# A Deep Learning Assisted Approach for Minimizing the Age of Information in a WiFi Network

Suyang Wang[1], Yu Cheng[1]
[1]*Department of Electrical and Computer Engineering, Illinois Institute of Technology, USA 60616*
*Email: swang133@hawk.iit.edu; cheng@iit.edu*

2022 IEEE 19th International Conference on Mobile Ad Hoc and Smart Systems (MASS) | 978-1-6654-7180-0/22/$31.00 ©2022 IEEE | DOI: 10.1109/MASS56207.2022.00016

***Abstract*—Motivated by the demands of time-sensitive applications, our paper studies methods for the age of information (AoI) minimization over a WiFi Network. Specifically, the AoI of a labeled device sending updates to the access point (AP) is of our concern. However, it is non-trivial to investigate the AoI optimization problem in such a scenario where an arbitrary number of background devices are also delivering updates to the AP; all of the network devices follow IEEE 802.11 based Medium access control (MAC) protocol to contend for the channel. Current works on AoI optimization over a WiFi network do not offer any practical methods for a single user to minimize its AoI readily and adaptively, limited to system AoI optimization with homogeneous traffic modeling or simplified MAC modeling. This paper develops a deep learning facilitated AoI optimization algorithm that provides direct guidance for easy implementation. Our novel method integrates service time analysis in 802.11 MAC and AoI queueing analysis with the deep learning enabled channel condition prediction. Specifically, we gather traffic rates of each node from the AP and train a channel condition estimation model. A labeled node can leverage the well-trained learning model to obtain accurate expected MAC service time and adaptively adjust its sending rate for minimal AoI. Simulation results show that our learning model can accurately estimate channels, and our algorithm guarantees fast adjustment of AoI-minimized sending rate.**

***Index Terms*—Age of information, WiFi, deep learning, queueing systems, medium access control, optimization.**

## I. Introduction

With the considerable advancements in network technology and the enormous growth in portable devices, modern life has undergone a significant transformation. The widespread connectivity and development of mobile devices foster real-time applications like remote monitoring and draw attention to timely information updates. As a result, the age of information (AoI), which indicates the amount of time since the most recent update, has lately been considered as a vital metric for determining how fresh the information is. Therefore, it is crucial to investigate the AoI minimization problem for remote monitoring via a WiFi network since many internet-of-things (IoT) devices access the internet via this technology. In addition, it is vital to recognize that "minimizing the AoI" and "minimizing the network delay" should not be used interchangeably.

AoI minimization is challenging in this situation because wireless devices could use generic and heterogeneous data flow to convey updates. Users' competition for channel access also has a complicated impact on queue analysis. The AoI-minimizing update rate using queue analysis has been studied in the past [1]–[6]. [7]–[10] analyze AoI optimization in random access networks, such as Slotted-ALOHA networks, CSMA networks, and massive Internet of Things (IoT). Specifically, to lower the network-wide AoI, the authors of [7] propose an ALOHA-like stationary random access. The fixed channel access probability that is assigned to each IoT device is not practical in most situations. Stochastic Hybrid Systems (SHS) are used by the authors of [10] to model a CSMA environment with the goal of lowering the overall average AoI of the entire network. [11], [12] study AoI optimization in centralized multiple access mechanisms. However, because the analyses are either based on a symmetric configuration in which the background nodes compete to access the wireless channel with a homogeneous traffic arrival rate or are oversimplified with strong assumptions, these studies only have a limited potential for practical application. Therefore, it is crucial to look for ways for portable devices to behave logically to get the best AoI in various network settings.

This work focuses on minimizing a WiFi user's AoI in a heterogeneous and arbitrary network environment and offers a straightforward solution for easy implementation. Notably, we study the method for a WiFi station to adjust its message updating rate to achieve minimal AoI. Since the contention-based WiFi network brings much fluctuation and uncertainty to the wireless environment, we thus leverage the deep learning technique for quick and robust channel condition prediction, thereby getting an accurate estimation of the current service rate of the device's MAC queue. Then following an AoI-optimal offered load we discovered through simulation, one can determine its traffic updating rate for a minimal average AoI. We also define a successful transmission probability among all the background nodes for accurate service rate estimation. With the assistance of our well-trained deep learning model, our simulation results reveal that our strategy is both universal and successful in a WiFi network with an arbitrary number of nodes of different updating rates.

The main contributions of this paper are listed as follows.

1) We propose a deep learning facilitated framework that minimizes a WiFi device's AoI. We combine the MAC layer service time analysis in 802.11 protocol and the AoI analysis in Single-Source Single-Server queues to enable the WiFi user to achieve optimal AoI promptly.
2) We design an innovative and reliable deep learning method to efficiently obtain the current channel conditions, providing crucial information for accurately eval-

uating service time. To the best of our knowledge, the current literature does not include an approach capable of reducing AoI while maintaining a low level of complexity.

3) To tackle the analysis of the network with heterogeneous background WiFi devices, We extract the actual successful transmission probability across all of the background nodes based on the packets received rates at AP from each device. Such a valuable method ensures accurate service time estimation and has little burden assisted by our well-trained deep learning model.
4) We gather vast amounts of data via simulation and develop a deep learning model with great accuracy for channel condition estimation. Such a fine-tuned model is valuable to the community.
5) We do simulations with different network configurations. The simulation findings support the efficacy and precision of our research by demonstrating that the minimum AoI may be very nearly reached using the suggested strategy in any network context.

