Recognition of Unknown Radar Emitters With Machine Learning

SABINE APFELD
ALEXANDER CHARLISH, Senior Member, IEEE
Fraunhofer Institute for Communication, Information Processing and Ergonomics, 53343 Wachtberg, Germany

Classifiers based on machine learning are usually trained to distinguish between several known classes. For an electronic intelligence application, however, it is of great importance to recognize if an intercepted signal belongs to an unknown radar emitter. In the machine learning literature, this task is called open-set recognition. This article investigates six approaches in several configurations to recognize unknown emitters. It is based on a hierarchical emission model that understands emissions as a language with an inherent hierarchical structure. We consider two general approaches, which are the "memoryless" Markov chain and the Long Short-Term Memory recurrent neural network, which is especially designed to "remember" the past. The performance is demonstrated with two evaluation metrics in ten scenarios that contain different combinations of known and unknown emitters. An evaluation with corrupted data provides an estimate on the methods' accuracies under challenging conditions. The results show that unknown emitters that do not use known waveforms are reliably recognized even with corrupted data, while unknown emitters that are more similar to known ones are harder to detect.

Manuscript received February 12, 2021; revised May 12, 2021; released for publication June 21, 2021. Date of publication July 20, 2021; date of current version December 6, 2021.

DOI. No. 10.1109/TAES.2021.3098125

Refereeing of this contribution was handled by D. A. Garren.

Authors' addresses: S. Apfeld and A. Charlish are with Department of Sensor Data and Information Fusion, Fraunhofer FKIE, 53343 Wachtberg, Germany E-mail: (sabine.apfeld@fkie.fraunhofer.de; alexander.charlish@fkie.fraunhofer.de). *(Corresponding author: Sabine Apfeld.)*

I. INTRODUCTION

The goal of electronic intelligence (ELINT) is to collect information about radar systems by intercepting and analyzing their signals. A passive ELINT receiver intercepts the radar signals in the environment and converts received pulses into pulse descriptor words (PDWs), which consist of at least the parameters radio frequency (RF) and pulsewidth (PW), as well as the time of arrival that is used to determine the pulse repetition interval (PRI) or pulse repetition frequency (PRF). Usually, several emitters are active such that PDWs from different emitters are interleaved. Before an analysis, the PDWs must, hence, be deinterleaved (see, e.g., [1]). After this step, separated PDW sequences are obtained, but no information about the emitters' identities is given. Fig. 1 provides a visualization. The input to an analysis method or classifier is shown on the right. It contains three separated PDW sequences that the classifier processes sequentially. However, it is not known whether the sequences belong to one, two, or three emitters, and usually, the classifier's input consists of deinterleaved PDW sequences from different emitters.

Traditionally, the waveform parameters extracted from the stream of PDWs are compared to entries in a database to identify the emitter type. However, with the introduction of agile multifunction radars, this pattern matching approach no longer provides satisfactory performance. Therefore, methods based on machine learning techniques such as neural networks have been proposed [2]–[7]. For an ELINT application, recognition of unknown emitters is of great importance. If unknown radars are detected, the corresponding signals need to be prioritized and recorded such that further analysis can be performed. However, classifiers such as neural networks are generally trained to identify a set of known classes, which is sometimes called the closed-set assumption. Recognizing if an input does not belong to any of the known classes is referred to as open-set or open-world recognition (see, e.g., [8]–[13]). The major differences between open-set recognition, classification, and anomaly detection are the goal, the available training data, and the output that the classifier provides in each task. Anomaly detection could also be used to classify input as unknown, but an anomaly detector does not distinguish between several known classes. A classification method, on the other hand, is not suited to recognize unknown input. An anomaly detector can be combined with a classification approach to yield a method for open-set recognition. The different types of input that a classifier for open-set recognition needs to handle are the following [9]:

1) *Known Classes*: the set of classes that the classifier should recognize and identify, denoted by $\mathbb{K}$ below;
2) *Known Unknown Classes*: the set of classes $\mathbb{V}$ that should be classified as unknown/rejected. Type and structure are known and either training data is available or can be generated;
3) *Unknown Unknown Classes*: data of unknown unknown classes, denoted by $\mathbb{U}$, should as well be

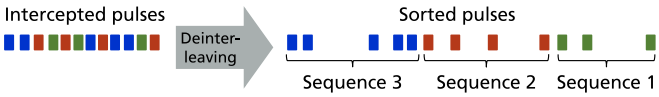

Fig. 1. Input to the classifier consists of sequences of deinterleaved PDWs, which possibly belong to different emitters [2].

classified as unknown, but is unavailable at training time and only encountered at test time.

Open-set recognition can be regarded as a learning task with these properties [12]:

1) If the input $\boldsymbol{x}$ belongs to a known class $k_c \in \mathbb{K}$, i.e., $\text{class}(\boldsymbol{x}) = k_c$, the label y corresponds to the class index $\text{idx}(k_c) = c$; otherwise, $y = \emptyset$, with $\emptyset$ meaning "unknown."
2) The goal is to train a classifier $\hat{c}$ that returns $\hat{c}(\boldsymbol{x}) = c$ for all $\boldsymbol{x}$ with $\text{class}(\boldsymbol{x}) = k_c \in \mathbb{K}$ and $\hat{c}(\boldsymbol{x}) = \emptyset$ otherwise.
3) Predicting the class $\hat{c}(\boldsymbol{x}) = \emptyset$ is called the reject option. The learning process is supposed to adjust the selection of this option.

This article investigates six methods to recognize unknown radar emitters, which are either based on a "memoryless" Markov chain (MC) or a Long Short-Term Memory (LSTM) recurrent neural network [14] that possesses a "memory." These approaches build on previous work of the authors on modeling and predicting emissions [15], [16], as well as the identification of the radar emitter type [2], [16].

A. Related Work

There are two alternative principles for open-set recognition. The first one is to compare the classifier's output $\hat{\boldsymbol{y}} = (\hat{y}_0, \hat{y}_1, \ldots, \hat{y}_{C-1})$ for each class $k_c \in \mathbb{K} = \{k_0, k_1, \ldots, k_{C-1}\}$ to a threshold δ and reject the input as unknown if none of the values exceeds it, i.e.,

$$\hat{c}(\boldsymbol{x}) = \begin{cases} T(\hat{\boldsymbol{y}}), & \text{if } \max(\hat{\boldsymbol{y}}) \geq \delta \\ \emptyset, & \text{else} \end{cases} \tag{1}$$

where $T(\hat{\boldsymbol{y}})$ denotes the index c of the most probable known class k_c. The second approach is to add an extra class with index C called "unknown" to the possible output classes. The classifier can be trained on the classes in $\mathbb{K}$ and $\mathbb{V}$ but obviously not on $\mathbb{U}$.

The work presented in [10] introduces *OpenMax*, which estimates the probability that the input belongs to an unknown class, to replace the commonly used softmax layer in a neural network with the softmax function being defined by

$$\text{softmax}(\tilde{y}_c) = \frac{\exp(\tilde{y}_c)}{\sum_{i=0}^{C-1} \exp(\tilde{y}_i)}, \quad c = 0, 1, \ldots, C-1. \tag{2}$$

The basis is the analysis of the logits, i.e., the activations $\tilde{\boldsymbol{y}}$ of the final dense layer before softmax. Per class, the logits of every correctly classified training input are combined into a mean activation vector to yield a single representation of the class. OpenMax then adapts the top k logits based on the distance between the input $\boldsymbol{x}$ and the mean activation vector of the class. In addition, a pseudo-activation of the unknown class is determined. If the probability for the unknown class is the highest after the normalization using softmax or all values are below a threshold, the input is rejected as unknown.

Although achieving good performance in comparison to directly using the softmax function with a threshold, it is not clear how to apply OpenMax to sequential data like radar signals. The mean activation vector would need to be calculated for many sequences with different offsets and lengths to capture the general distribution of the logits for a certain class, which does not seem to be practical.

A network is designed in [17] to find data samples that are close to the known classes but still rejected. The classifier's training data are augmented with this known unknown data, which results in a regularization of the network by increasing the uncertainty for unknown input. The work [13] introduces a new loss function, called the *entropic open-set (EOS) loss*, to evenly distribute the probability across known classes if the input belongs to an unknown class. This loss function is described in Section III, as well as the approach called *deep open classification (DOC)* presented in [11]. In addition to the EOS loss, the work [13] introduces the *objectosphere loss* as an extension, which targets the "deep feature layer" that exists in all convolutional neural networks but not in the LSTM architecture of this article. Hence, the evaluations of this article are restricted to the EOS loss.

Literature on recognizing an unknown emitter or signal for an ELINT application is very sparse. Moreover, most of the existing papers either use methods that have not established themselves and have been replaced by neural network types like LSTMs [18]–[20] or exhibit a lack of details on how training for the unknown class is performed [20], [21]. In [22], the *class probability output network* as introduced in [23] is employed, which normalizes the output of a classifier in order to obtain probability values, i.e., an alternative to using the softmax function.

B. Our Contributions

This article provides a thorough investigation of six approaches in several configurations to recognize unknown emitters based on the hierarchical emission model presented in [15], [24], and [25], which is briefly repeated in Section II-A. Four of the methods employ LSTMs and two are based on MCs. The LSTMs are implemented with three different loss functions, which are cross-entropy (CE), EOS, and DOC. To the best of the authors' knowledge, this is the first implementation of the EOS loss with LSTMs in general and the first implementation of the DOC loss with unidirectional LSTMs. Moreover, it is also the first application of LSTMs and MCs for the task of unknown emitter recognition.

In addition, an MC-based method for generating known unknown data is proposed. All considered classifiers for unknown emitter recognition are trained with five training cases, which differ in the contained known unknown data.

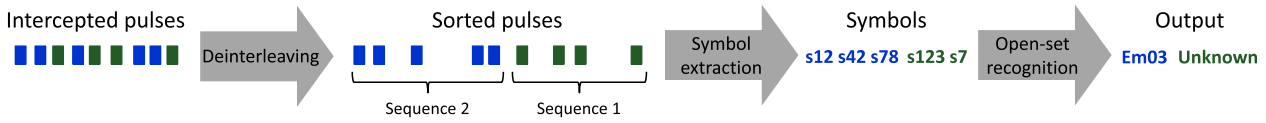

Fig. 2. Example of the signal processing chain. The pulses of two emitters are intercepted and deinterleaved. Afterward, the PDW sequences are mapped to symbols, e.g., syllables. The symbol sequences are processed by the open-set recognition method. Image based on [2].

Another training case that follows the closed-set assumption is used for the LSTM-CE and the MC, as shown in [2], [15], and [16]. The evaluation is performed with ten different test cases containing several combinations of classes from $\mathbb{K}$, $\mathbb{V}$, and $\mathbb{U}$. It is analyzed how the generated known unknown data influences the rejection accuracy for unknown input.

