Deutscher Luft- und Raumfahrtkongress 2020
DocumentID: 530331

Predictive Scheduling and Opportunistic Medium Access for Shared-Spectrum Radio Systems in Aeronautical Communication

Leonard Fisser, Sebastian Lindner, Andreas Timm-Giel
Institute of Communication Networks (ComNets)
Hamburg University of Technology (TUHH), Germany
{leonard.fisser, sebastian.lindner, timm-giel}@tuhh.de

***Abstract*—Cognitive Radios (CRs) tackle the spectrum scarcity problem by allowing unlicensed access on already-licensed spectrum. Where primary user (PU) radio systems hold a privileged medium access, secondary users (SUs) try to utilize and use spare time-frequency resources to establish communication. At their core, CRs need to sense, detect and predict the medium access pattern of PUs in order to facilitate communication. A promising approach for inferring these predictions is the use of Machine Learning techniques and in particular Artifical Neural Networks (ANNs). ANNs try to learn a mapping from a set of inputs to a specific desired output and can therefore be directly applied to the problem of PU activity prediction. Especially Recurrent Neural Networks (RNNs) show adequate performance for noisy measurement data and prolonged training periods.**

In this work, the applicability of ANN-based channel state prediction is examined via a case study on the upcoming aeronautical communication technology L-band Digital Aeronautical Communications System (LDACS). LDACS is envisioned to reuse spectral resources currently primarily allocated to other aeronautical systems such as the Distance Measuring Equipment (DME) system. Primary and secondary users are modeled with respect to preliminary LDACS specifications and the performance of the proposed algorithms is evaluated via simulation.

We show that RNNs can function as prediction agents for SU LDACS medium access and that stringent reliability and interference requirements can be met. A supervised learning problem, together with an incremental learning strategy is proposed to address time-varying PU channel access patterns. Finally, a brief discussion on the use of predictions in a distributed scheduling approach is given. [1]

I. Introduction

Cognitive Radios (CRs) are a common solution to the problem of spectrum scarcity, as they attempt to ensure interference free non-cooperative coexistence between new and legacy systems. Non-cooperative coexistence requires no interface between primary user (PU) and secondary user (SU), allowing for simple and cost efficient deployments. Any information necessary for the prediction and collision avoidance of PUs has to be extracted from external sources such as channel sensing measurements. One of the most prominent examples for such a coexistence is the joint operation of Bluetooth and 802.11 WiFi. However, in this case both systems have equal rights to the spectrum access. In general, PUs can expect a privileged channel access, whereas SUs should minimize collisions with the PU system. A promising approach to achieve this is a prediction-based channel access for SU communication.

To infer predictions on future medium occupancy, stochastic or analytical methods can be deployed. However, analytical methods require deep knowledge about the present systems and operational circumstances which are not readily available in dynamic environments. As an alternative to this, the field of Machine Learning offers great flexibility and tools to address such time series prediction problems. Especially ANN in the context of supervised learning appear to be well suited to detect, learn and predict persistent PU medium access patterns and allow for interference-free coexistence. Traditional ANN applications assume that the underlying pattern to be predicted is static and shows few changes during the training time and duration of deployment. This assumption does not hold true for the channel access in dynamic communication networks as nodes can enter, move and leave the network. To ensure high long-term prediction quality, suitable learning methods have to be found that allow for adaption to changing environments while also ensuring that wrong predictions do not interfere with PUs.

In this paper, an incremental learning method for a supervised learning problem is proposed and compared to traditional one-shot learning. While incremental learning solves the problem of adapting to changing patterns long-term, it cannot ensure that interference is kept low during the relearning phase. To address this challenge, a dynamic threshold algorithm is proposed which adjusts the necessary level of confidence predictions have to show to be allowed for SU communication. We assume an oracle-based scheduler to focus solely on the learning and threshold aspect of this analysis. In a real implementation, an overhead-efficient distributive coordination approach has to be devised. In the final part of this work, the transfer of aforementioned concepts to a distributed scheduling approach is discussed and a simple Media Access Control (MAC) protocol is proposed addressing key challenges.

All approaches are validated by network simulations based on a case study about the upcoming LDACS technology. LDACS is envisioned to operate on frequencies currently already used by several aeronautical radio systems. A future deployment of LDACS represents a perfect testing environ-

[1] This work was conducted by the first author during his Master's thesis at the Institute of Communication Networks (ComNets) at the Hamburg University of Technology (TUHH).

doi:10.25967/530331

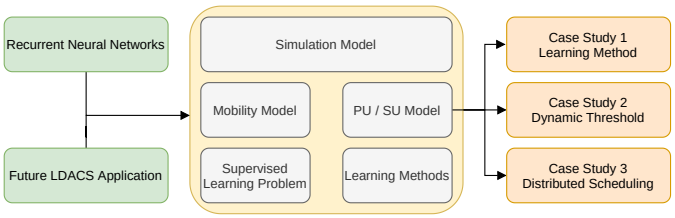


Fig. 1. Overview about the topics and structure of this paper.

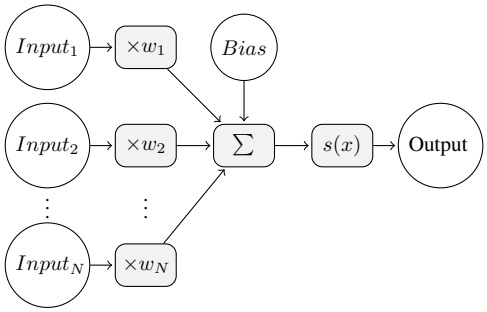


Fig. 2. Computational graph for a single neuron with N inputs, bias and sigmoid activation $s(x)$.

ment for the applicability of prediction-based CR using ANNs. An extensive simulation model for this scenario is derived and implemented in the OMNeT++ network simulator.