The remainder of this paper is structured as follows. The system model is illustrated in Section II. In Section III, we describe the methods proposed for AoI optimization. Numerical results and methods of the simulation are presented in Section IV. The related work to the problem we investigate is discussed in Section V. Finally, we summarize our work in Section VI.

## II. SYSTEM MODEL

In this part, we outline our proposed framework, give major aspects of the AoI analysis for the scenario of interest, and show each step in detail.

### A. Data Updating over WiFi

In the context of an IEEE 802.11 network, we are interested in a scenario where a specific wireless device known as a labeled device (labeled DEV) is associated with an access point (AP). The other wireless devices that are linked to this access point are referred to as background devices. These electronic devices compete with one another to access the wireless channel per the contention-based IEEE 802.11 standard. In this scenario, we plan to investigate a rational strategy for the labeled DEV to achieve optimal average AoI based on the knowledge requested from the AP and the prediction of the deep learning model in a cost-efficient manner. The terms DEV and node may be used interchangeably for the rest of this study.

A WiFi network with various wireless devices coupled with an AP is illustrated in Fig. 1. The AP serves as a server for the MAC queues of all connected wireless devices, acting as a portal for network services. The data updates are transmitted by the labeled DEV at arrival rate $\lambda$ via a MAC queue and served by the AP with a service rate $\mu$. In order to attain the ideal AoI for the labeled DEV, we request from the AP the packets received rates from each background device as the input of the learning model and compute the service rate $mu$

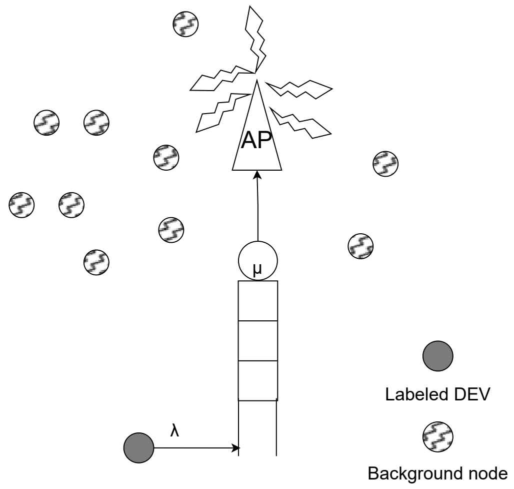


Fig. 1: A labeled DEV transmits updates across a WiFi network's MAC queue.

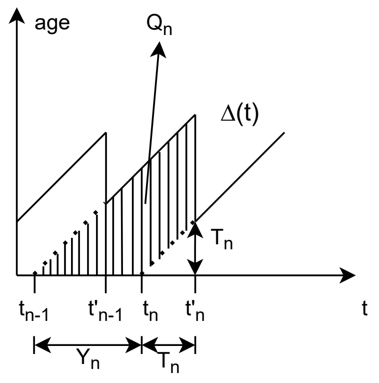


Fig. 2: Sawtooth age waveform.

using the channel condition prediction produced from a well-trained deep learning model. Then, using the improved AoI queue analysis led by simulation findings, determine the ideal arrival rate $lambda$ that minimizes the AoI.

AoI specifies how much time has passed since the most recent update. The update $i$ is sent through the system and eventually arrives at the monitor when a source creates update $i$ with timestamp $u_i(t)$. The monitor observes that its most recent update was received at time $t$ and was time-stamped by $u_i(t)$ with age $t - u_i(t)$. The term "AoI" in this work refers to the average AoI, which is assessed graphically using the sawtooth age waveform $\Delta(t)$, demonstrated in Fig. 2.

$$\langle\Delta\rangle_\tau = \frac{1}{\tau}\int_0^\tau \Delta(t)dt \qquad (1)$$

suppose $\tau$ is large enough. For the $n$th update, the interarrival and system time are represented by $Y_n = t_n - t_{n-1}$ and $T_n = t_n^{'} - t_n$, respectively. According to Fig. 1, the sum of each shaded area

$$Q_n = \frac{1}{2}(T_n + Y_n)^2 - \frac{1}{2}T_n^2 \quad (2)$$

equal to the integral denoted by (1). The average AoI is calculated using

$$\Delta = \lim_{\tau \to \infty} \langle \Delta \rangle_\tau = \frac{E[Q_n]}{E[Y_n]}. \quad (3)$$

### *B. Distributed Coordination Function*

In the IEEE 802.11-based WLAN standard (Wi-Fi), the distributed coordination function (DCF) is a layout for preventing collisions. It is a carrier-sense multiple access with collision avoidance (CSMA/CA) sublayer technology for medium access control (MAC). To address the hidden-terminal issue, a supplementary but commonly-adopted method known as request-to-send/clear-to-send (RTS/CTS) is used. Fig. 3 shows the RTS/CTS mechanism of IEEE 802.11 MAC layer protocol. The station changes to a backoff stage once the channel senses idle for a period greater than DCF Interframe Space (DIFS). The backoff stage time is slotted and uniformly selected from $[0, CW - 1]$, where $CW$ is the size of the contention window. If an empty slot is detected, the backoff counter will drop by one, while a busy channel will cause it to freeze. The frozen backoff counter will continue when the channel is idle for more than a DIFS. Once the backoff counter is zero, the node would transmit an RTS at the start of the subsequent slot and would then receive a CTS following a Short Interframe Space (SIFS). The node may then begin transmitting at the start of the next slot after receiving the CTS for SIFS. An acknowledgment (ACK) will be sent to the sender when the transmission is complete for SIFS, confirming receipt of the data frame. When the CTS timeout expires and the sender does not receive the CTS, the RTS is said to have collided. Therefore, a retransmission will be set with a new contention window size that is twice as large, up to $CWmax$. The sender will discard the data frame after the maximum number of retransmissions has been met.