A compromise between the true and false rejection rate, as well as the accuracy for discriminating between known classes, is found by selecting the configuration of the classifier, which consists of the training case and the value of the threshold δ, if a threshold is used. This article estimates the expected performance for the distinction between known and unknown input when choosing the best configuration for the identification of known classes and *vice versa*. We also analyze whether a single classifier is enough for a good accuracy or a hierarchical combination of different classifiers is to be preferred. An evaluation with corrupted data further provides estimates on the robustness of the classifiers.

Section II details the radar data used in the evaluations of this article and describes the cases employed for training the approaches introduced in Section III. Section IV presents the results. Finally, Section V concludes this article.

II. RADAR DATA AND TRAINING CASES

A. Hierarchical Emission Model

A hierarchical emission model that considers the emitters as systems that speak a language [15], [24], [25] is the basis for the presented methods. In analogy to natural language, the different modeling levels are defined as follows:

1) *Letters*: Letters correspond to the radars' emitted pulses, which are defined, e.g., by PRI or PRF, RF, and PW.
2) *Syllables*: Syllables, which correspond to radar bursts, are common combinations of letters.
3) *Words*: Syllables can be combined to form words, which resemble radar dwells.
4) *Commands*: Commands describe word types or classes on a higher level of abstraction.
5) *Functions*: Different purposes of the emissions map to functions, e.g., searching or tracking targets.

Fig. 2 illustrates the processing chain. In this example, the pulses of two emitters (blue and green) are intercepted. At first, the pulses are sorted by common properties into two separated sequences, such that each sequence only contains the PDWs of a single emitter. After deinterleaving, the symbol extraction maps the PDW sequences to the symbols defined by the emission model. For example, the result might be a sequence of blue and a sequence of green syllables. Finally, these symbol sequences are the input to the open-set recognition method. Note that a low signal-to-noise ratio might lead to a missed or wrong detection of symbols, but does not influence the processing after a correct detection. The radar symbols are encoded into a numerical representation using word embeddings, namely, *word2vec* [26], [27]. Each emitter e has its individual dictionary Ω_e^l at modeling level l with $l \in \{\text{letters, syllables, words, commands, functions}\}$. In addition, there exists a global dictionary Ω^l containing all symbols. word2vec introduces a symbol called UNK, which is a placeholder for new input without an embedding. Details on the parameters are given in [2] and [15]. As the identification accuracies for letters, commands, and functions are not satisfactory for the LSTMs even without unknown emitters [2], [16], only syllables and words are considered, i.e., $l \in \{\text{syllables, words}\}$.

B. Example Emitter

Training and testing of the approaches is performed with data from simulations of an airborne multifunction radar. It is implemented with three different resource management methods such that the emissions differ in their order and agility, but the waveform parameters highly overlap. The first resource management method employs a quality-of-service (QoS) approach, which uses the expected utility of the tasks to assign the radar resources, e.g., time. The other two approaches are rule-based. One employs a set of simple rules, while the other is more sophisticated. Since the resource management method influences the emissions a lot, the radar can be regarded as three emitters with the same language but different grammars. In the following, they are abbreviated with QoS, Rules-v1, and Rules-v2. The QoS radar uses 25 380 syllables and 26 653 words, the Rules-v1 radar uses 103 syllables and 21 words, and the Rules-v2 radar emits 27 786 syllables and 34 440 words. More details are given in [15], [28], and [29].

C. Generation of Known Unknown Emitters

This section describes the generation of data in $\mathbb{V}$. For simplicity, the unknowns are also called "emitters" although the data might be artificially generated.

The signal of an unknown emitter either contains only unknown symbols, known and unknown symbols, or only known symbols. In the first case, the input sequence to the open-set recognition method always consists of UNK

symbols introduced by the word embedding and is, hence, known in advance. So, this case falls in $\mathbb{V}$ and can be included in the training. If a completely unknown signal is received, the symbol extraction step cannot connect it to any of the symbols in the global dictionary Ω^l and needs to assign new symbols to it. As these are not part of Ω^l, no vector representations exist, and the word embedding maps them to UNK. Hence, a signal that does not contain any known symbols is always represented as a sequence consisting of UNK symbols only, which is referred to as "UNKs" in the following. Of course, recognizing a sequence of UNKs as unknown does not require a sophisticated method, but it is included for completeness.

Based on the data from the classes in $\mathbb{V}$, the classifier is supposed to learn to distinguish between known and unknown input. If the unknown input is similar to $\mathbb{K}$, this task is more complicated. In the application considered in this article, the classifier should also be able to tell that an emitter is unknown if it uses the same symbols as the known classes but with different frequencies and agility. Hence, emitters in $\mathbb{V}$ that make use of known symbols are needed. The simplest way is to generate random sequences of known symbols. This approach is referred to as "Random." To make the data in $\mathbb{V}$ even more similar to $\mathbb{K}$, altered versions of the QoS and the Rules-v1 radar are created. The data of the Rules-v2 radar are not used here since it is defined to belong to $\mathbb{U}$ later on. The altered versions are called "QoS_{alt}" and "$\text{Rules-v1}_{\text{alt}}$", respectively. The data are created from the MC emitter models that are developed in [16] for predicting an emitter's next symbol. The symbol transition matrices are modified, and sequences are sampled from the new distribution $\hat{P}_e^{\text{alt}}(\omega_j|\omega_i)$ with $\omega_i, \omega_j \in \Omega_e^l$ being symbols in the dictionary of emitter e. Each row of the transition matrix is altered by randomly choosing one of three different operations. The first one reverses the order of the n most probable entries, while n is chosen uniformly at random in $[2, 3, \ldots, \lfloor \frac{1}{2}|\Omega_e^l| \rfloor]$. For example, with $n = 2$, the second most probable next symbol is assigned the probability of the most probable symbol and *vice versa*. With $n = 3$, reversing the order of the entries equals swapping the first and the third entry. The second operation reweights the entries by adding random values between 0 and N, followed by a normalization of the row. Due to the normalization, the choice of N is irrelevant as long as $N > 0$. The third operation leaves the row unchanged. The transition matrices are modified 16 times per radar. Afterward, symbols are sampled from the altered distributions to create data in $\mathbb{V}$.

D. Training Cases

Table I gives an overview of the different training cases defined for the classifiers considered in this article. Cases 0 and I contain all three emitters, as introduced in Section II-B, as $\mathbb{K}$, while case 0 corresponds to the conventional training and case I contains additional data in $\mathbb{V}$, as described in Section II-C. Cases II–V do not include the Rules-v2 radar in $\mathbb{K}$ because it is used in $\mathbb{U}$ for testing later on. Hence, the dictionary used by the open-set recognition methods does not contain the symbols exclusively emitted by the Rules-v2 radar in these cases, which are about 12% of its syllables and 38% of its words. Several combinations of emitters in $\mathbb{K}$ and $\mathbb{V}$ are defined in the different training cases.

TABLE I
Training Cases

		Training Case					
	Emitter	0	I	II	III	IV	V
known	QoS	x	x	x	x	x	x
known	Rules-v1	x	x	x	x	x	x
known	Rules-v2	x	x				
known unknown	UNKs		x	x		x	x
known unknown	Random		x	x		x	x
known unknown	QoS_{alt}				x	x	x
known unknown	Rules-v1_{alt}				x	x	

TABLE II
Overview of the Methods Employed for Unknown Emitter Recognition

Method	Case	Dictionary	Thres.	Classes
LSTM - Cross-Entropy	0	Ω^l	yes	QoS, Rv1, Rv2
	I	Ω^l	no	QoS, Rv1, Rv2, unk
	II-V	$\Omega^l_{QoS} \cup \Omega^l_{Rv1}$	no	QoS, Rv1, unk
LSTM - Entropic Open Set	I	Ω^l	yes	QoS, Rv1, Rv2
	II-V	$\Omega^l_{QoS} \cup \Omega^l_{Rv1}$	yes	QoS, Rv1
LSTM - Deep Open Class.	I	Ω^l	yes	QoS, Rv1, Rv2
	II-V	$\Omega^l_{QoS} \cup \Omega^l_{Rv1}$	yes	QoS, Rv1
LSTM - Unknown Gate	I	Ω^l	no	knwn, unk
	II-V	$\Omega^l_{QoS} \cup \Omega^l_{Rv1}$	no	knwn, unk
MC	0	Ω^l	yes	QoS, Rv1, Rv2
	I	Ω^l	no	QoS, Rv1, Rv2, unk
	II-V	$\Omega^l_{QoS} \cup \Omega^l_{Rv1}$	no	QoS, Rv1, unk
MC - Unknown Gate	I	Ω^l	no	knwn, unk
	II-V	$\Omega^l_{QoS} \cup \Omega^l_{Rv1}$	no	knwn, unk

Rv1 = Rules-v1, Rv2 = Rules-v2.

III. IMPLEMENTATION OF THE RECOGNITION METHODS

As described in the introduction, two alternative approaches can be applied for open-set recognition, which are employing a threshold or adding an extra unknown class. In this article, both approaches are analyzed, as well as a classifier that only distinguishes between known and unknown, i.e., an anomaly detector. It is not a method for open-set recognition, but it can be turned into one in combination with a classification method. Table II gives an overview of the methods. The implementation uses Python and TensorFlow. Training and evaluation are performed on a computer with two Intel Xeon E5-2637 v2 CPUs and two GPUs (NVIDIA GeForce RTX 2080 Ti and GTX TITAN), as well as 130 GB RAM. The resources allowed to run several training and evaluation sessions in parallel.

The general architectures of the LSTM-based methods are shown in Fig. 3, and the authors' previous work [2]

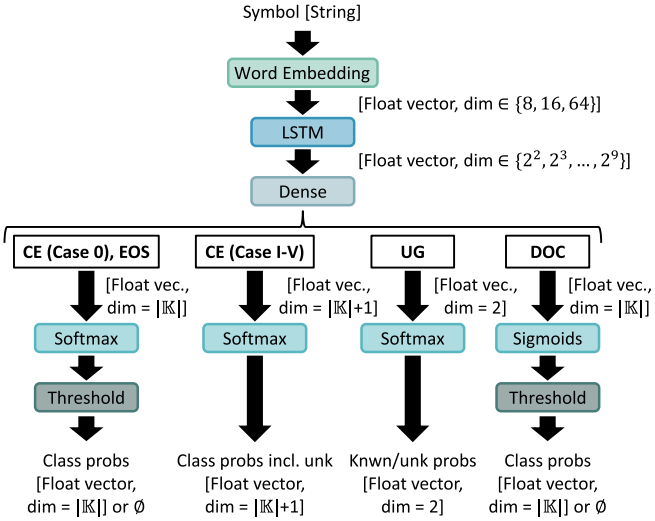

Fig. 3. Architectures of the different LSTM-based approaches.