The rest of this paper is structured as follows. Section II summarizes relevant literature focusing on channel state prediction using Machine Learning. Afterwards, Section III briefly discusses the differences, drawbacks and advantages of feed-forward and recurrent ANNs. In Section IV the channel prediction problem is formulated and both a one-shot as well as a continuous learning method are proposed. Section V defines an approach to ensure reliability constraints for the interference on PU communication. The proposed techniques are evaluated via simulations of an aeronautical communication scenario in Section VI. A distributed scheduling protocol incorporating channel state predictions is proposed and briefly evaluated using the previously introduced simulation model in Section VII. Finally, Section VIII concludes this paper and gives research directions for future works.

Figure 1 gives an overview about the relationship between the discussed topics and the scope of the paper.

II. Related Literature

The field of CRs distinguishes between cooperative and non-cooperative - differentiated by the capability to coordinate medium access between primary and secondary users. Non-cooperative cognitive radio networks are more prominent, as no changes to the legacy PU system protocol are necessary. The success of the coexistence in this case is relying solely on the ability of the SU to non-intrusively access the medium and ensure an unimpaired operation of both systems. Such a medium access is non-trivial and research efforts have investigated means to infer information about the primary user activity. Possible primary user evasion strategies are:

- Geo-location assisted lookup of radio coverage maps and databases containing unused PU frequencies [1].
- Spectral sensing, analyzing and predicting primary user activity through statistical analysis, Machine Learning or a-priori known access patterns [2].
- Inherently planned coexistence mechanisms integrated into the MAC protocol.
  (Coexistence of WiFi and Bluetooth via Adaptive Frequency Hopping (AFH))

The work presented in this paper focuses on methods deploying Machine Learning to learn and predict PU activity.

In [3], Deep Reinforcement Learning (DRL) was used to find the optimal dynamic spectrum access for multiple SUs in a multi-channel environment. Each SU starts to transmit on a randomly selected channel and the feedback of all attempted transmissions are used to iteratively adapt the channel selecting policy to achieve maximal channel utilization. The ANN used for the DRL process features a leading Long-Short-Term-Memory (LSTM) layer. However, in contrast to the work presented in this paper, no PU was active in the studied scenarios. The objective of the DRL was to organize the medium access of a set of SUs without control overhead and not so much the avoidance of collisions with PUs.

In [2] the performance of LSTM and Feed-Forward Artificial Neural Networks (FFANNs) are compared for the time series prediction problem of medium access patterns. The superposition of multiple PU medium access patterns were given as input to the Artificial Neural Networks (NNs) with the objective to predict the subsequent channel occupation. Overall, LSTM networks showed higher prediction accuracy than simple FFANNs for the given prediction problem. Similarly, the work presented here will only focus on LSTM based networks, but will increase the complexity of the prediction problem by modeling real-world scenarios in which the pattern changes over time.

III. Feed-Forward and Recurrent Neural Networks

Artifical Neural Networks (ANNs) are widely used in today's Machine Learning applications and are especially prominent in supervised learning problems. In general, ANNs consist of layers of neurons and edges connecting neurons between subsequent layers. Neurons represent the mathematical operations which are applied to the respective layer's input. An example for a simple neuron is given in Figure 2, showing incoming edge weights w, bias as well as activation function $s(x)$. Common activation functions include the sigmoid and rectified linear unit.

In traditional FFANNs, sets of neurons are aggregated to layers and complemented by an input and output layer. The flow of information as well as mathematical operation is always from input layer to output layer. In contrast, RNNs can include feedback loops connecting layers to preceding ones, thus creating feedback loops. RNNs are designed to adapt to temporal patterns inside the feature vector. LSTM networks [4]

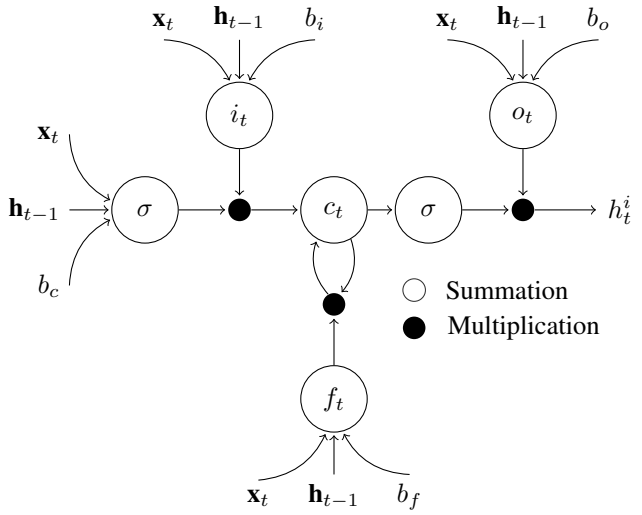


Fig. 3. Single cell LSTM block with cell c_t, input gate i_t, output gate o_t, forget gate f_t and sigmoid activation σ.

TABLE I
SUMMARY OF ANN PARAMETERS USED THROUGHOUT THIS WORK.