### *C. AoI in Single-Source Single-Server Queues*

We consider the WiFi device produces data updates and sends through a MAC queue of a single-source single-server type. Update packets created by the top layer are typically put into the MAC queue and await service by the IEEE 802.11 radio. The packet service time is the amount of time between the beginning of a packet's channel contention and the dequeuing event, which may be caused by a successful transmission or a failed transmission that is discarded if it meets the maximum retransmission limit. In queueing theory, the offered load of a queue is $\rho = \lambda/\mu$, where $\lambda$ is the arrival rate and $\mu$ is the service rate. Assuming a queue with constant and fixed $\mu$, updates with high $\lambda$ will contain detailed observations but may congest the queue, resulting in poor throughput and a high system time $E[T]$. However, a low $\lambda$ can eliminate the congestion but increase the interarrival $E[Y]$. As indicated in equations (2,3), we need to research an optimal point of $\lambda$ for minimizing average AoI.

From the viewpoint of a single node, adjusting the $lambda$ and measuring the AoI directly is not feasible since it takes a long time for the AoI to converge to its minimal value. According to the discussion in [1], if a queue is an M/M/1 queue, the ideal point for minimum average AoI is $\rho^* \approx 0.53$. For the M/D/1 and D/M/1 queues, the $\rho*$ value is slightly different. This $\rho*$ offers theoretical guidance for AoI minimization. Once the device knows the queue's service rate, it can adjust its arrival rate $\lambda$ to approach this target $\rho*$. The monitoring of the queue to determine the service rate, however, consumes a lot of node resources. Additionally, the medium access's randomization introduces unpredictable fluctuations in the assessment of service time. The conditional collision probability of the node is a method offered by [13] for computing the expectation of $\mu$. However, the service rate would also vary as a result of changing the update rate. It is challenging to attain $\rho*$ in a short time because of the intricate interactions between the background node traffic, the single test node $\lambda$, and the computed $\mu$. We will outline our approach to quick and precise arrival rate adjustment in the following section.

## III. Age of Information Optimization

This part shows our deep learning-facilitated optimization approach for minimizing AoI by determining the appropriate data update rate. The PGF approach is then used to compute the average frame service time at layer 2. Specifically, we define a transmission success probability over all background devices to precisely determine the expected service time. Our technique can handle generic, heterogeneous background traffic at a meager cost.

### *A. Deep Learning Facilitated Optimization Framework*

Fig. 4 illustrates the proposed deep learning facilitated optimization framework. In this WiFi network, multiple devices coupled with the AP are deemed as background nodes. Moreover, a labeled device is of our interest to AoI minimization. Our framework requires the AP to collect the number of received data frames from each device. After entering the WiFi network, the labeled DEV would make an inquiry request to the AP inquiring about the network's current statistics; specifically, the rate arrived at the AP from each device in the unit of frames per second. Secondly, the labeled DEV inputs the network's current statistics into the deep learning model and fetches the transmission success probability over all background devices denoted by $P_{suc}$ and the conditional collision probability of the labeled DEV, denoted by $P_c$. The labeled DEV may easily obtain the mean of the MAC layer service rate $\mu$ using the MAC service time calculation methods described in [13] and substituting our newly defined $P_{suc}$ for the one in [13]. The labeled DEV's last step is to begin transmitting data updates at the rate $\lambda = \mu \times \rho^*$. Thus, it is anticipated that the updates will have a minimum average AoI. In addition, the network's statistics inquiry should be performed periodically and use the

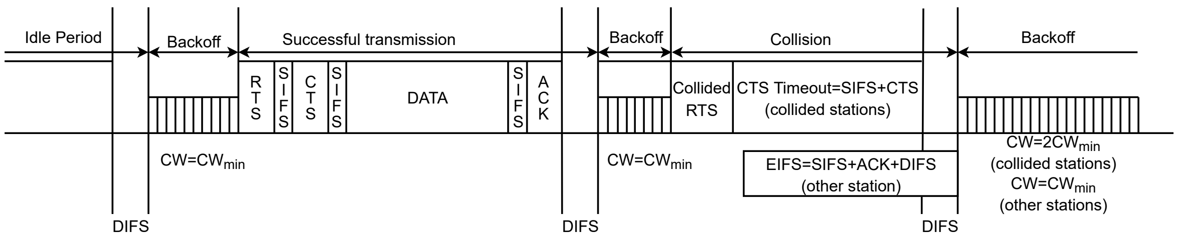


Fig. 3: RTS/CTS mechanism of IEEE 802.11.