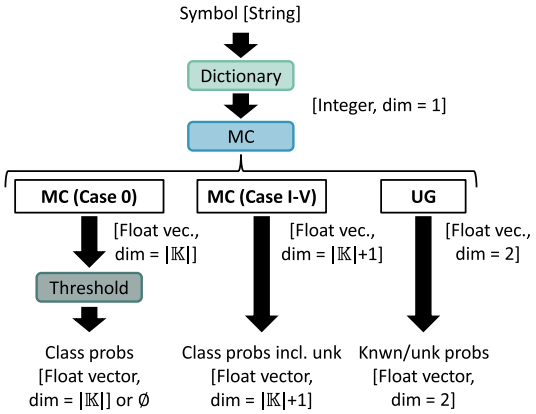

Fig. 4. Architectures of the MC variants.

provides the details. Fig. 4 visualizes the different MC variants. In contrast to the LSTM-based approaches, the MCs use the index of the symbol in the dictionary Ω^l instead of the embedding vector. Therefore, the symbols are mapped to an integer.

For the known emitters, this article uses the same data as [2], [15], and [16]. It contains eight different scenarios, which include situations with raids of hostile aircraft, missiles, and jamming. Each scenario consists of 300 Monte Carlo runs per emitter type, i.e., 7200 runs in total. Six of the eight scenarios are used for training, and the validation set, which is only used for the LSTMs, contains two scenarios. Testing uses all eight scenarios with 480 runs per emitter type.

The LSTMs are trained with a batch size of

$$\left\lfloor \frac{120}{|\mathbb{K} \cup \mathbb{V}|} \right\rfloor \cdot |\mathbb{K} \cup \mathbb{V}| \tag{3}$$

where each batch is split into chunks of size $|\mathbb{K} \cup \mathbb{V}|$ such that several emitters are represented in each batch, and on average, the same amount of data for each emitter is used during training. A basic batch size of 120 is chosen because it is a common factor of the number of simulation runs in the training, validation, and test set. By training several networks with different parameters, the best number of layers and number of LSTM cells per layer are found. During training, checkpoints are created when the current validation loss is lower than all previous values. The final network is then the checkpoint with the lowest loss on the validation set. Table III provides the parameters for each LSTM-based method. For all methods, also the MC-based approaches, the dictionary only contains the symbols of $\mathbb{K}$. This includes the symbols of $\mathbb{V}$ as the data are generated from the known emitters.

TABLE III
Parameters of the LSTM Architectures

	Symbol	Parameter	Training Case 0	I	II	III	IV	V
LSTM-CE	Syllables	# layers	1	1	1	1	2	1
		# cells/layer	8	16	16	8	32	16
	Words	# layers	1	1	1	1	1	1
		# cells/layer	4	8	8	16	16	8
LSTM-EOS	Syllables	# layers	–	1	1	1	1	2
		# cells/layer	–	16	16	16	16	16
	Words	# layers	–	1	1	1	1	1
		# cells/layer	–	8	8	32	8	8
LSTM-DOC	Syllables	# layers	–	2	1	2	1	1
		# cells/layer	–	64	64	16	64	128
	Words	# layers	–	1	1	1	1	1
		# cells/layer	–	256	256	512	512	512
LSTM-UG	Syllables	# layers	–	1	1	1	1	1
		# cells/layer	–	16	16	8	16	16
	Words	# layers	–	1	1	1	1	1
		# cells/layer	–	4	8	16	32	8

A. Long Short-Term Memory With Cross-Entropy Loss

The CE describes the difference between two probability distributions P and Q as

$$\mathbb{H}(P, Q) = -\mathbb{E}_{x \sim P} \log Q(x) \tag{4}$$

where $\mathbb{E}$ is the expected value. In information theory, the CE corresponds to the average number of bits required to encode an event of Q with a code that is optimal for P. The more similar P and Q are, the closer the required length is to the optimal code for P. In the context of neural networks, P describes the true probability distribution that is to be learnt and Q the distribution estimated by the network. During training, the distributions should become more similar, and the code length approaches the optimum.

The LSTM-CE is considered in two variants, which either employs a threshold δ for rejection or uses an extra unknown class. The variant with a threshold equals the network used for classification in [2] and corresponds to training case 0. The variant employing an extra unknown class is described by the training cases I–V. It is also trained with the CE loss, while the data from $\mathbb{V}$ are labeled to belong to the unknown class. Therefore, the training data consist of pairs $(\omega, \text{idx}(e))$, where $\omega \in \Omega^l$ is a symbol at modeling level l and $\text{idx}(e)$ is the class index c of the emitter $e \in \mathbb{K} \cup \mathbb{V}$. Here, $\text{idx}(\text{QoS}) = 0$ and $\text{idx}(\text{Rules-v1}) = 1$.

In training cases 0 and I, idx(Rules-v2) = 2 and idx(e) = 3 $\forall e \in \mathbb{V}$. In the other training cases, idx(e) = 2 $\forall e \in \mathbb{V}$.

The work [2] presents the architectures and the training procedure with different sequence lengths for the LSTMs in the training case 0. The sequence length is the number of symbols from the same emitter received in a row, e.g., the number of consecutive blue symbols in Fig. 2. For syllables, the LSTM trained on the complete scenarios is shown to provide the best results and is, hence, used here. For words, the LSTM trained with random sequence lengths achieves the highest accuracies with ideal data. However, it is not as robust with respect to corrupted data. It is still employed for words in this article as the conducted investigations serve as a feasibility study for recognizing a very similar emitter as unknown and, therefore, require high accuracies with ideal data.

B. Long Short-Term Memory with Entropic Open-Set Loss

The EOS loss function [13] forces the classifier to uniformly distribute the class probabilities for unknown input such that it is possible to define a threshold on the confidence for rejection:

$$J_E(\boldsymbol{x}) = \begin{cases} -\log \hat{y}_c, & \text{if } \text{class}(\boldsymbol{x}) = k_c \in \mathbb{K} \\ -\frac{1}{|\mathbb{K}|} \sum_{k_c \in \mathbb{K}} \log \hat{y}_c, & \text{if } \text{class}(\boldsymbol{x}) \in \mathbb{V}. \end{cases} \tag{5}$$

Here, $\hat{y}_c$ is the estimated probability value obtained by the softmax function for class $k_c \in \mathbb{K}$. For an input $\boldsymbol{x}$ with $\text{class}(\boldsymbol{x}) = k_c \in \mathbb{K}$, the EOS loss is equal to the CE loss. Hence, it is not trained with training case 0.

C. Long Short-Term Memory with Deep Open Classification Loss

Usually, the logits $\tilde{\boldsymbol{y}}$ of a neural network are normalized by the softmax function such that the output can be interpreted as a probability distribution. Like this, the network is forced to output a probability > 0 for at least one class. However, if the input does not belong to $\mathbb{K}$, the correct output might be a probability of 0 for all the known classes. Hence, Shu *et al.* [11] replace the softmax by a layer of independent sigmoids $\sigma(\tilde{y}_c)$ $\forall c \in \{0, 1, \ldots, C-1\}$ with

$$\sigma(x) = \frac{1}{1 + \exp(-x)}. \tag{6}$$

The probability assigned to each known class k_c is considered individually, and therefore, the probabilities of all classes do not need to sum to 1. The corresponding loss function J_{DOC} for training the network is defined by

$$J_{\text{DOC}}(\boldsymbol{x}) = \sum_{k_c \in \mathbb{K}} -\mathbb{1}_{k_c}(\text{class}(\boldsymbol{x})) \log(\sigma(\tilde{y}_c)) - (1 - \mathbb{1}_{k_c}(\text{class}(\boldsymbol{x}))) \log(1 - \sigma(\tilde{y}_c)) \tag{7}$$

where $\mathbb{1}_a(x)$ is a variation of the indicator function to determine equality instead of set membership

$$\mathbb{1}_a(x) = \begin{cases} 1, & \text{if } x = a \\ 0, & \text{otherwise.} \end{cases} \tag{8}$$

This loss function corresponds to the binary CE loss with the result of $\mathbb{1}_{k_c}(\text{class}(\boldsymbol{x}))$ as the label. For an input $\boldsymbol{x}$ with $\text{class}(\boldsymbol{x}) \in \mathbb{V}$, the desired output of the network consists of all 0. To reject an input, a threshold is needed.

Unfortunately, the LSTMs are hard to train using the DOC loss, as seen in the evaluation below. The best validation loss for training case I with syllables was achieved directly after random initialization of the weights and diverged afterward. No architecture could be found that solved the problem. Also, the networks are much larger than with the other loss types. The LSTMs with DOC loss are not trained with case 0 since it is especially designed to handle unknown input, and the experience during the training suggests that it would not provide better results than the LSTM-CE.

D. Long Short-Term Memory as Unknown Gate

Adding an extra output class might decrease the accuracy for $\mathbb{K}$. Therefore, a classifier that only distinguishes between known and unknown is trained. It can act as a gate that only passes the known data to the classifier that distinguishes between different known classes. The LSTM-based unknown gate (UG) is trained with the CE loss, while the training data consist of pairs $(\omega, \text{idx}(e))$ with idx(e) = 0 if $e \in \mathbb{K}$ and idx(e) = 1 if $e \in \mathbb{V}$. It is only trained with cases I–V since case 0 does not contain any classes in $\mathbb{V}$, and hence, idx(e) = 0 for the complete training data.

E. Markov Chain

For every known emitter $e \in \mathbb{K}$, the transition matrix $\hat{P}_e(\omega_j|\omega_i)$ that describes the probability that symbol ω_j is the next symbol having observed ω_i is estimated. Based on this matrix, the probability that the emitter generated the input sequence $\overline{\omega} = \omega_1\omega_2 \ldots \omega_{|\overline{\omega}|}$ is determined by Bayes' rule, i.e.,

$$\hat{P}(e|\overline{\omega}) = \frac{\hat{P}(\overline{\omega}|e) \cdot \hat{P}(e)}{\hat{P}(\overline{\omega})}. \tag{9}$$

This principle of the MC-based approach is the same as described in [16], and the MC trained with case 0 is equal to the approach from [16]. In this case, a threshold δ is applied to reject an input. For the other training cases, the MCs are extended to contain an unknown class, like for the LSTM-CE. The probability of the unknown class is also obtained by Bayes' rule

$$\hat{P}(\emptyset|\overline{\omega}) = \frac{\hat{P}(\overline{\omega}|\emptyset) \cdot \hat{P}(\emptyset)}{\hat{P}(\overline{\omega})}. \tag{10}$$

Here, $\hat{P}(\overline{\omega}|\emptyset)$ needs to be estimated based on the data in $\mathbb{V}$. A common symbol transition matrix is learnt for $\mathbb{V}$ because calculating $\sum_{e \in \mathbb{V}} \hat{P}(\overline{\omega}|e)$ to obtain $\hat{P}(\overline{\omega}|\emptyset)$ would result in a probability of 0 for most of the emitters in $\mathbb{U}$ since they did not actually generate the data. With a common transition

TABLE IV
Test Cases for the Different Classifiers

(a) Trained with cases 0 or I.