Parameter	Value
Neurons LSTM Layer	96
Neurons Dense Layer 1	96
Dropout	0.2
Recurrent Dropout	0.1
Loss Function	Binary Cross-Entropy (BCE)
Optimizer	ADADELTA

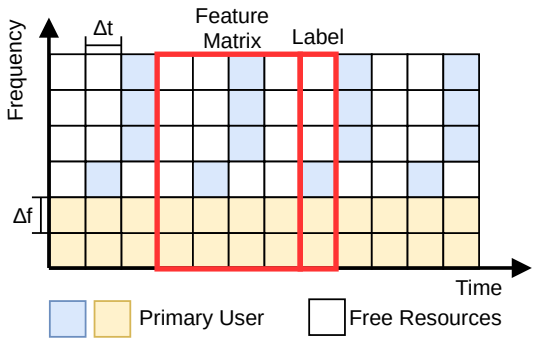


Fig. 4. Medium access prediction problem visualization. Colored squares represent occupied resources.

are a special implementation of RNNs and include, in addition to feedback edges, a memory cell storing information from previous inputs. Besides the feedback edges, three gates are used to facilitate control over the flow of new data into the cell, flow out of the cell and influence of the memory cell value. A LSTM cell is given in Figure 3 showing the additional features compared to the simple neuron in Figure 2, where x_t is the input vector, h_{t-1} is the last output vector and b_c,b_i,b_f,b_o are the different gate biases. Further iterations of the LSTM cell layout were proposed, adding more feedback edges [5] and a forget gate [6]. The ability to store information inside the RNN, together with feedback edges, allow LSTM networks to learn temporal patterns over several time slots with reduced computational complexity and are therefore prime candidates for time series prediction problems.

Supervised learning problems require samples of the data set to be a tuple of feature vector and corresponding label. The feature vector represents the input and the label represents the desired output given the feature vector. It is the ANN's objective to learn the unknown mapping between the two. During the training stage, a subset of all available samples, the training set, is presented to the ANN's and its output is compared to the expected label. Based on the difference between actual output and expected one, the error is back-propagated through the ANN and computational weights are updated to generate a smaller error the next time the same sample is presented. After one iteration over the whole data set is completed, the ANN's performance is tested on a new subset of samples, the validation set. Since no training was conducted on the samples of the validation set, the prediction quality achieved by the ANN is a metric for the ability to generalize the detection of patterns and not overtrain on specific training set samples. After each sample of the data set is presented to the ANN once, the process is repeated until a suitable preemption condition is met. Popular preemption strategies include but are not limited to validation accuracy, maximum number of iterations or convergence.

In this work only RNNs are used during evaluation as they are specifically designed to address time-series prediction problems and showed faster convergence in preliminary tests. The final LSTM RNN configuration deployed in all following evaluations as well as complementary details are summarized in Table I.

IV. Channel State Prediction Problem Formulation

In this section the channel prediction problem is formulated with respect to one-shot and continuous learning. The general prediction problem is motivated by the situation depicted in Figure 4. Available spectral resources are discretized in time as well as in frequency, resulting in a grid like structure of individual resource elements, each with a duration of Δt and bandwidth of Δf. The resource grid is then populated by channel assessment information for each discrete time slot. After sensing the channel for PU activity over the duration of Δt each resource element of the respective time slot is either marked free or blocked. At each time slot, the channel measurements add more information to the channel state matrix, slowly building up an extensive history of medium accesses. The objective of the prediction agent is to identify patterns in the channel state matrix and infer predictions for the next time slot. It has to be mentioned that while the SU may utilize the resource grid directly as a basis for a Time Devision Multiplex (TDM) medium access scheme, PUs are not aware of the grid and may operate synchronously or asynchronously with respect to Δt or Δf.

While the channel matrix represents the complete measurement history, learning methods have to operate on smaller parts called samples. Samples represent the smallest data unit a learning algorithm can work with and a high number of unique samples generally increases the ANN's ability to learn.

A sample consists of two parts as discussed in Section III. First, the feature matrix contains a given sequence of time slots spanning from x_t to x_{t+H} where H is the duration of the matrix in number of time slots. Second, the sample is supplemented by a label which can be considered the desired prediction output for a given feature matrix. As we are interested in the channel occupation in the next time slot, the label for a given data sequence from t to $t+H$ can be found at time slot $t+H+1$. In order to generate the most samples given a finite channel state history, a sliding window slicing operation is deployed. This method generates $N-H-1$ samples (with feature matrix and label part) given a channel state history of N time slots.

A. One-Shot Learning

Considering the resource grid and sample generation described in the previous section, the prediction problem can be categorized as a simple supervised learning problem for which training data is easily generated. One-shot learning refers to the completeness of the data set containing all available samples and a singular training effort. After a finite channel measurement period, no new samples are added to the channel state matrix. The data set can be considered complete/closed. Then, the channel state matrix is converted to the data set by the aforementioned slicing operation and is subsequently ready for training of the ANN. Training is conducted using the ADADELTA [7] optimizer algorithm, which showed a slightly better performance compared to the more commonly used ADAM optimizer and is suited for one-shot as well as continuous learning. The available data set is split with a 75/25 ratio into training and validation set and the training is preempted after 30 successive epochs with no improvement in the validation accuracy. After training, the neural network is deployed to predict the channel access of PUs. During operational use of the neural network, the channel state history of the last H time slots is input and the ANN infers a prediction on the channel state for the next time slot.

B. Continuous Learning

The main drawback of the previously introduced one-shot learning is the incapability to adapt to a changing pattern. If the channel access pattern of the collective sum of the PUs changes due to positional changes, the prediction quality will deteriorate with respect to changes in the pattern. The neural network would only be able to adapt to the new pattern, if a new measuring and training phase would be triggered. In many applications the activity and presence of PUs is dynamic and may change to quick over time to warrant repeated resets. To combat this, incremental training methods can be designed and continuously used during the operation of the ANN. The incremental nature of the training process can adapt the neural network to the changing environment without requiring a complete reset. It operates under the assumptions that the changes in the pattern between subsequent time slots is small.

