal*, vol. 21, no. 24, pp. 27 905–27 917, Dec. 2021.

[36] L. Gabrielli, L. Migliorelli, M. Cantarini, A. Mancini, and S. Squartini, "An advanced multimodal driver-assistance prototype for emergency-vehicle detection," *Integrated Computer-Aided Engineering*, vol. 31, no. 4, pp. 381–399, Jul. 2024.

[37] M. Zohaib, M. Asim, and M. ELAffendi, "Enhancing Emergency Vehicle Detection: A Deep Learning Approach with Multimodal Fusion," *Mathematics*, vol. 12, no. 10, p. 1514, May 2024.

[38] R. Ravi and J. Kanniappan, "ZeroEVNet: A multimodal zero-shot learning framework for scalable emergency vehicle detection," *Expert Systems with Applications*, vol. 275, p. 126934, May 2025.

[39] S. Damiano, B. Cramer, A. Guntoro, and T. van Waterschoot, "Synthetic data generation techniques for training deep acoustic siren identification networks," *Frontiers Signal Process.*, vol. 4, p. 1358532, Jul. 2024.

[40] B. Kim, S. Chang, J. Lee, and D. Sung, "Broadcasted Residual Learning for Efficient Keyword Spotting," in *Proc. Interspeech 2021*, Brno, Czech Republic, Sep. 2021, pp. 4538–4542.

[41] R. Ali and T. van Waterschoot, "A Frequency Tracker Based on a Kalman Filter Update of a Single Parameter Adaptive Notch Filter," in *Proc. 26th Int. Conf. Digital Audio Effects (DAFx)*, Copenhagen, Denmark, Sep. 2023.

[42] S. Damiano, T. Dietzen, and T. van Waterschoot, "Frequency Tracking Features for Data-efficient Deep Siren Identification," in *Proc. 9th Int. Workshop Detection Classification Acoust. Scenes Events (DCASE)*, Tokyo, Japan, Oct. 2024, pp. 36–40.

[43] F. Schmid, T. Morocutti, S. Masoudian, K. Koutini, and G. Widmer, "Distilling the knowledge of transformers and CNNs with CP-mobile," in *Proc. Detection and Classification of Acoustic Scenes and Events 2023 Workshop (DCASE2023)*, 2023, pp. 161–165.

[44] S. Damiano and T. Dietzen, "An ANF-based siren identification system," Github Repository, 2024. [Online]. Available: https://github.com/steDamiano/anf-siren-identification

[45] F. Schmid, "DCASE2025 - Task 1 - Baseline Systems," GitHub Repository, 2025. [Online]. Available: https://github.com/CPJKU/dcase2025_task1_baseline

[46] F. Schmid, P. Primus, T. Heittola, A. Mesaros, I. Martín-Morató, and G. Widmer, "Low-complexity acoustic scene classification with device information in the dcase 2025 challenge," *arXiv:2505.01747*, 2025.

Authorized licensed use limited to: KU Leuven University Library. Downloaded on January 05,2026 at 10:23:58 UTC from IEEE Xplore. Restrictions apply.
© 2025 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.

This article has been accepted for publication in IEEE Transactions on Audio, Speech and Language Processing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TASLPRO.2025.3646824

[47] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. 3rd Int. Conf. Learn. Repr. (ICLR)*, San Diego, USA, May 2015.
[48] K. Hirano, S. Nishimura, and S. Mitra, "Design of Digital Notch Filters," *IEEE Trans. Commun.*, vol. 22, no. 7, pp. 964–970, Jul. 1974.
[49] T. van Waterschoot and M. Moonen, "Fifty years of acoustic feedback control: State of the art and future challenges," *Proc. IEEE*, vol. 99, no. 2, pp. 288–327, 2011.
[50] J. Travassos-Romano and M. Bellanger, "Fast least squares adaptive notch filtering," in *Proc. 1988 IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, New York, USA, 1988, pp. 1391–1394.
[51] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Journal of Basic Engineering*, vol. 82, no. 1, pp. 35–45, Mar. 1960.
[52] S. Damiano and T. van Waterschoot, "Pyroadacoustics: a Road Acoustics Simulator Based on Variable Length Delay Lines," in *Proc. 25th Int. Conf. Digital Audio Effects (DAFx20in22)*, Vienna, Austria, Sep. 2022, pp. 216–223.
[53] M. Cartwright, J. Cramer, A. E. Mendez Mendez, Y. Wang, H.-H. Wu, V. Lostanlen, M. Fuentes, G. Dove, C. Mydlarz, J. Salamon, O. Nov, and J. P. Bello, "SONYC-UST-V2: an Urban Sound Tagging Dataset with Spatiotemporal Context," in *Proc. 5th Workshop Detection Classification Acoust. Scenes Events (DCASE)*, Tokyo, Nov. 2020, pp. 16–20.
[54] W. Falcon and The PyTorch Lightning team, "PyTorch Lightning," Github Repository, Mar. 2019. [Online]. Available: https://github.com/Lightning-AI/lightning
[55] D. S. Park, W. Chan, Y. Zhang, C.-C. Chiu, B. Zoph, E. D. Cubuk, and Q. V. Le, "SpecAugment: A Simple Data Augmentation Method for Automatic Speech Recognition," in *Proc. Interspeech 2019*, Graz, Austria, Sep. 2019.
[56] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. 3rd Int. Conf. Learn. Represent. (ICLR)*, San Diego, USA, 2015.
[57] I. J. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.
[58] A. Shah and A. Singh, "sireNNet-Emergency Vehicle Siren Classification Dataset For Urban Applications," Mendeley Data, 2023, doi: 10.17632/j4ydzzv4kb.1.
[59] M. Asif, M. Usaid, M. Rashid, T. Rajab, S. Hussain, and S. Wasi, "Large-scale audio dataset for emergency vehicle sirens and road noises," *Scientific Data*, vol. 9, no. 1, p. 599, Oct. 2022.
[60] F. Schmid, P. Primus, T. Heittola, A. Mesaros, I. Martín-Morató, K. Koutini, and G. Widmer, "Data-efficient low-complexity acoustic scene classification in the dcase 2024 challenge," *arXiv:1706.10006*, 2024.
[61] L. J. van der Maaten and G. E. Hinton, "Visualizing High-Dimensional Data Using t-SNE," *J. Mach. Learn. Res.*, vol. 9, no. nov, pp. 2579–2605, 2008.
[62] A. V. Oppenheim and R. W. Schafer, *Discrete-Time Signal Processing*, 3rd ed. Prentice Hall Press, 2009.
[63] S. Chimmalgi, P. J. Prins, and S. Wahls, "Fast nonlinear fourier transform algorithms using higher order exponential integrators," *IEEE Access*, vol. 7, pp. 145 161–145 176, 2019.

Authorized licensed use limited to: KU Leuven University Library. Downloaded on January 05,2026 at 10:23:58 UTC from IEEE Xplore. Restrictions apply.
© 2025 IEEE. All rights reserved, including rights for text and data mining and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission. See https://www.ieee.org/publications/rights/index.html for more information.