	Emitter	Test Case 1a	2a	3a	4a	5a
known	QoS	x	x	x	x	x
	Rules-v1	x	x	x	x	x
	Rules-v2	x	x	x	x	x
knwn unk	UNKs	x		x	x	x
unknown unknown	Unk-1			x		x
	Unk-2			x		x
	Unk-3				x	x
	Unk-4				x	x

(b) Trained with cases II to V.

	Emitter	Test Case 1b	2b	3b	4b	5b
known	QoS	x	x	x	x	x
	Rules-v1	x	x	x	x	x
knwn unk	UNKs	x		x	x	x
unknown unknown	Rules-v2	x	x	x	x	x
	Unk-1			x		x
	Unk-2			x		x
	Unk-3				x	x
	Unk-4				x	x

matrix, the data of $\mathbb{U}$ do not need to specifically match one of the emitters in $\mathbb{V}$ but only the common distribution, which is what is desired. The different training cases, hence, influence the estimated unknown distribution. The probabilities $\hat{P}(\emptyset)$ and $\hat{P}(e)$ for $e \in \mathbb{K}$ are set to $\frac{1}{|\mathbb{K}|+1}$. Note that obtaining $\hat{P}(\emptyset|\overline{\omega})$ by $1 - \sum_{e\in\mathbb{K}} \hat{P}(e|\overline{\omega})$ without estimating $\hat{P}(\overline{\omega}|\emptyset)$ is not possible since $\hat{P}(\overline{\omega})$ is computed by marginalization

$$\hat{P}(\overline{\omega}) = \hat{P}(\overline{\omega}|\emptyset) + \sum_{e\in\mathbb{K}} \hat{P}(\overline{\omega}|e). \tag{11}$$

F. Markov Chain as Unknown Gate

Analogous to the LSTM employed as an UG, an MC is learnt. The probability for the unknown class is determined as described above, while the probability for the known class is calculated by summing the probabilities of the known emitters

$$\hat{P}(\text{known}|\overline{\omega}) = \sum_{e\in\mathbb{K}} \hat{P}(e|\overline{\omega}). \tag{12}$$

Therefore, the MC is basically identical to the one described in the section before, except for this summation.

IV. EXPERIMENTAL RESULTS

The classifiers are tested with several combinations of emitters in $\mathbb{K}$, $\mathbb{V}$, and $\mathbb{U}$. The test cases 1a–5a for the classifiers trained with cases 0 and I are displayed in Table IV a, and the test cases 1b–5b for the other classifiers are given in Table IV b. The difference is in the role of the Rules-v2 radar, which belongs to $\mathbb{K}$ in the test cases "a" and to $\mathbb{U}$ in the test cases "b". Test case 2a corresponds to the evaluation performed in [2] and [16] and is included to see whether training with an extra unknown class decreases the performance in a scenario without unknown emitters. The only known unknown emitter contained in the test cases is the sequence consisting only of the UNK symbol since this is expected to actually appear during deployment of the system. Four additional emitters in $\mathbb{U}$ are used, which are artificially generated. They emit known and unknown syllables and words. Emitters that use more than one symbol randomly switch between them. The emitters are the following:

1) Unk-1: one known word with one known syllable;
2) Unk-2: one known word with eight known syllables;
3) Unk-3: two known words with one/three known syllables and one unknown word with one unknown syllable;
4) Unk-4: one known word with one known syllable and two unknown words with three/five known syllables.

Note that the Unk-1 emitter makes use of the most common word in the data of the known radars and, hence, their syllables.

The methods that need a threshold for accepting the current input (see Table II) are evaluated with the values $\delta \in \{0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0\}$. The minimum is chosen to be 0.4 as with three known emitters, a lower confidence value comes close to random guessing. For the methods with an extra unknown class in the output, the threshold can be interpreted being equal to 0.0. Therefore, each approach exists in several configurations, which consist of a training case and an acceptance threshold. The methods are tested at different sequence lengths, i.e., the number of consecutive symbols from the same emitter, in the set $\mathcal{S} = \{1, 10, 50, 100, 200, 400, 600, 800, 1000, 1200, 1400\}$. After the specified number of symbols has been processed by the classifier, data from a different emitter are presented at the input. Two different evaluation metrics are employed, which are the *distinction* and the *identification accuracy*. The distinction accuracy acc_{dist} is defined as the mean accuracy of distinguishing between known and unknown, while ignoring confusions between known emitters. It is composed of two parts, which are the *acceptance accuracy* acc_{acpt} and the *rejection accuracy* acc_{rej}. These use the top k accuracy $\text{acc}(\hat{Y}, Y, k)$, which gives the percentage of cases that the true class is among the top k outputs of the network sorted by probability. Here, $\hat{Y}$ is the output of the network and Y are the labels. The accuracies are given by

$$\text{acc}_{\text{acpt}}(s) = \frac{1}{|\mathbb{K}|} \sum_{e_i\in\mathbb{K}} \sum_{e_j\in\mathbb{K}} \text{acc}(\hat{Y}_s^{e_i}, Y_{e_j}, k), \quad k = 1 \tag{13}$$

$$\text{acc}_{\text{rej}}(s) = \frac{1}{|\mathbb{V} \cup \mathbb{U}|} \sum_{e\in\mathbb{V}\cup\mathbb{U}} \text{acc}(\hat{Y}_s^{e}, Y_{\text{unk}}, k), \quad k = 1 \tag{14}$$

where $\hat{Y}_s^e$ corresponds to the output of the classifier for the data of emitter e after a sequence of s symbols and Y_e is the set of labels containing only $\text{idx}(e)$, while Y_{unk} is the set of labels only containing the index of the unknown class. The distinction accuracy is then defined as

$$\text{acc}_{\text{dist}}(s) = \frac{1}{2}\left(\text{acc}_{\text{acpt}}(s) + \text{acc}_{\text{rej}}(s)\right). \tag{15}$$

The identification accuracy acc_{id} is given by the mean classification accuracy for the emitters in $\mathbb{K}$

$$\text{acc}_{\text{id}}(s) = \frac{1}{|\mathbb{K}|} \sum_{e\in\mathbb{K}} \text{acc}(\hat{Y}_s^{e}, Y_e, k), \quad k = 1. \tag{16}$$

TABLE V
Configurations (Training Case, δ) With the Best Distinction Accuracies, Averaged Over the Test Cases "a" or "b", Respectively

		Syllable Sequence length			Word Sequence length		
	Method	1	600	1400	1	600	1400
Test cases 'a'	LSTM-CE	(I, 0.0)	(0, 0.5)	(0, 0.5)	(I, 0.0)	(I, 0.0)	(I, 0.0)
	LSTM-EOS	(I, 0.5)	(I, 0.5)	(I, 0.5)	(I, 0.5)	(I, 0.5)	(I, 0.5)
	LSTM-DOC	(I, 0.4)	(I, 0.4)	(I, 0.4)	(I, 0.5)	(I, 0.5)	(I, 0.4)
	LSTM-UG	(I, 0.0)	(I, 0.0)	(I, 0.0)	(I, 0.0)	(I, 0.0)	(I, 0.0)
	MC	(0, 0.4)	(0, 0.4)	(0, 0.5)	(0, 0.4)	(0, 0.5)	(0, 0.5)
	MC-UG	(I, 0.0)	(I, 0.0)	(I, 0.0)	(I, 0.0)	(I, 0.0)	(I, 0.0)
Test cases 'b'	LSTM-CE	(IV, 0.0)	(IV, 0.0)	(IV, 0.0)	(II, 0.0)	(III, 0.0)	(III, 0.0)
	LSTM-EOS	(IV, 0.6)	(IV, 0.8)	(IV, 0.7)	(II, 0.6)	(IV, 0.8)	(IV, 0.8)
	LSTM-DOC	(IV, 0.4)	(IV, 0.4)	(IV, 0.4)	(II, 0.4)	(IV, 0.6)	(II, 0.5)
	LSTM-UG	(II, 0.0)	(IV, 0.0)	(IV, 0.0)	(V, 0.0)	(V, 0.0)	(V, 0.0)
	MC	(III, 0.0)	(IV, 0.0)	(IV, 0.0)	(II, 0.0)	(IV, 0.0)	(IV, 0.0)
	MC-UG	(III, 0.0)	(IV, 0.0)	(IV, 0.0)	(II, 0.0)	(IV, 0.0)	(IV, 0.0)

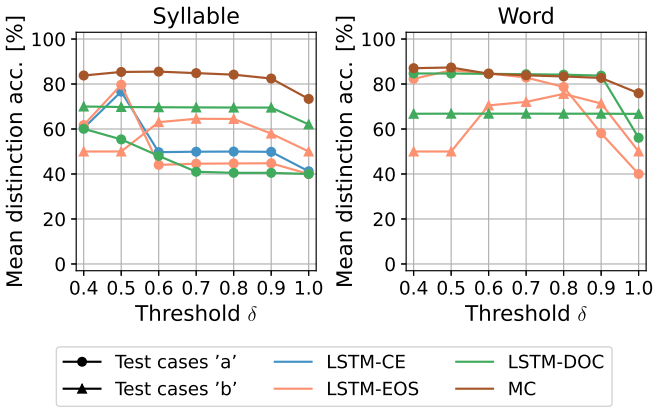

Fig. 5. Dependence of the distinction accuracy on the threshold at a sequence length of 1400 symbols.

Both metrics are considered because in several test cases, there are more unknown emitters than known ones, resulting in a high overall accuracy by rejecting every input.