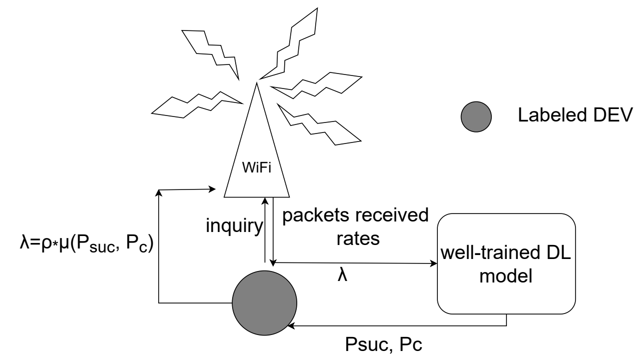


Fig. 4: The proposed deep learning facilitated AoI optimization framework.

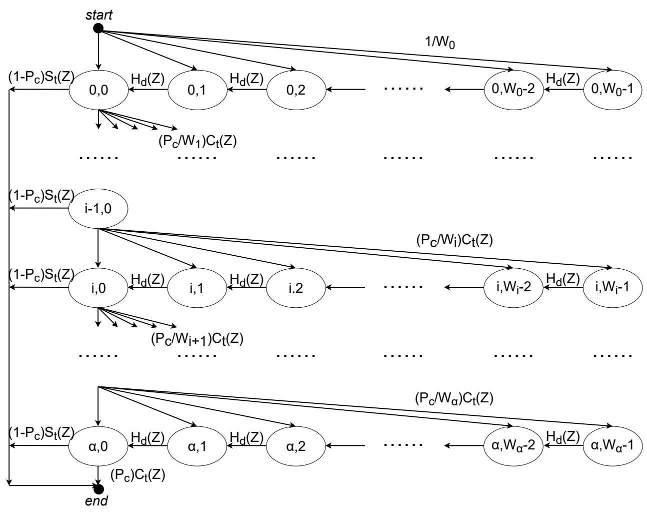


Fig. 5: Generalized state transition diagram for transmission process. [13]

newest channel information to adaptively adjust the DEV's traffic arrival rate for a long-term optimal AoI.

## B. MAC Queue Service Rate

A first-come, first-serve (FCFS) queue's MAC layer service time measures the time between when a packet moves to the front of the queue and starts to compete for medium access and when an acknowledgment is received, or when the packet is discarded because it has exceeded the allowed number of retransmissions. Obtaining the service rate by measurement requires precise timing and constant MAC queue monitoring, which uses a significant amount of the mobile device's resources and time. Using the PGF analysis and Markov chain analysis of 802.11 DCF, [13] offered a technique for computing service time based simply on the observed collision probability.

According to the IEEE 802.11 DCF, Fig. 5 redraws the generalized state transition diagram for the transmission process. The label of each branch in the generalized state transition diagram is produced by multiplying the state transition probability by the state transition duration, which is given as an exponent of the Z variable. Using the well-known Mason formula [14], it is possible to get the probability generating function of the total transition time from the generalized state transition diagram.

Let $W$ stand for the size of the initial contention window, $alpha$ for the retransmission limit, and $m$ for the maximum backoff stage. The successful transmission time and collision time, which can be easily calculated in accordance with the 802.11 MAC standard, are denoted by the variables $T_{suc}$ and $T_{col}$, respectively. And their corresponding PGF equal $Z^{T_{suc}}$ and $Z^{T_{col}}$. These formulas used to calculate the PDF of the data frame's service time, abbreviated as B(Z), are as follows [13]:

$$HW_i(Z) = \begin{cases} \sum_{j=0}^{2^j W-1} \frac{(H_d(Z))^j}{2^i W}, & (0 \le i \le m) \\ HW_m(Z), & (m < i \le \alpha) \end{cases} \tag{4}$$

$$H_i(Z) = \prod_{j=0}^{i} HW_j(Z), \qquad (0 \le i \le \alpha) \tag{5}$$

$$\begin{aligned} B(Z) =& (1-P_c)S_t(Z)\sum_{i=0}^{\alpha}(P_cC_t(Z))^i H_i(Z) \\ &+ (P_cC_t(Z))^{\alpha+1}H_\alpha(Z) \end{aligned} \tag{6}$$

Thus, the mean of the service time $E[T_S]$ can be obtained by differentiation:

$$E[T_S] = \frac{dB(Z)}{dZ}|_{Z=1} = \mu^{-1}. \tag{7}$$

In the calculation shown above, the term $H_d(Z)$ stands for the PGF of the random time needed for the backoff counter to decrease by one. We discovered that the referenced study in [13] only addresses the case with homogeneous background devices. They simplify $H_d(Z)$ that each background device updates at the same data arrival rate. Obviously, the existing solution is insufficient for practical demand. The following subsection defines a more general $H_d(Z)$ to handle an accurate service rate calculation in arbitrary heterogeneous WiFi networks.

### *C. Accurate $\mu$ in Heterogeneous WiFi Networks*

From [13], the signal transfer function of the decrement process of the backoff timer's generalized state transition diagram is

$$H_d(Z) = \frac{(1-P_c)Z^{\sigma}}{[1 - P_{suc}S_t(Z) - (P_c - P_{suc})C_t(Z)]}, \tag{8}$$

where $\sigma$ denotes the empty slot time.