The exact implementation of the incremental learning algorithm used in this work is summarized in Algorithm 1. To better compare the performance, the same loss function and optimizer as in the one-shot training are used. The ADADELTA optimizer is especially well equipped to deal with prolonged learning periods, as only a limited number of batches are considered in the learning rate adjustment. At the beginning of every time slot, the newly available measurement is appended to the channel state matrix and the thereby newly available sample is added to the sample pool of finite size. If the maximum number of samples in this pool is exceeded, the oldest one is purged from it. This ensures that the sample pool only includes the most recent samples, representing the current PU behavior. After the new sample is added, an incremental training process is triggered. The training during this step is similar in procedure to the process of one-shot learning, but is significantly lower in effort. N_s samples are randomly selected from the sample pool and put into the training pool. The training pool is then passed to the ANN for a single learning effort. This process is repeated several times specified by the maximum number of epochs per time slot. Using the newly adapted ANN, a prediction for the next time slot is inferred and returned. This procedure is repeated at the beginning of every time slot as a new channel measurement becomes available.

Algorithm 1 Online Learning Method

```
function ONLINELEARNING(new channel measurement)
    sample pool ← add new sample
    if size(sample pool) ≥ maxPoolSize then
        delete oldest sample
    end if
    for i ≤ maxEpochs do
        training pool ← pick N_s samples randomly
        single training effort using training pool
        i ← i + 1
    end for
    prediction ← ANN forward pass
    return prediction
end function
```

V. Reliability Requirements in Prediction Problems

The ANN's output is a soft value ($r \in \mathbb{R}$, $0 \leq r \leq 1$) describing the networks certainty that a given frequency resource element is free or occupied in the subsequent time slot. In order to deploy traditional scheduling methods, like Round Robin or Proportionally Fair, the soft-valued predictions have to be rounded to binary decisions of occupied and free channels. An exemplary histogram of prediction soft-values is given in Figure 5 for a well performing and a bad performing ANN. It can be seen that the well performing ANN is able to separate good and bad channels more clearly, by outputting more values closer to the extreme values of zero and one. The bad performing network on the other hand, is only barely able to issue predictions below the 0.5 threshold and overall achieves worse prediction quality.

A decision threshold of 0.5 naturally maximizes the confidence of the resulting hard-valued prediction. However, in the case of stringent requirements on interference or transmission successes, maximizing the confidence might not be the

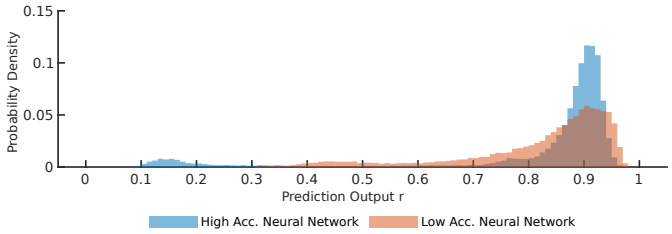


Fig. 5. Prediction histogram for two networks achieving high and low prediction accuracy. '1'-labels are roughly five times more prevalent in the data set.

Algorithm 2 Adaptive Threshold Management

function UPDATETHRESHOLD(reported collision rate γ_t)
  $\gamma = (1-\beta)\gamma + \beta\gamma_t$
  if $\gamma > \gamma_{\text{target}}$ **then**
    $T_{t+1} = (\gamma/\gamma_{\text{target}})T_t$
  else if T_t insufficient **then**
    $T_{t+1} = \max(T_t - \Delta T, 0.5)$
  end if
  return T_{t+1}
end function

optimal policy for classifying the resource elements as free or occupied. Instead, the decision threshold can be increased to require a higher prediction confidence to allow the use of a given resource element for transmission of SU data. Such a conservative policy will decrease the number of collisions between PUs and SUs but will also result in overall less resource elements to be detected as free, leading to a decreased throughput for the SU system. The trade-off between interference and achieving high throughput can be considered as a traditional control system problem. In the scope of this work, a simple decision threshold Algorithm 2 is used to dynamically enforce a given target collision rate γ_{target}. Every time slot t, the collision γ_t is reported to the threshold management agent and passed through an Exponentially Weighted Moving Average (EWMA) filter. The resulting collision rate γ describes the caused interference at the PU side and has to be kept as close as possible to its target value γ_{target}. The threshold management agent tries to influence the number of collisions caused by the SU system by adjusting the decision threshold. If the collision rate is higher than the target the threshold is multiplicatively increased by the ratio of reported and target error rate. If the collision rate is below the target and if in the last time slot not all SUs were scheduled, then the threshold is additively decreased by a fixed step size ΔT. An additive increase multiplicative decrease (AIMD) strategy ensures that the target error rate is able to react quickly in the case of a high number of errors while also releasing resources slow enough to not cause oscillations. The decision threshold cannot be decreased below the 0.5 threshold.

TABLE II
SUMMARY PU AND SU TECHNOLOGY PARAMETER.

Parameter	LDACS	DME	JTIDS
Frequency Range [MHz]	985 - 1008 1048 - 1071	960 - 1215	960 - 1215
Channel Bandwidth	0.5 MHz	1 MHz	3 MHz
Medium Access	tba	Static	Frequency Hopping
Frame Length	6.4 ms	15.5 µs	12 s

TABLE III
SUMMARY OF SCENARIO AND TOPOLOGY PARAMETERS.