A. Results for the Distinction Accuracy

Table V provides an overview of each method's configurations that achieve the highest mean distinction accuracy. These are obtained by averaging the accuracies for each configuration over the test cases 1a–5a and 1b–5b, respectively, and choosing the best configuration. Sequence lengths of 1, 600, and 1400 symbols are exemplarily shown. For most methods using a threshold, a value of $\delta \geq 0.5$ is beneficial to distinguish between known and unknown (see Fig. 5). As seen in Table V, the majority of the methods achieve the best results using the same configuration with 600 and 1400 symbols. Only in two cases, the best configurations differ for these lengths, both for syllables and words. For syllables, the best configuration for a single symbol differs from the configuration with 600 symbols in five out of 12 cases. For words, it is different in six scenarios. The LSTM-CE and the MC are the only methods trained with case 0. For both, the best results in the test cases "a" are most often obtained with training case 0 when using

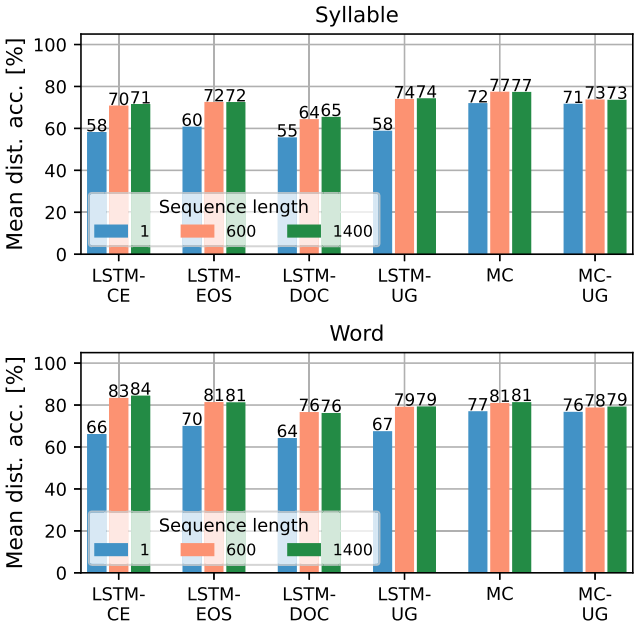

Fig. 6. Mean distinction accuracies of the best configurations.

syllables. For words, however, the training case I works best for the LSTM-CE but not for the MC.

For the test cases "b", the highest distinction accuracies with syllables are most often achieved with training case IV. This shows that to distinguish between known and unknown, training with all known unknown data provides an advantage. The results are not as clear for words, but four of the six methods provide the best results with training case IV at a sequence length of 600 words and three methods at 1400 words. The LSTM-CE achieves the highest accuracies with training case III for words, which does not contain the UNKs or random sequences. Still, it rejects the UNKs sequence (see Fig. 17). The LSTM-EOS applies high thresholds of up to 0.8. The relation between threshold and distinction accuracy is shown in Fig. 5 for 1400 symbols. Only the results of the best configurations that employ a threshold are depicted for the corresponding training case. As is seen, the accuracy variation with respect to the threshold depends on the method.

Fig. 6 presents the mean distinction accuracies of the best configurations, averaged over all test cases "a" and "b". As observed in [16], the MC outperforms all methods for syllables, especially with only one symbol. However, the LSTM-UG provides higher accuracies than the MC-UG with longer sequences. For words, the MC is most accurate with only one symbol, but the MC-UG comes close. With longer sequences, the LSTM-CE is the best method on average. The LSTM-EOS and the MC achieve about the same results, just as the LSTM-UG and the MC-UG. The LSTM-DOC slightly falls behind the other methods. Overall, the distinction accuracies are higher for words since the unknown emitters have less words than syllables in common with the known emitters, which shows the benefit of the hierarchical emission model.

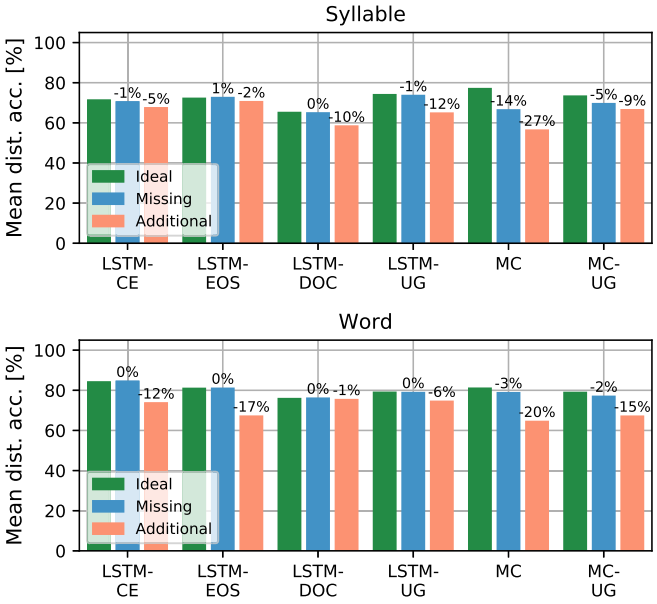

Fig. 7. Mean distinction accuracies of the best configurations with 20% missing or additional symbols, respectively, at a sequence length of 1400 symbols.

To provide a more realistic scenario, the methods are also tested with corrupted data. Note that they are trained with ideal data only. As an example, the evaluation on sequences of length 1400 is presented with 20% missing or additional symbols, which is the worst-case scenario in [2] and [16]. There is no need for a short reaction time in ELINT, hence, longer sequences can be processed before making a decision. The symbols are removed or inserted randomly, choosing additional symbols from the global dictionary Ω^I. Hence, also symbols of other emitters might be inserted.

Fig. 7 (top) shows the results of the best configurations for the distinction accuracy with missing or additional syllables in comparison to the ideal results. The labels of the bars are the accuracies relative to the ideal case

$$\mathrm{acc}^{\mathrm{rel}} = \frac{\mathrm{acc}^{\mathrm{corrupt}} - \mathrm{acc}^{\mathrm{ideal}}}{\mathrm{acc}^{\mathrm{ideal}}} \cdot 100\%. \qquad (17)$$

All LSTM-based methods are very robust with respect to missing syllables. The LSTM-CE and the LSTM-EOS only exhibit a small accuracy decrease with additional syllables. As also shown in [16], the MC is not robust against corrupted data. The MC-UG is much more robust than the MC but still outperformed by the LSTMs. Fig. 7 (bottom) depicts the distinction accuracies for words with corrupted data. All LSTM-based methods are trained with random sequence lengths as this variant works best with ideal data. However, it is not as robust with respect to additional words as the LSTM trained with the complete scenarios [2]. Missing words cause no accuracy loss for the LSTM-based methods and only a slight decrease for the MCs. With additional words, however, the accuracy decreases significantly. In this case, the LSTM-CE is outperformed by the LSTM-DOC and the LSTM-UG. With missing words, it still provides the best average performance.

TABLE VI
Configurations (Training Case, δ) With the Best Identification Accuracies, Averaged Over the Test Cases "a" or "b", Respectively

		Syllable Sequence length			Word Sequence length		
	Method	1	600	1400	1	600	1400
Test cases 'a'	LSTM-CE	(0, 0.4)	(0, 0.4)	(0, 0.4)	(0, 0.4)	(0, 0.4)	(0, 0.4)
	LSTM-EOS	(I, 0.4)	(I, 0.4)	(I, 0.4)	(I, 0.4)	(I, 0.4)	(I, 0.4)
	LSTM-DOC	(I, 0.4)	(I, 0.4)	(I, 0.4)	(I, 0.4)	(I, 0.4)	(I, 0.4)
	MC	(0, 0.6)	(I, 0.0)	(I, 0.0)	(0, 0.4)	(I, 0.0)	(I, 0.0)
Test cases 'b'	LSTM-CE	(III, 0.0)	(III, 0.0)	(III, 0.0)	(II, 0.0)	(II, 0.0)	(II, 0.0)
	LSTM-EOS	(II, 0.4)	(II, 0.4)	(II, 0.4)	(II, 0.4)	(IV, 0.4)	(IV, 0.4)
	LSTM-DOC	(II, 0.4)	(IV, 0.4)	(IV, 0.4)	(II, 0.4)	(II, 0.4)	(II, 0.4)
	MC	(II, 0.0)	(II, 0.0)	(II, 0.0)	(II, 0.0)	(II, 0.0)	(II, 0.0)

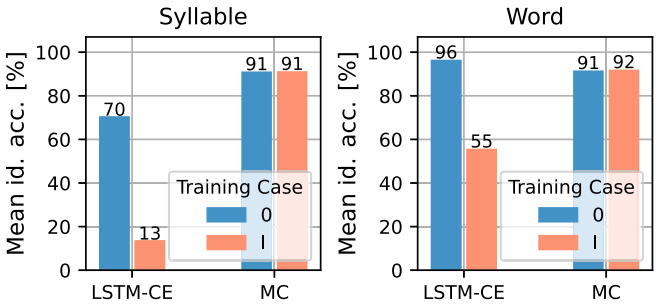

Fig. 8. Comparison of the mean identification accuracies in the test cases "a" with training case 0 or I at a sequence length of 1400 symbols.

B. Results for the Identification Accuracy

Table VI depicts the configurations with the best identification accuracies. In contrast to the distinction accuracy for syllables, the identification accuracy is highest with training cases that do not contain all known unknown data in the test cases "b" for most methods. Especially for the LSTM-CE, the conventional training (case 0) works best. A comparison between the training cases 0 and I is shown in Fig. 8, which are only employed in the test cases "a". Hence, the results in this figure are different from those presented in Fig. 10, which takes the test cases "b" into account as well.

With only one exception, all methods that apply a threshold use $\delta = 0.4$ for a high identification accuracy, which reduces the false rejection rate. Fig. 9 shows the dependence of the identification accuracy on the threshold with 1400 symbols. The training case that is part of the best configuration is depicted for each method that employs a threshold. Most of the methods reject every input using $\delta = 1.0$. Especially the LSTM-EOS, shows a distinctive behavior for syllables in the test cases "b". Both known radars are correctly identified with a confidence of at least 0.9, but they are rejected with $\delta = 1.0$.

The MC achieves much higher identification accuracies than all other methods with syllables (see Fig. 10). The difference is even bigger than for the distinction accuracy. For words, only the LSTM-CE achieves about the same accuracy as the MC with 1400 symbols. In the other cases, the MC significantly outperforms all other methods, especially with only one symbol. This confirms the results from [16].

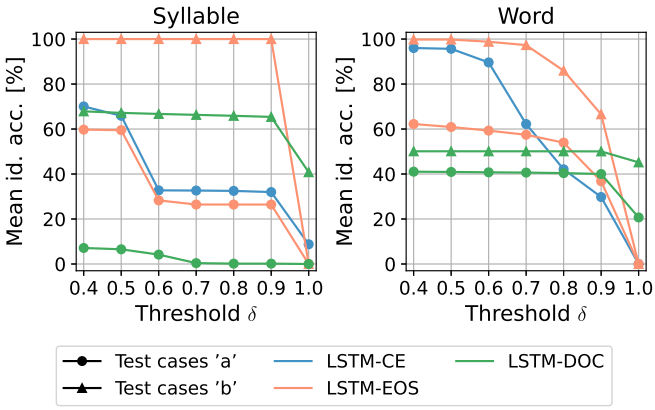

Fig. 9. Dependence of the identification accuracy on the threshold at a sequence length of 1400 symbols.