In a WiFi network containing $n$ users, the labeled DEV's conditional probability $P_c$ is equivalent to the probability that at least one user among other $n-1$ is transmitting data frames. As a result, after $\sigma$, the backoff timer of the labeled DEV has a chance of $1-Pc$ to decrease by 1. $P_{suc}$ indicates the probability that given the labeled DEV does not transmit, there is one and only one user among all the background nodes successfully transmitting a data frame. Consequently, $P_{suc}S_t(Z)$ denotes that the labeled DEV's backoff timer has a $P_{suc}$ probability of remaining in its initial state for $T_{suc}$ time as a result of a successful background node transmission. Similarly, $(P_c - P_{suc})C_t(Z)$ represents that the labeled DEV's backoff timer has a $(P_c - P_{suc})$ probability of remaining in its initial state for $T_{col}$ time as a result of a collided transmission.

In a WiFi network containing wireless users with heterogeneous transmission traffic, mathematically expressing $P_{suc}$ becomes infeasible due to the random essence of the IEEE 802.11 protocol. However, for each practical case, we can get $P_{suc}$ via traffic monitoring on the AP side. To calculate the exact value of $P_{suc}$, in a certain testing period, AP should count the number of all the successful transmissions from the background devices, the number of idle slots where nobody transmits, the number of successfully received data transmission virtual slots from all devices, and the number of collided slots, denoted as $N_{otherSuc}$, $N_{noNodeTx}$, $N_{allSuc}$, and $N_{colSlots}$, respectively. Thus, by definition,

$$P_{suc} = \frac{N_{otherSuc}}{N_{noNodeTx} + N_{allSuc} + N_{colSlots}}. \tag{9}$$

Also, the $P_c$ is measured at the labeled DEV's side using

$$P_c = \frac{N_{colRTS}}{N_{allRTS}}. \tag{10}$$

Recall that the above measurement regarding $P_{suc}$ and $P_c$ needs to be implemented every time, which is still not feasible since 1) the measurement increases computational and transmission overheads and 2) these two characteristics are subject to changes during this slow procedure. These disadvantages impel the development of an efficient and fast evaluation strategy. In recent years, a tremendous amount of research has been conducted on machine learning (ML) to learn from a large number of previous occurrences to generate intelligent judgments or accurate predictions in networking problems [15]. Therefore, the deep learning technique fits well for solving this complex problem. Once we establish a fine-tuned deep learning model that is able to output $P_{suc}$ and $P_c$ close enough to the actual values according to the current channel condition, an accurate MAC queue's service rate can be obtained.

### *D. Accurate Channel Estimation Using Deep Learning*

As defined in the contention-based IEEE 802.11 protocol, the channel access for every WiFi user is random. To avoid the aforementioned disadvantages of $P_{suc}$ and $P_c$ measurements, a well-trained deep learning model would be a good option. We want to put the training process offline and then directly use the model as a black box for fast and precise prediction of the current channel condition reflected by $P_{suc}$ and $P_c$.

The factors influencing the $P_{suc}$ and $P_c$ should be listed as candidates for ML input. In the WiFi network, the traffic generating rate of each node is the key. However, collecting the traffic generating rate in a practical WiFi environment is challenging. It may require even more communication overhead between the labeled DEV and each background device. Also, the background nodes may not be willing to share due to certain privacy concerns or extra burdens.

An alternative solution is to use each device's traffic received rate at AP. This method would only require the deployment of an AP featuring counting the incoming traffic and answering the inquiry from the labeled user. Also, each device's traffic received rate at AP contains information about the arrival rate in each device's MAC queue, the overall channel utilization, and the interplay between all the WiFi users. Let the $R_i$ denote the $i^{th}$ background user in the WiFi network, where $i \in [0, N]$, $N$ is the maximum number of background nodes considered in our model, which is equivalent to the total number of background inputs. Besides, we define one more input as $R_A$, which is the packet arrival rate of the labeled DEV. Thus, the input vector of the DL model is of the form: $(R_1, R_2, \cdots, R_i \cdots, R_N, R_A)$. The outputs of the DL model contains $P_{suc}$ and $P_c$.

The training dataset $D_{train} \triangleq \{(\mathbf{x^{(i)}}, \mathbf{y^{(i)}})\}_i$ captures the optimization results of the past problem instances collected through simulation. $\mathbf{x^{(i)}}$ represents the input vector $i$, and $\mathbf{y^{(i)}}$ is the ground truth of the output vector values $(P_{suc}^{(i)}, P_c^{(i)})$. A multilayer perceptron (MLP) is trained over training set $D_{train}$ to learn a reasonable mapping from $\mathbf{x^{(i)}}$ to $\mathbf{y^{(i)}}$. When a new problem instance $\mathbf{x^{(j)}}$ was input to the trained MLP model, the output vector $\hat{\mathbf{y}}^{\mathbf{(j)}}$ indicates the predicted probability $P_{suc}^{(j)}$ and $P_c^{(j)}$.