Parameter	Value
No. D2D Connections	32
No. Base Stations	32
No. DME Connections	32
No. JTIDS Connections	32
No. Channels	96
Channels DME	0 - 63
Channels JTIDS	64 - 95
Channels LDACS	0 - 95
Channel Sensing Error Prob.	5%

VI. CASE STUDY: COEXISTENCE OF DME/JTIDS AND LDACS

In order to test performance and analyze the proposed learning method as well as the dynamic threshold algorithm a network model is derived and simulated using the OMNeT++ network simulator [8] and INET framework [9]. The analysis section is structured as follows. First, all necessary simulation model parameters as well as two evaluation scenarios are defined in Section VI-A Afterwards, the performance of the proposed algorithms is evaluated in Section VI-B and discussed.

A. Simulation Model

The upcoming deployment of the LDACS technology is chosen as a test framework, as it will have to coexist with multiple legacy systems. LDACS is envisioned to share the available bandwidth with up to five different radio systems:

- Distance Measuring Equipment (DME)
- Joint Tactial Information Distribution System (JTIDS)
- Tactical Air Navigation (TACAN)
- Secondary Surveillance Radar (SSR)
- Universal Access Transceiver (UAT)

This evaluation concentrates on the DME and JTIDS systems as they exhibit predictable patterns and their operation will probably interfere with the LDACS the most. The LDACS takes thereby the role of SU whereas JTIDS and DME take the role of PUs. Since the LDACS technology also covers aircraft to aircraft communication, the SUs hold full-duplex Device-to-Device (D2D) capabilities. Operational parameters for LDACS, DME and JTIDS are summarized in Table II An overview about complementary simulation parameters is given in Table III From the parameters it can be observed that several simplification can be made without losing general validity. First, the fast channel hopping, combined with a long transmission duration deployed by the JTIDS will not allow

sufficient resource to be allocated by the LDACS in any way. If the JTIDS systems actively operates, all used bandwidth will be completely unavailable for the LDACS. In comparison to this, the DME system allows enough resources to be allocated for LDACS transmissions between two interrogation and response pulses. Summarizing, perfect predictions would always block any resources in the JTIDS spectrum and would periodically allow for transmission on frequencies of the DME system. A summary of the spectrum occupation of JTIDS and DME is given in Figure 4, where yellow entries represent JTIDS activity, blue entries DME and white entries available resources. Further details about the interference and interaction between LDACS and legacy systems can be found in [10] and [11]. Both works exhaustively describe modeling of legacy systems with respect to LDACS.

In addition to the modeling of the primary and secondary users, the actual position of aircraft and DME base stations have to be specified. Positions for DME base stations are taken from online resources available at [12]. For the position of aircraft a $50\,\mathrm{M} \times 50\,\mathrm{M}$ area around the Heathrow Airport is used. Aircraft are randomly distributed in the vicinity of DME base stations and assign themselves to the nearest available one. Since aircraft are generally not stationary, mobility has to be modeled as well. As we are only interested in the change of the spectrum access pattern caused by topological changes, a simplified mobility model is used. After an initial time period, a set of aircraft change their positions randomly to a different DME base station and re-assign themselves to it. While this mobility model is unrealistic and assumes instantaneous position changes, it is sufficient to cause changes in the PU channel access pattern. In order to analyze the transient behavior of the incremental learning method and the sensitivity of the prediction quality to changes in the pattern, two different mobility events are considered.

a) Mobility Event A: starts after $1.8\,\mathrm{s}$ of simulation time. After this initial period 22 aircraft interchange positions over a period of $1\,\mathrm{s}$ - two aircraft every $100\,\mathrm{ms}$. If one considers the time slot length of LDACS, this disruption can be considered slow, as the event spans over 154 samples. After all 22 aircraft changed positions, no more changes occur.

b) Mobility Event B: is closely related to Mobility Event A. But this time, aircraft change position over a duration of only $200\,\mathrm{ms}$ instead of $1\,\mathrm{s}$. Roughly 30 time slots pass during this $200\,\mathrm{ms}$ and therefore the changes are considered fast.

For the sake of completeness, two further simulation assumptions have to be mentioned. First, analog transmission pulses used by the DME and JTIDS systems are modeled by packet transmissions by setting according packet sizes. No analog modeling of signals takes place. Second, any collision between two transmissions corrupts the SU transmission but only adds interference to the PU one. This simplification is justified by the fact that PU systems transmit on much higher transmissions powers than the envisioned LDACS system.

B. Simulation Results

a) One-Shot Learning vs. Incremental Learning: In this simulation study the proficiency of the incremental learning method is analyzed and compared to the performance of one-shot learning. It is expected that changes in the PU medium access pattern cause the quality of predictions to deteriorate resulting in more collisions between PU and SU. For the simulation setup, the before described simulation parameters and mobility patterns are used. For the incremental learning approach, the algorithm presented in Section IV is used. The batch mean of collisions caused by SU is taken as the Key Performance Indicator (KPI) to evaluate the prediction quality. Figure 6 shows the KPI over the simulation time for four different configurations. Colors blue and red show the performance of the one-shot learning algorithm for mobility events A and B, similarly colors yellow and purple show the performance for the incremental learning approach. Both learning approaches are able to correctly learn the static PU pattern as can be seen in the performance from $0.5\,\mathrm{s}$ up to $1.8\,\mathrm{s}$ where the number of collisions is close to zero. However, different behaviors can be observed for the two approaches. The one-shot learning approach produces gradually more collisions after the start of the mobility event, indicating a degradation in the prediction quality. Interestingly, the performance loss follows the change of the mobility event - no sudden jump can be observed. After both events finish, the performance of the one-shot learning ANN remains constant, but worse than before the disruptions. Similar observations can be made for the incremental learning approach. However, after around $200\,\mathrm{ms}$, sufficient new samples are collected and added to the sample pool to allow the ANN to adapt to the new pattern. Consequentially, the effect of the mobility event starts to vanish. Interestingly, a return to pre-mobility event performance levels is only possible after the event terminates and the pattern becomes static again. The continuous learning efforts efficiently reduced the effect of changes in the prediction pattern.