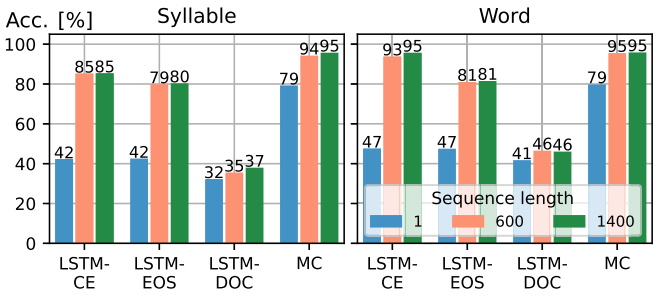

Fig. 10. Mean identification accuracies of the best configurations.

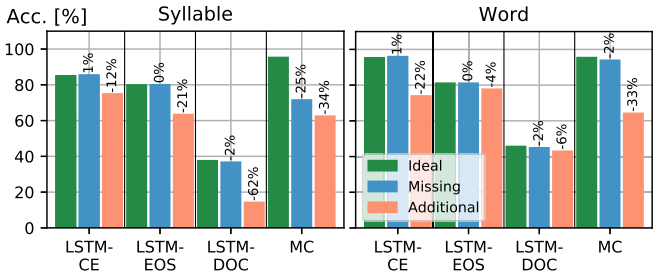

Fig. 11. Mean identification accuracies of the best configurations with 20% missing or additional symbols, respectively, at a sequence length of 1400 symbols.

Both clearly outperform the other two LSTM-based approaches. The LSTM-DOC is again by far the least accurate method.

Fig. 11 (left) displays the results for syllables of the best configurations for the identification accuracy with corrupted data. Also, in this case, missing syllables barely affect the LSTMs, but the accuracy of the MC decreases significantly, mainly because of a confusion between emitters in $\mathbb{K}$. With additional syllables, the MC rejects some of the known input and accepts about 25% of the UNKs in some cases. The LSTM-CE is the most robust method with higher average accuracies than the MC. The other LSTMs exhibit more decrease with additional syllables. Especially, the LSTM-DOC is not robust and sometimes rejects the complete input.

The results for words of the best configurations for the identification accuracy are shown in Fig. 11 (right). Again, missing words do not cause much harm, but with additional words the MC's accuracy decreases by 33%. This is due to a higher false rejection rate and since the emitters in $\mathbb{K}$ are more often confused. Also, the LSTM-CE's accuracy decreases significantly with additional words due to confusing the known emitters. The LSTM-EOS' configuration for the identification accuracy is much more robust than for the distinction accuracy, and it provides the best results with corrupted data, but its performance with ideal data is not competitive.

C. Results for the Rejection and Acceptance Accuracy

A high rejection accuracy might be accompanied by a low acceptance accuracy and *vice versa*. Figs. 12 and 13 depict these accuracies for each method and each test case with 1400 symbols using ideal data and 20% additional symbols. Test case 2a only contains known emitters, so to avoid a division by 0 in (14), its rejection accuracy is set to 100%.

Most methods provide an acceptance accuracy $\geq 90\%$ for syllables and $\geq 80\%$ for words with ideal data, while the rejection accuracy varies greatly between the test cases. The LSTM-DOC, however, tends to reject nearly every input for the test cases "a" with syllables and also has the lowest acceptance accuracy in the test cases "b". Still, it achieves much higher rejection accuracies for syllables. As some of the unknown emitters are very similar to $\mathbb{K}$, correctly rejecting the unknown emitters also results in a false rejection of some of the known sequences. The best configuration of the LSTM-EOS includes a high threshold of 0.8 in the test cases "b" for words, which leads to a lower acceptance accuracy. However, with ideal data, more than 80% of the input is accepted with a value ≥ 0.8. With additional symbols, on the other hand, the high threshold leads to a low acceptance accuracy.

Overall, the rejection results are better with words than with syllables. For most of the test cases and methods, the rejection accuracies are above 25% with words and ideal data, while the majority lies in the range 10–60% for syllables. With 20% additional symbols, the methods reject more sequences, leading to a higher rejection but a lower acceptance accuracy.

D. Rejection Accuracies for the Unknown Emitters

Fig. 14 shows the rejection accuracies with syllables for the Unk-1 to Unk-4 emitters in the test cases "a" and "b". The figure considers sequences of 1400 syllables and the best configurations for the distinction accuracy. An accuracy of 100% is not marked. As the LSTM-DOC classifies nearly every input as unknown in the cases "a", its rejection accuracy is high. For the Unk-3 and the Unk-4 emitter, the MC also achieves high accuracies but not for Unk-1 and Unk-2. Both only use known syllables, while Unk-3 and Unk-4 also emit unknown ones, aiding a rejection. The rejection accuracies are much higher in the test cases "b" than in the cases "a". Only the accuracies of the MC significantly decrease. The Rules-v2 emitter belongs to $\mathbb{U}$ here, which helps the LSTM-based methods to distinguish known from unknown.

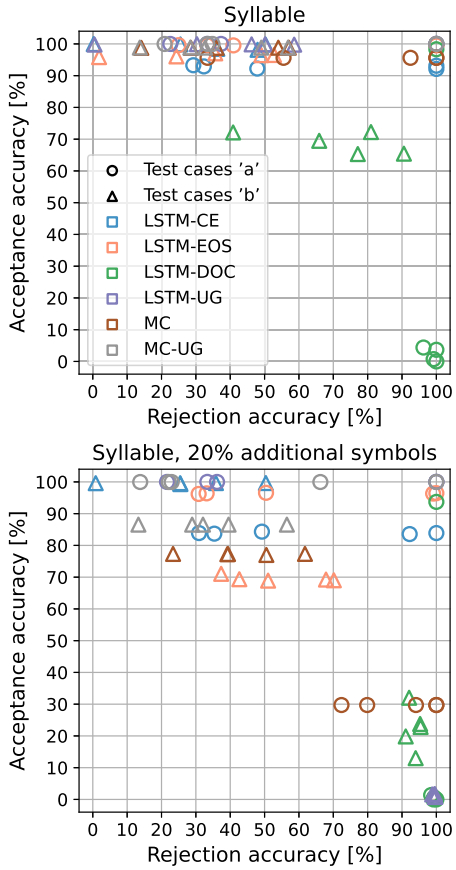

Fig. 12. Rejection and acceptance accuracies with 1400 syllables.

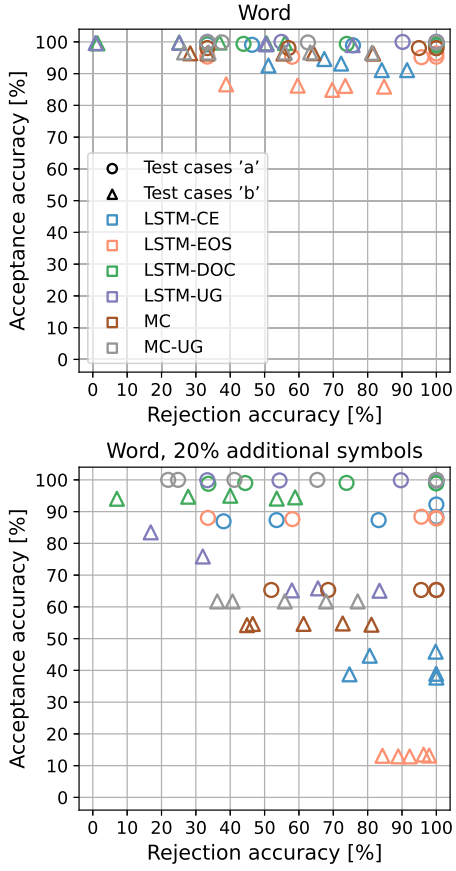

Fig. 13. Rejection and acceptance accuracies with 1400 words.

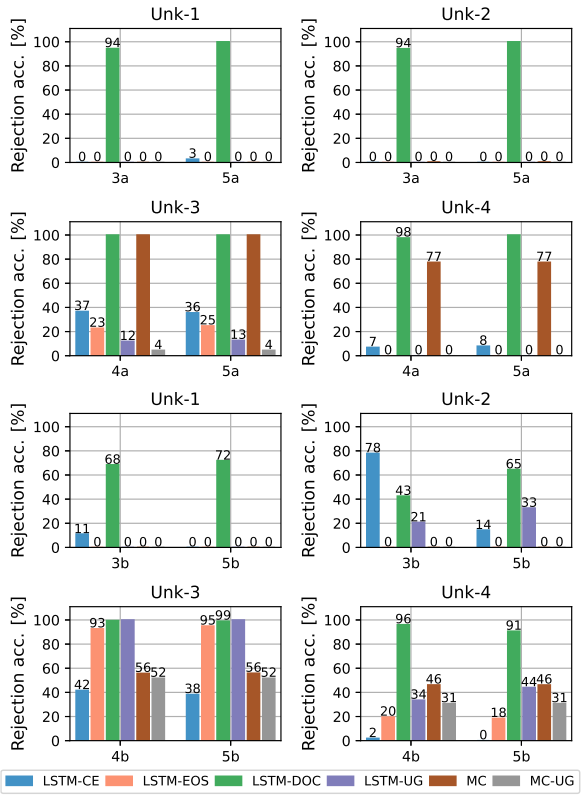

Fig. 14. Rejection accuracies for the Unk-1 to Unk-4 emitters with 1400 syllables.

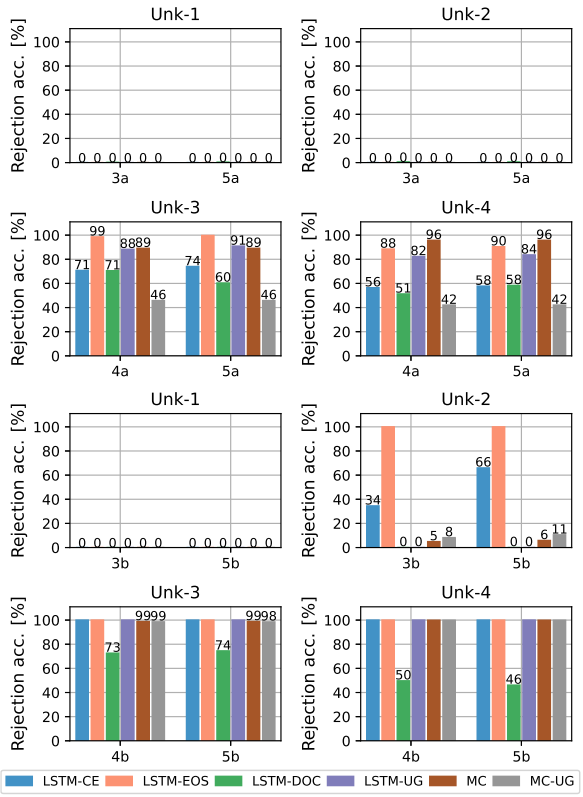

Fig. 15. Rejection accuracies for the Unk-1 to Unk-4 emitters with 1400 words.