During the training process, the parameters of the MLP are updated iteratively via a supervised learning technique called backpropagation (BP), with the goal of minimizing a loss function that quantifies the discrepancy between the predicted output and the ground truth. Since the target $\mathbf{y^{(i)}}$ is a continuous variable that is always in the range $[0, 1]$, the probability prediction is a classic regression task. In this way, we choose mean absolute error (MAE) to be the loss function, i.e.,

$$\mathcal{L}(\hat{\mathbf{y}}^{(\mathbf{i})}, \mathbf{y}^{(\mathbf{i})}) = \frac{\sum_{i=1}^{n} \left|\hat{\mathbf{y}}^{(\mathbf{i})} - \mathbf{y}^{(\mathbf{i})}\right|}{n}, \tag{11}$$

where $n$ is the total number of instances in our training dataset. The training process is to solve a optimization problem

$$minimize \quad \mathcal{L}(\hat{\mathbf{y}}^{(\mathbf{i})}(\boldsymbol{\theta}), \mathbf{y}^{(\mathbf{i})}) \tag{12}$$

$$\hat{\mathbf{y}}^{(\mathbf{i})}(\boldsymbol{\theta}) \triangleq \phi(\mathbf{x}^{(\mathbf{i})}; \boldsymbol{\theta}), \tag{13}$$

where $\hat{\mathbf{y}}^{(\mathbf{i})}$ is the output produced by DL model $\phi$ parameterized by $\boldsymbol{\theta}$ according to the input $\mathbf{x}^{(\mathbf{i})}$. An ideal DL model should be very well-trained with a minimal loss across all the instances.

### E. AoI-minimizing offered load $\rho^*$

Assuming that we already know an accurate service rate $\mu$ of the MAC queue, we still need to find a target offered load $\rho^*$ resulting in a minimum average AoI. Theoretical results given in [1] indicate that the optimal $\rho^*$ of a first-come-first-serve M/M/1, M/D/1, and D/M/1 queue is around 0.53, 0.625, and 0.48, respectively. In the simulation, we first apply these rules and also do a wide range search with different offered load $\rho$. The results suggest a down-shifting between the actual optimal point and the theoretical ones. Detailed results and more discussion will be in the next section.

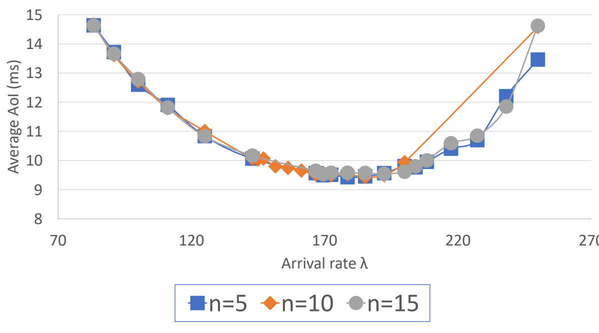
## IV. Numerical Results

This section presents our simulation setup using the discrete-event network simulator NS-3. Simulation results indicate optimal sending rates given specific total background traffic. Also, the DL model is trained using an existing software framework Pytorch in Python. The details of the MLP training process are demonstrated as valuable guidance to the community. Finally, we show the excellent performance of our well-trained DL model.

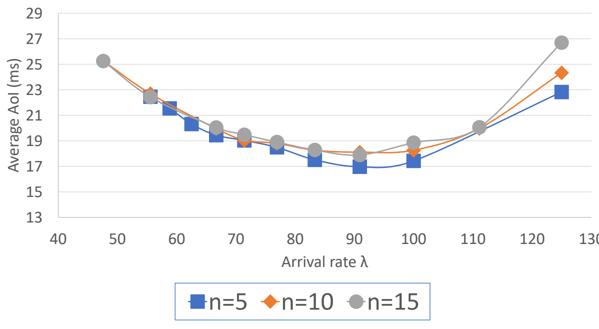
### A. AoI over WiFi Simulations Setup

The simulations construct a WiFi network following the MAC protocol's IEEE 802.11b distributed coordination function. We adopt the RTS/CTS scheme to eliminate uncertain influence caused by hidden terminal problems. Also, to make to network analysis clean and neat, all the WiFi users are sending via UDP protocol. The detailed configurations are shown in Table I.

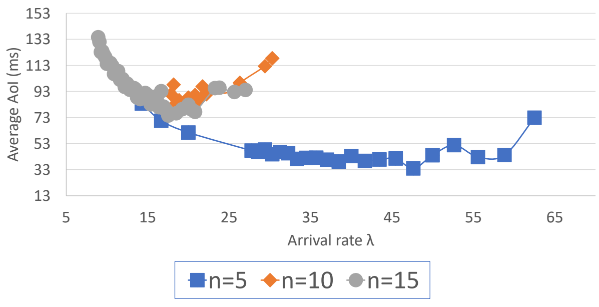
We consider the WiFi network to be within a circle of 80 meters radius. The WiFi users are randomly distributed at the edge of the area. The AP is allocated at the center of the circle. The number of background users ranges from 5 to 15, each equipped with a traffic generator following constant bit rate or Poisson distribution. Each successful data transmission only contains 1 data frame. All the users' MAC queues have infinite buffer sizes. We run each simulation for 300 seconds and disable the default ARP table update to eliminate any effect on AoI calculation.



(a) Total backround traffic $\lambda = 150$



(b) Total backround traffic $\lambda = 300$



(c) Total backround traffic $\lambda = 450$

Fig. 6: Labeled DEV's AoI vs. $\lambda$ in M/M/1 queue

### B. Investigation for the Optimal $\rho^*$

Although the literature has offered ideal offered loads $\rho^*$ for various types of FCFS MAC queues, we still do a wide range of tests to verify these suggested values through simulation results, as the practical values always deviate from the theoretical ones.