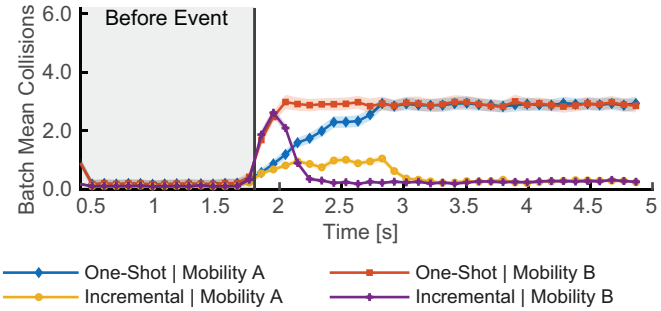
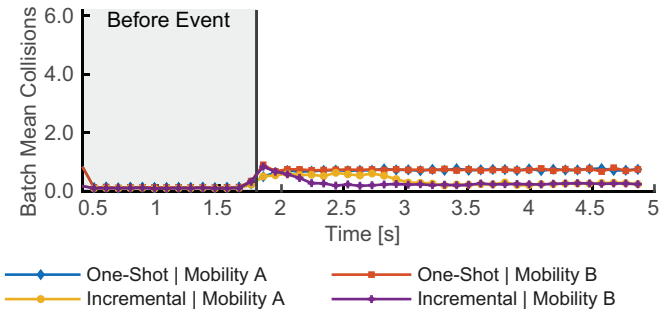
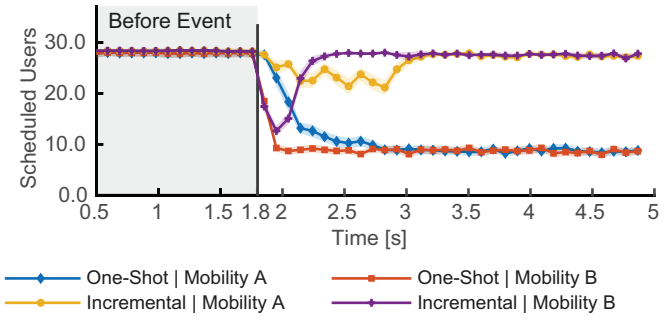


Fig. 6. Number of collisions caused by the different learning methods for the two different mobility events over the simulation time.

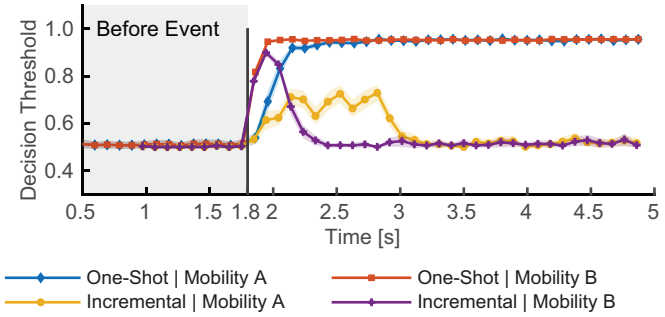
b) Static Threshold vs. Dynamic Threshold: In the previous study, the capabilities of an incremental learning approach were analyzed. While the approach was able to return to good performance levels after the changes in the PU access pattern ceased, the number of collisions caused during this relearning period was still unacceptably high. Therefore, the dynamic threshold algorithm introduced in Section V is implemented



(a) Batch Mean Collisions



(b) Number of Scheduled Users



(c) Decision Threshold Trend

Fig. 7. KPIs for configurations in which a dynamic threshold is used to control the interference between PU and SU communication.

and added to both learning methods. No other changes were made to the rest of the simulation model. In addition to the number of collisions cased by the SU system, the number of active connections as well as the decision threshold are tracked and analyzed. By increasing the threshold to accept less free resource predictions, the number of resources allowed to be used for SU communication decreases as well, resulting in less throughput for the SU system. The target collision rate is set to $\gamma_{\text{target}} = 0.02$ (equal to 0.64 collisions per time slot), the filter variable is set to $\beta = 0.1$ and the threshold decrement is set to $\Delta T = 0.05$. All three KPIs are displayed in Figure 7. The simulation results show that the number of scheduled users and the number of collisions caused are the same for all configurations before any mobility event happens. In fact, the decision threshold remains at the 0.5 minimum, resulting in the same behavior and performance as in the previous simulation. By the time the mobility events cause sufficient performance degradation, the threshold increases across all configurations, resulting in less scheduled SUs. The number of collisions is kept in check and below the target collision rate. In comparison to the previous analysis, the number of collisions during the mobility events is kept low by sacrificing throughput and transmission opportunities. Since the incremental learning approach is able to regain prediction accuracy after the mobility events, the threshold is lowered back to levels around 0.5 again. The threshold for the one-shot learning approach is not able to return to its minimum but has to remain high instead, resulting in long-term SU system performance degradation.

VII. Concept: Distributive Scheduling

In the previously presented simulation results and analysis oracle-based scheduling as well as a perfect signaling channels were assumed. A central entity collected channel measurements, trained the ANN and inferred a prediction for the next time slot. The resulting transmission scheduling assigns resources to each SU optimally.