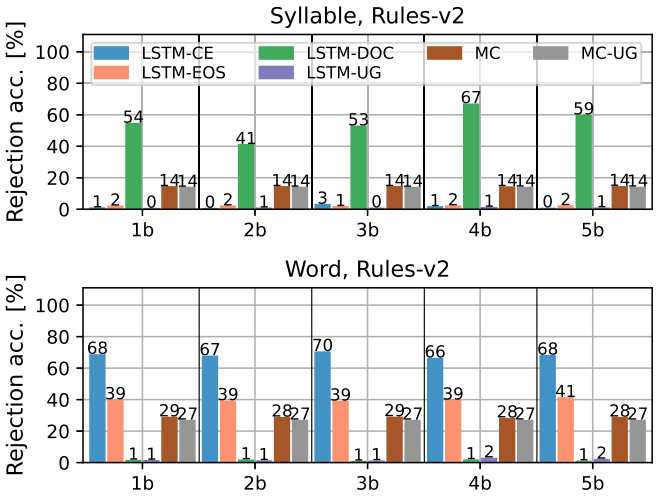

Fig. 16. Rejection accuracies for the Rules-v2 emitter with 1400 symbols.

Fig. 15 depicts the individual rejection accuracies with words of the Unk-1 to Unk-4 emitters per method. Also when using words, the Unk-1 and Unk-2 emitters are not rejected in the test cases "a". In the cases "b", the LSTM-EOS reliably rejects the Unk-2 emitter, and the LSTM-CE also achieves higher accuracies than in the test cases "a". All methods classify the Unk-3 and the Unk-4 emitter as unknown with more than 40% accuracy in all cases.

Fig. 16 compares the rejection accuracies for the Rules-v2 radar with syllables or words. As for the other radars in $\mathbb{U}$, words are much better suited for rejection than syllables. Only the LSTM-DOC achieves accuracies of more than 40% with syllables, while wrongly rejecting at least 25% of $\mathbb{K}$. Although the Rules-v2 radar is very similar to the known emitters, it is rejected with about 70% accuracy by the LSTM-CE with words, which is enough to recognize that there is something unknown in the intercepted data. With syllables, the best rejection accuracy is 3% for this method. One reason is that the Rules-v2 radar has less words than syllables in common with the emitters in $\mathbb{K}$, and hence, more words are mapped to UNK. This shows the benefits of the hierarchical structure of the emission model.

Overall, the results of the LSTMs for the emitters in $\mathbb{U}$ depend on the test case. In contrast, the performance of the MCs is stable except for small statistical deviations. As an MC does not have a memory, processing the data of other emitters in the scenario does not influence the output, while past sequences impact the LSTMs' internal state.

The only emitter belonging to $\mathbb{V}$ considered in the evaluation is the UNKs, because this kind of data is expected to actually appear when employing the system. All methods reject the UNKs sequences with at least 96% accuracy in all cases. Therefore, the results for ideal data are not shown, but, instead, the accuracies with 20% additional symbols are depicted in Fig. 17. Missing symbols do not cause a difference since the sequences consist of the UNK symbol only. All methods except for the MC-UG in the test cases "a" are able to reliably reject the UNKs emitter with 20% known symbols. Hence, if the symbol extraction step wrongly maps an unknown input to a known symbol, it does not change the result as long as there are sufficiently many UNK symbols.

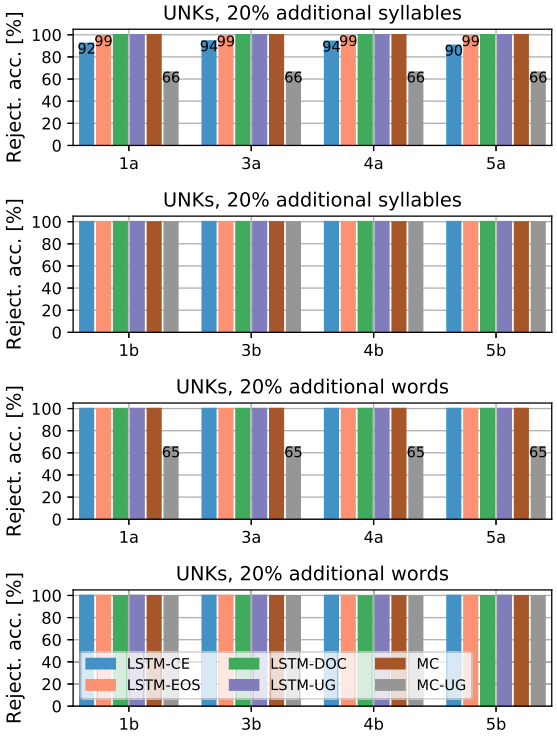

Fig. 17. Rejection accuracies for the UNKs emitter with 1400 symbols.

TABLE VII
Evaluation Matrices at a Sequence Length of 1400 Syllables

(a) LSTM-CE.

Config$_i$		Evaluation metric acc_{dist}	acc_{id}	mean
	acc_{dist}	83.96	73.84	78.90
	acc_{id}	64.41	95.15	79.78
	mean	82.30	94.07	88.19

(b) LSTM-EOS.

Config$_i$		Evaluation metric acc_{dist}	acc_{id}	mean
	acc_{dist}	80.85	73.42	77.14
	acc_{id}	66.15	81.01	73.58
	mean	79.07	79.11	79.09

(c) LSTM-DOC.

Config$_i$		Evaluation metric acc_{dist}	acc_{id}	mean
	acc_{dist}	75.75	45.57	60.66
	acc_{id}	75.73	45.57	60.65
	mean	75.75	45.57	60.66

(d) MC.

Config$_i$		Evaluation metric acc_{dist}	acc_{id}	mean
	acc_{dist}	80.92	93.55	87.24
	acc_{id}	77.03	95.36	86.19
	mean	80.92	93.55	87.24

The rows show the configuration that is best for the specified metric and the columns denote the metric used. Values in [%].

E. Impact of the Configuration

The results above represent the accuracies of the best configuration for the specific evaluation metric. Table VII provides "evaluation matrices" for syllables, whose rows represent the best configuration for the specified metric, while the columns denote the metric that it is evaluated with. For example, the row "$\mathrm{acc}_{\mathrm{dist}}$" in combination with the column "$\mathrm{acc}_{\mathrm{id}}$" provides the mean identification accuracy of the configuration that is best for the distinction accuracy. The values on the diagonal are the accuracies

TABLE VIII
Configurations (Training Case, δ) With the Best Mean of Distinction and Identification Accuracy, Averaged Over the Test Cases "a" or "b", Respectively

		Syllable Sequence length			Word Sequence length		
	Method	1	600	1400	1	600	1400
Test cases 'a'	LSTM-CE	(I, 0.0)	(0, 0.5)	(0, 0.5)	(I, 0.0)	(0, 0.5)	(0, 0.5)
	LSTM-EOS	(I, 0.5)	(I, 0.5)	(I, 0.5)	(I, 0.4)	(I, 0.5)	(I, 0.5)
	LSTM-DOC	(I, 0.4)	(I, 0.4)	(I, 0.4)	(I, 0.4)	(I, 0.4)	(I, 0.4)
	MC	(0, 0.4)	(0, 0.5)	(0, 0.5)	(0, 0.4)	(0, 0.5)	(0, 0.5)
Test cases 'b'	LSTM-CE	(II, 0.0)	(III, 0.0)	(III, 0.0)	(II, 0.0)	(III, 0.0)	(III, 0.0)
	LSTM-EOS	(IV, 0.6)	(II, 0.9)	(II, 0.9)	(II, 0.6)	(IV, 0.6)	(IV, 0.7)
	LSTM-DOC	(II, 0.4)	(IV, 0.4)	(IV, 0.4)	(II, 0.4)	(II, 0.9)	(II, 0.5)
	MC	(II, 0.0)	(IV, 0.0)	(IV, 0.0)	(II, 0.0)	(IV, 0.0)	(IV, 0.0)

TABLE IX
Evaluation Matrices at a Sequence Length of 1400 Words

(a) LSTM-CE.

Config.		Evaluation metric acc_{dist}	acc_{id}	mean
	acc_{dist}	83.96	73.84	78.90
	acc_{id}	64.41	95.15	79.78
	mean	82.30	94.07	88.19

(b) LSTM-EOS.

Config.		Evaluation metric acc_{dist}	acc_{id}	mean
	acc_{dist}	80.85	73.42	77.14
	acc_{id}	66.15	81.01	73.58
	mean	79.07	79.11	79.09

(c) LSTM-DOC.

Config.		Evaluation metric acc_{dist}	acc_{id}	mean
	acc_{dist}	75.75	45.57	60.66
	acc_{id}	75.73	45.57	60.65
	mean	75.75	45.57	60.66

(d) MC.

Config.		Evaluation metric acc_{dist}	acc_{id}	mean
	acc_{dist}	80.92	93.55	87.24
	acc_{id}	77.03	95.36	86.19
	mean	80.92	93.55	87.24

The rows show the configuration that is best for the specified metric and the columns denote the metric used. Values in [%].

of "matching" configuration and evaluation metric. The results are shown for sequences of 1400 syllables, averaged over all test cases "a" and "b". The label "mean" refers to the mean of the identification and the distinction accuracy. The corresponding best configurations are given in Table VIII. For the LSTM-DOC, the best configuration is always the same, and hence, the results do not change with another evaluation metric (see Table VII c). For the other LSTM-based methods, the distinction accuracy decreases especially with the configuration that is best for the identification accuracy. By selecting a configuration, a decision between true and false rejection rate is made. As ELINT is supposed to collect new information, it is better to reject some known input than to misclassify something unknown. However, if much of $\mathbb{K}$ is rejected, more manual analysis is required.

Table IX shows the evaluation matrices for words, exhibiting similar patterns. The best configuration for the identification accuracy provides much worse results for the distinction accuracy with the LSTM-CE and the LSTM-EOS. The best configuration for the average of both metrics seems to be a good compromise if no hierarchical combination of an UG and a classifier for $\mathbb{K}$ is desired. For the MC, the decrease of the distinction accuracy is smaller. Again, the LSTM-DOC obtains its best results for all metrics with the same configuration in most cases.

TABLE X
Comparison of the Methods With Respect to Accuracy, Robustness, and the Performance With Short Sequence Lengths

Method	Level	Accuracy $dist$	Accuracy id	Robustness $dist$	Robustness id	Short seqs $dist$	Short seqs id
LSTM-CE	Syllable	+	+	+	o	-	--
	Word	++	++	-	-	-	--
LSTM-EOS	Syllable	+	+	++	-	-	--
	Word	++	+	-	++	-	--
LSTM-DOC	Syllable	o	--	o	--	-	--
	Word	+	--	++	+	-	--
LSTM-UG	Syllable	+	n/a	o	n/a	-	n/a
	Word	+	n/a	+	n/a	-	n/a
MC	Syllable	++	++	--	--	+	++
	Word	++	++	-	--	+	++
MC-UG	Syllable	+	n/a	o	n/a	+	n/a
	Word	+	n/a	-	n/a	+	n/a

++ very high, + high, o satisfactory, - low, -- very low.