We show our simulation results with the tests containing 5, 10, and 15 background nodes, respectively. The traffic model of each user is regulated as needed to construct the labeled DEV's queue types as M/M/1, M/D/1, and D/M/1. Specifically, we define three levels of total traffic arrival rates, 150/sec, 300/sec, and 450/sec, respectively, and use them to construct different network environments that reflect the various extent of the busyness.

*a) Optimal $\rho^*$ for M/M/1 queue:* In Fig. 6, for an M/M/1 queue, WiFi scenarios with total background traffic arrival rate equaling $150/sec$ and 5, 10, and 15 background nodes have light traffic loads, with $P_c < 2\%$. As a result, the networks stay stable with strict convex AoI versus $\lambda$ curves. The optimal $\rho^* \approx 0.48$, which is slightly lower than the theoretical value of 0.53.

For an M/M/1 queue, WiFi scenarios with total background traffic arrival rate equaling $300/sec$ and 5, 10, and 15 background nodes have moderate traffic loads, with $P_c < 7\%$. The overall WiFi traffic is relatively steady with strict convex AoI versus $\lambda$ curves. All these three scenarios reach minimal average AoI at $\lambda = 90.909$. The optimal $\rho^* \approx 0.33$, which is much lower than the theoretical value of 0.53. However, the average AoI obtained following the suggested theoretical value of 0.53 is very close to the minimal average AoI we obtained via simulation.

For an M/M/1 queue, WiFi scenarios with total background traffic arrival rate equaling $450/sec$ and 5 background nodes have relatively high $P_c \approx 17\%$. The minimal AoI is achieved when $\rho^* \approx 0.45$. The other two scenarios with 10, and 15 background nodes have heavy traffic loads. The networks are in poor condition with high $P_c \approx 24\%$. The AoI versus $\lambda$ curves exhibit severe fluctuations. The AoI stays at a relatively low level with $\rho^*$ ranging from 26.3% to 41.3%.

Overall, for the M/M/1 queue, the actual $\rho^*$ may be slightly lower than the theoretical 0.53, but the corresponding AoIs are very close. We suggest taking $\rho^* = 0.48$ when $P_c < 20\%$ and $\rho^* = 0.413$ when $P_c > 20\%$.

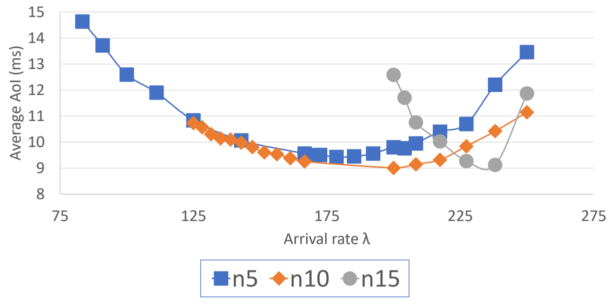
*b) Optimal $\rho^*$ for M/D/1 queue:* The DEV's AoI versus $\lambda$ obtained via simulation are shown in Fig. 7. For an M/D/1 queue, WiFi scenarios with total traffic arrival rates equaling $150/sec$ and 5, 10, and 15 background nodes have light traffic loads, with $P_c < 2\%$. The networks stay stable with strict convex AoI versus $\lambda$ curves. The optimal $\rho^*$ varies in these three scenarios. $\rho^* = 0.466, 0.532, 0.625$ for $n = 5, 10, 15$, respectively. However, the AoIs' difference is minor.

For an M/D/1 queue, WiFi scenarios with total traffic arrival rates equaling $300/sec$ and 5, 10, and 15 background nodes have moderate traffic loads, with $P_c < 7\%$. The networks still stay at a stable level with strict convex AoI versus $\lambda$ curves. The optimal $\rho^* \approx 0.4$ for all three scenarios is lower than the theoretical value of 0.625.

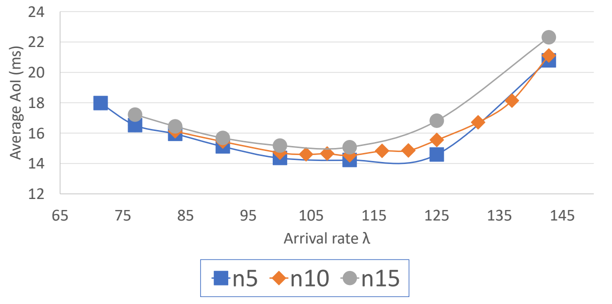
For an M/D/1 queue, WiFi scenarios with total traffic arrival rates equaling $450/sec$ and 5 background nodes have relatively high $P_c \approx 17\%$. The other two scenarios with 10 and 15 background nodes have heavy traffic loads. The networks are in poor condition with high $P_c$. The AoI versus $\lambda$ curves exhibit severe fluctuations. The AoI stays relatively low with $\rho^* \approx 18\%$.

In conclusion, it is rational to adjust the offered load to be 0.625 for a network with low traffic and lower the offered load to 0.4 in a moderately congested network.