However, the use of prediction based medium access is not limited to central scheduling algorithms, but can be applied to ad-hoc device-to-device networks as well. Instead of a central entity, each SU is equipped with measuring equipment, a neural network and a control channel to adjacent users. The core idea of cooperative scheduling is that each SU first predicts which channels are likely to be available in the next time slot locally and then compares its own result with the information of its transmission destination. Such a scheduling approach introduces three new challenges which have to be addressed in a future protocol design. First, the process of negotiating resource elements suitable for both SUs is linked to high control overhead. If two nodes want to find a suitable resource element for communication in the next time slot, at least one node has to share and forward its prediction table. Second, in addition to collisions between PU and SU, collisions between SUs can occur as well. Two different pairs of SUs might decide on the same resource element to transmit data on and collisions might occur. Therefore, suitable collision avoidance or detection mechanisms have to be deployed, introducing even more overhead. Finally, SU traffic will be present in the channel measurements and consequentially in the channel state matrix. The centralized approach was able to eliminate SU transmissions from measurements because it had perfect knowledge about them. This assumption does not hold true for the distributed scheduling case where nodes only know about their own transmissions. Channel acitvity measurements cannot differentiate between PU and SU transmission. The activity of SU transmissions will therefore mask any other medium access. A possible solution would to be to design the distributed scheduling to follow a persistent, detectable and predictable pattern. This way other SUs would be able to avoid collisions with SU transmissions in the same way they avoid PU transmissions.

Because the integration of channel state prediction into new or already existing MAC protocols surpasses the scope of

this paper, only a simple cooperative scheduling approach is given and evaluated using the previously described simulation model. The scheduling process for transmissions between two nodes is given in Figure 8. Each node (Transmitter A and Receiver B) receives new prediction values from their local neural network at the start of a scheduling period. Afterwards, each prediction value larger than 0 is discretized with respect to k values. For $k = 2$ the predictions represent only binary decision variables, whereas higher k values allow for a differentiation between confidences. Predictions below 0.5 are directly set to zero, analog to the dynamic threshold consideration presented in the previous section. Transmitter A sends a request channel primitive to receiver B via an ideal control channel. The request could include common parameters like number of bits to be transmitted, synchronization information or routing information as well as the discretized prediction vector. On the receiver side, both prediction vectors are combined via multiplication resulting in a joint prediction. A multiplicative combination is used to ensure that a single bad prediction will result in the receiver discarding the possible resource element, while channels which are good for both nodes retain high confidence values. After soft-combining, a simple persistent scheduling mechanism is used. If transmitter A successfully transmitted data to receiver B in the previous time slot, the channel of this previous transmission ought to be selected again. Obviously, the transmission is not to be confirmed, if the prediction value for said channel is zero. If the previous channel is to be predicted as occupied, transmission is abstained and a counter is incremented. After $c_{\max}$ abstentions or if the last transmission was not with transmitter A, a new channel is selected via a weighted random draw. The weighted random draw makes use of soft-valued prediction confidence levels, while also addressing the problem of multiple SUs predicting the same channel to be free in the next time slot. Channels with good confidences are selected often but not always. Overall, the mechanism introduces the necessary persistent behavior of SU channel accesses, which can be detected and predicted by the ANNs of other SUs.

a) Simulation Results for Distributed Scheduling: The scheduling algorithm is tested via the same simulation model introduced in the previous section minus the the mobility model. However, no central entity takes care of predicting and scheduling free resource to SUs but instead each node has to negotiate transmission opportunities with its peer in a distributed way. Similar to the evaluation in the previous section, the number of collisions per attempted transmission is given in Figure 9 for the central approach, the distributed approach (with two different quantization resolutions) and for two random scheduling approaches. The random scheduling approaches serve as benchmark and are divided into two classes. In *True Random* each receiver selects a random channel and confirms the transmission to the transmitter. In *Bounded Random* the channels occupied by the JTIDS are excluded from the channel pool and cannot be selected by the random draw. This bounding ensures that resources which are known to be always occupied (see discussion in Section VI) do not bias the benchmark. The number of collisions are further differentiated into collisions between SUs and collisions between SUs and PUs to give more insight into the resulting interference. Comparing the result shown in Figure 9 it is obvious that the prediction based scheduling approaches show less collisions per transmission than any random approach. However, in comparison to the central scheduling approach, the distributed scheduling approach shows roughly six times more collisions. This is mainly due to the fact that collisions between SUs are possible now, which was not the case in the centralized approach. For the distributed approaches, it can be seen that binary prediction values performed as well as soft-value prediction values with a resolution of $1\,\mathrm{Byte}$. The main reason for no differences between both approaches is in the prediction soft-value distribution. A well trained ANN will only output predictions close to their maximum values one and zero. With that, the general loss of information by converting them to binary predictions is low. Subsequently, the reduced control effort necessary heavily outweighs the marginal performance loss.

Overall, the discussion given in this section can only highlight the achievable performance of a prediction-based distributed scheduling scheme. A more detailed and thorough integration of predictions into existing or new MAC protocols would be necessary to make a full statement on applicability.

VIII. Conclusion and Future Work

In this paper we showed that RNNs can be used to predict specific PU patterns in the sense of CR. The careful design of a supervised learning method together with extensions like incremental learning ensured reliable long-term performance despite dynamic environments. A trade-off could be identified between sacrificing SU throughput for a more reliable interference mitigation. The proposed dynamic threshold algorithm is able to adjust the number of scheduled users in periods of dynamic pattern changes and uncertainty. After sufficient relearning by the incremental learning approach, the threshold algorithm is able to restore performance. For oracle-based scheduling systems, RNNs used for channel state prediction showed promising performance.