F. General Comparison of the Methods

Table X provides a more general comparison of the methods with respect to the accuracy, robustness, and the performance that is achieved with short sequence lengths. The LSTM-CE, the LSTM-EOS, and the MC work well in comparison to the other approaches. When selecting the most promising method for an actual application, the task and the quality of the data need to be considered. If the data are expected to be corrupted, the LSTM-CE or LSTM-EOS should be chosen. The decision for one of the two methods depends on whether the distinction between known and unknown or the identification of known emitters is more important. With ideal data and short sequences, i.e., if a fast decision is needed, the MC is the best choice.

V. CONCLUSION

This article investigates open-set recognition to detect unknown radar emitters in an electronic intelligence (ELINT) context. The presented methods are based on a hierarchical emission model, which interprets the radar emissions as a language that consists of letters, syllables, words, commands, and functions. We compare several variants of Long Short-Term Memory networks (LSTMs) and Markov chains (MCs), which differ in the training and output classes. The LSTMs are employed with the cross-entropy (CE), entropic open-set (EOS), and deep open classification (DOC) loss. All methods are tested with different combinations of known and unknown emitters, as well as ideal and corrupted data.

The challenge in open-set recognition is that no data for the "unknown unknown" classes exist, but the classifier needs to be trained with "known unknown" data, which

hopefully allow for the rejection of the unknown unknown input. We suggest approaches for the generation of known unknown data, which include sampling random sequences from the emitters' dictionaries or altering an MC-based emitter model.

All methods reject sequences consisting only of the special symbol UNK, even though not all of them are trained to do so. The LSTM-CE trained without known unknown data also rejects the UNKs with a confidence threshold of 0.5. Although "thresholding softmax" is claimed to be problematic [13], it works very well in this case. Hence, a completely unknown emitter can be reliably detected. This is even possible when the UNKs sequences contain 20% known symbols.

Unknown emitters that are more similar to the known ones are less reliably recognized, while higher accuracies are achieved with words. For four of the five unknown unknown emitters, the best rejection rates are above 65% with words, which is enough to tell that there are active unknown emitters. This shows the benefits of the hierarchical emission model.

With corrupted data, the performance naturally decreases. Missing symbols only have a small impact, and additional symbols can cause a very large accuracy decrease. Especially the MC is not robust, while most of the LSTM-based methods exhibit less accuracy decrease. Both for syllables and words, the LSTM outperforms the MC with corrupted data at longer sequence lengths, although the MC's accuracy in the ideal case is much higher for syllables and comparable for words.

ACKNOWLEDGMENT

Sabine Apfeld would like to thank Isabel Schlangen for the helpful discussions that improved this article.

REFERENCES

[1] R. Wiley
ELINT: The Interception and Analysis of Radar Signals. Norwood, MA, USA: Artech House, 2006.

[2] S. Apfeld, A. Charlish, and G. Ascheid
Identification of radar emitter type with recurrent neural networks
in *Proc. Sensor Signal Process. Defence Conf.*, 2020, pp. 1–5.

[3] L. Cain, J. Clark, E. Pauls, B. Ausdenmoore, R. Clouse, and T. Josue
Convolutional neural networks for radar emitter classification
in *Proc. IEEE Annu. Comput. Commun. Workshop Conf.*, 2018, pp. 79–83.

[4] S. A. Shapero, A. B. Dill, and B. O. Odelowo
Identifying agile waveforms with neural networks
in *Proc. Int. Conf. Inf. Fusion*, 2018, pp. 745–752.

[5] Z.-M. Liu and P. S. Yu
Classification, denoising, and deinterleaving of pulse streams with recurrent neural networks
IEEE Trans. Aerosp. Electron. Syst., vol. 55, no. 4, pp. 1624–1639, Aug. 2019.

[6] P. Notaro, M. Paschali, C. Hopke, D. Wittmann, and N. Navab
Radar emitter classification with attribute-specific recurrent neural networks
2019, *arXiv: 1911.07683*.

[7] Z. M. Liu
Recognition of multifunction radars via hierarchically mining and exploiting pulse group patterns
IEEE Trans. Aerosp. Electron. Syst., vol. 56, no. 6, pp. 4659–4672, Dec. 2020.

[8] W. J. Scheirer, A. de Rezende Rocha, A. Sapkota, and T. E. Boult
Toward open set recognition
IEEE Trans. Pattern Anal. Mach. Intell., vol. 35, no. 7, pp. 1757–1772, Jul. 2013.

[9] W. J. Scheirer, L. P. Jain, and T. E. Boult
Probability models for open set recognition
IEEE Trans. Pattern Anal. Mach. Intell., vol. 36, no. 11, pp. 2317–2324, Nov. 2014.

[10] A. Bendale and T. E. Boult
Towards open set deep networks
in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 1563–1572.

[11] L. Shu, H. Xu, and B. Liu
DOC: Deep open classification of text documents
in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2017, pp. 2911–2916.

[12] D. O. Cardoso, J. Gama, and F. M. França
Weightless neural networks for open set recognition
Machine Learn., vol. 106, no. 9-10, pp. 1547–1567, 2017.

[13] A. R. Dhamija, M. Günther, and T. E. Boult
Reducing network agnostophobia
in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2018, pp. 9157–9168.

[14] S. Hochreiter and J. Schmidhuber
Long short-term memory
Neural Comput., vol. 9, no. 8, pp. 1735–1780, 1997.

[15] S. Apfeld, A. Charlish, and G. Ascheid
Modelling, learning and prediction of complex radar emitter behaviour
in *Proc. IEEE Int. Conf. Mach. Learn. Appl.*, 2019, pp. 305–310.

[16] S. Apfeld, A. Charlish, and G. Ascheid
The value of memory: Markov chain versus long short-term memory for electronic intelligence
in *Proc. IEEE Radar Conf.*, 2021, pp. 1–6.

[17] I. Jo, J. Kim, H. Kang, Y.-D. Kim, and S. Choi
Open set recognition by regularising classifier with fake data generated by generative adversarial networks
in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2018, pp. 2686–2690.

[18] E. Granger, S. Grossberg, P. Lavoie, and M. A. Rubin
Comparison of classifiers for radar emitter type identification
Intell. Eng. Syst. Artif. Neural Netw., vol. 9, pp. 3–11, 1999.

[19] E. Granger, M. A. Rubin, S. Grossberg, and P. Lavoie
A what-and-where fusion neural network for recognition and tracking of multiple radar emitters
Neural Netw., vol. 14, no. 3, pp. 325–344, 2001.

[20] L. Anjaneyulu, N. Murthy, and N. Sarma
Radar emitter classification using self-organising neural network models
in *Proc. Int. Conf. Recent Adv. Microw. Theory Appl.*, 2008, pp. 431–433.

[21] P. Fitch
Pulse signal and source identification using fuzzy-neural techniques
IEEE Aerosp. Electron. Syst. Mag., vol. 28, no. 1, pp. 22–33, Jan. 2013.

[22] L. S. Kim, H. B. Bae, R. M. Kil, and C. H. Jo
Classification of the trained and untrained emitter types based on class probability output networks
Neurocomputing, vol. 248, pp. 67–75, Jul. 2017.

[23] W. J. Park and R. M. Kil
Pattern classification with class probability output network
IEEE Trans. Neural Netw., vol. 20, no. 10, pp. 1659–1673, Oct. 2009.

[24] N. A. Visnevski
Syntactic modeling of multi-function radars
Ph.D. dissertation, Dept. Elect. Comput. Eng., McMaster Univ., Hamilton, ON, Canada, 2005.

[25] N. A. Visnevski, V. Krishnamurthy, A. Wang, and S. Haykin
Syntactic modeling and signal processing of multifunction radars: A stochastic context-free grammar approach
Proc. IEEE, vol. 95, no. 5, pp. 1000–1025, May 2007.

[26] T. Mikolov, K. Chen, G. Corrado, and J. Dean
Efficient estimation of word representations in vector space
in *Proc. Int. Conf. Learn. Representations*, 2013.

[27] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean
Distributed representations of words and phrases and their compositionality
in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2013, pp. 3111–3119.

[28] A. Charlish and F. Katsilieris
Array radar resource management
in *Novel Radar Techniques and Applications Volume 1: Real Aperture Array Radar, Imaging Radar, and Passive and Multistatic Radar.* London, U.K.: Inst. Eng. Technol., 2017, pp. 135–171.

[29] A. Charlish and F. Hoffmann
Cognitive radar management
in *Novel Radar Techniques and Applications Volume 2: Waveform Diversity and Cognitive Radar, and Target Tracking and Data Fusion*. London, U.K.: Inst. Eng. Technol., 2017, pp. 157–193.

Sabine Apfeld received the M.Sc. degree in computer science from the University of Bonn, Bonn, Germany, in 2014.

In 2014, she joined the Department of Sensor Data and Information Fusion, Fraunhofer Institute for Communication, Information Processing and Ergonomics, Wachtberg, Germany. Her research interests include radar signal analysis, sensor management, and machine learning for electronic intelligence applications.

Alexander Charlish (Senior Member, IEEE) received the M.Eng. degree in electronic and computer engineering from the University of Nottingham, Nottingham, U.K., in 2006, and the Ph.D. degree in multifunction radar resources management from University College London, London, U.K., in 2011.

In 2011, he joined the Department of Sensor Data and Information Fusion, Fraunhofer Institute for Communication, Information Processing and Ergonomics, Wachtberg, Germany, where he currently leads the Sensor and Resources Management Group. In this role, he leads a group of scientists conducting research on intelligent sensing with a focus on cognitive radar and resources management for sensor systems. Additionally, he is a visiting Lecturer with RWTH Aachen University, Aachen, Germany.

Dr. Charlish is an Associate Editor for Radar Systems for the IEEE TRANSACTIONS ON AEROSPACE AND ELECTRONIC SYSTEMS, a Subject Editor for Radar, Sonar and Navigation for *IET Electronic Letters*, and an Associate Editor for *IET Radar, Sonar and Navigation*. He is currently an elected member of the IEEE Aerospace and Electronic Systems Society (AESS) Radar Systems Panel and the AESS Board of Governors. He is the 2021 recipient of the IEEE AESS Fred Nathanson Memorial Radar Award for contributions to radar resources management and cognitive radar. He is also a co-recipient of the 2021 IEEE AESS Harry Rowe Mimno Award for coauthoring the best IEEE AESS Systems Magazine paper in 2019 on "An Overview of Cognitive Radar: Past, Present and Future". He is active in the NATO community and received the NATO SET Panel Excellence Award in 2018 and the NATO SET Panel Early Career Award in 2019.