Moving the inferring process, learning and scheduling to each node instead results in a distributed scheduling problem. The aspect of prediction exchange and interaction between SUs was investigated and a suitable scheduling algorithm was derived. The proposed scheduling takes care of the three main challenges experienced in this investigation. Namely, the need for persistent channel access, the avoidance of SU collisions and the exchange of control information including channel state predictions. In comparison to the central scheduling approach and both random benchmarks, the proposed approach performed well. However, a fairer comparison would include more sophisticated MAC protocols specifically designed for device-to-device communication.

Building on this work, future research should focus on a full integration of prediction-based scheduling into an existing or new MAC protocol. Additionally, further actions have to be taken to ensure high reliability requirements in aeronautical communication and the performance should be evaluated in

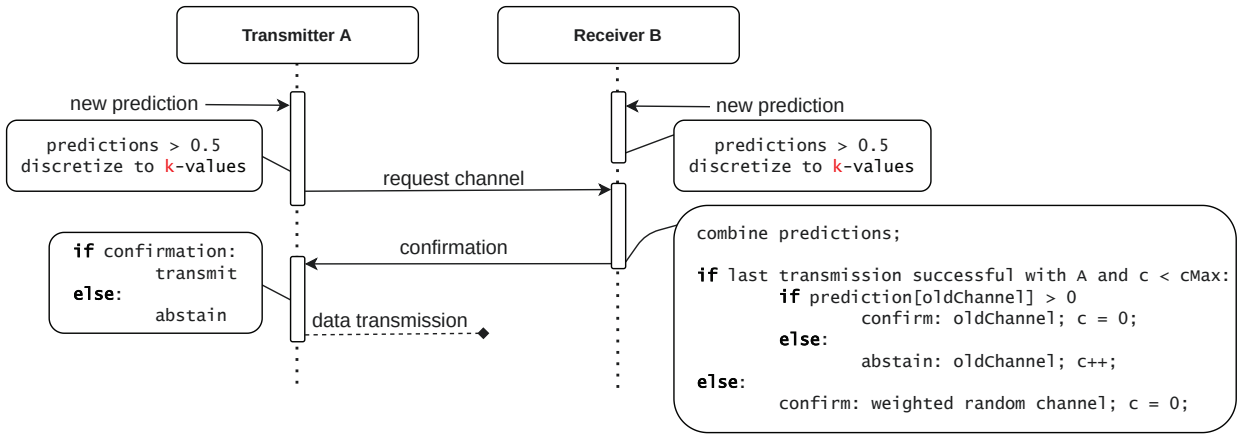


Fig. 8. Distributed scheduling using prediction values discretized to k values.

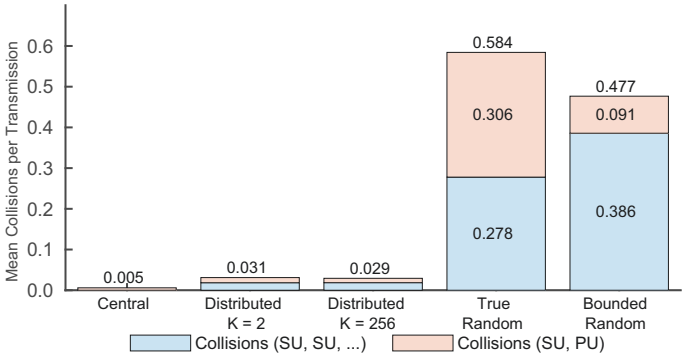


Fig. 9. Number of collisions per transmission for the five different configurations.

more sophisticated simulation models. Extensions to the presented approach in this paper might include predicting more than one time slot. Using multi time slot predictions can decrease signaling overhead while also increasing throughput for the SU system.

References

[1] M. Nekovee, "Quantifying the availability of TV white spaces for cognitive radio operation in the UK," in *2009 IEEE International Conference on Communications Workshops*. IEEE, 2009, pp. 1–5.

[2] L. Yu, J. Chen, and G. Ding, "Spectrum prediction via long short term memory," in *2017 3rd IEEE International Conference on Computer and Communications (ICCC)*. IEEE, 2017, pp. 643–647.

[3] O. Naparstek and K. Cohen, "Deep multi-user reinforcement learning for dynamic spectrum access in multichannel wireless networks," in *GLOBECOM 2017-2017 IEEE Global Communications Conference*. IEEE, 2017, pp. 1–7.

[4] S. Hochreiter and J. Schmidhuber, "LONG SHORT-TERM MEOMRY," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[5] F. A. Gers, N. N. Schraudolph, and J. Schmidhuber, "Learning precise timing with LSTM recurrent networks," *Journal of machine learning research*, vol. 3, no. Aug, pp. 115–143, 2002.

[6] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with LSTM," *Neural computation*, vol. 12, no. 10, pp. 2451–2471, 2000.

[7] M. D. Zeiler, "ADADELTA: AN ADAPTIVE LEARNING RATE METHOD," *arXiv preprint arXiv:1212.5701*, 2012.

[8] "OMNeT++ discrete event simulator."

[9] "INET framework - INET framework."

[10] U. Epple and M. Schnell, "OVERVIEW OF INTERFERENCE SITUATION AND MITIGATION TECHNIQUES FOR LDACS1," in *2011 IEEE/AIAA 30th Digital Avionics Systems Conference*. IEEE, 2011, pp. 4C5–1.

[11] U. Epple, F. Hoffmann, and M. Schnell, "MODELING DME INTERFERENCE IMPACT ON LDACS1," in *2012 Integrated Communications, Navigation and Surveillance Conference*. IEEE, 2012, pp. G7–1.

[12] "http://www.paris.icao.int/," *European and North Atlantic (EUR/NAT) Office of the International Civil Aviation Organization*.