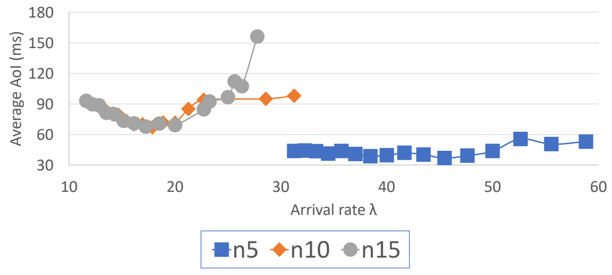
*c) Optimal $\rho^*$ for D/M/1 queue:* Finally, we investigate the ideal offered load in the D/M/1 queue. The results in Fig. 8 show that, with total traffic arrival rates equaling $150/sec$ and 5, 10, and 15 background nodes, the labeled users perform very similarly with optimal $\rho^* = 0.48$, which perfectly matches the theoretical value.



(a) Total backround traffic $\lambda = 150$



(b) Total backround traffic $\lambda = 300$



(c) Total backround traffic $\lambda = 450$

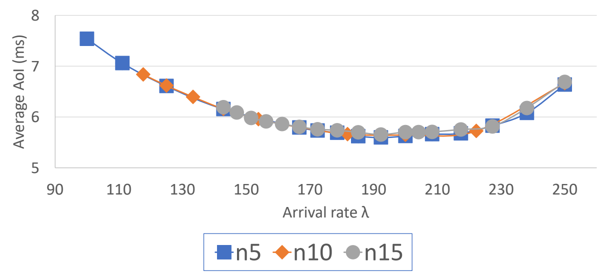
Fig. 7: Labeled DEV's AoI vs. $\lambda$ in M/D/1 queue

The WiFi networks with total traffic arrival rates equaling $300/sec$ and 5, 10, and 15 background nodes have moderate traffic loads. The networks still stay at a stable level with strict convex AoI versus $\lambda$ curves. The optimal $\rho^* \approx 0.33$, which is slightly lower than the theoretical value.

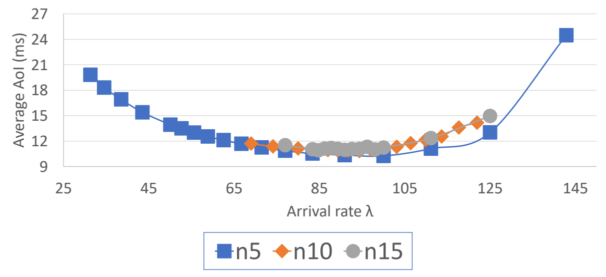
The scenarios with $450/sec$ and 5, 10, and 15 background nodes have heavy traffic loads. The networks are in poor condition with high $P_c$. Like in other queue models, the AoI versus $\lambda$ curves exhibit severe fluctuations. In this situation, it is challenging to determine the optimal $\rho^*$. We suggest taking $\rho$ between 0.25 to 0.5.

## C. Deep Learning Training Process

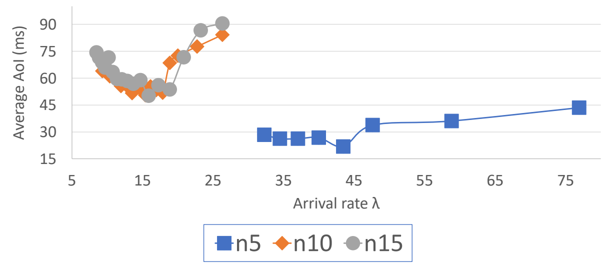
In our WiFi setting, we set the maximum number of background nodes to be 15. Thus, to capture all the possible network scenarios, our input vector size is 16 of the form



(a) Total backround traffic $\lambda = 150$



(b) Total backround traffic $\lambda = 300$



(c) Total backround traffic $\lambda = 450$

Fig. 8: Labeled DEV's AoI vs. $\lambda$ in D/M/1 queue

$(R_1, R_2, \cdots, R_i \cdots, R_{15}, R_A)$. The DL's output contains $P_{suc}$ and $P_c$.

We make great efforts conducting 2390 simulations, extracting the packets received rates at the AP's side, and measuring the actual $P_{suc}$ and $P_c$ for each experiment. We test the WiFi network with $N \in [5, 15]$ in our simulation. All the users' traffic varies case by case. These instances' inputs size is different. To keep the input vector's size constant, we zero-pad the corresponding input value for those scenarios containing less than 15 background users.

The deep learning neural network we use is a fully connected multilayer perceptron (ML). We train the neural network by tuning the hyperparameters to get a small enough loss. Over hundreds of attempts, we successfully narrow down the loss with the parameters listed in Table II.

In Fig. 9, we show the training progress with the MAE vs. training epochs, demonstrating that our trained deep learning can predict the $P_{suc}$ and $P_c$ precisely with MAE as small as $6 \times 10^{-3}$. Moreover, the loss drops sharply in the initial phase of the training. We believe the performance of our deep learning

TABLE I: IEEE 802.11b CONFIGURATIONS

| | |
|---|---|
| Bit rate for DATA frame | 11 Mbps |
| Bit rate for RTS/CTS/ACK frame | 1 Mbps |
| Bit rate for PLCP & Preamble | 1 Mbps |
| Slot time | 20 $\mu s$ |
| DIFS | 50 $\mu s$ |
| SIFS | 10 $\mu s$ |
| PHY header | 192 bits |
| MAC header | 224 bits |
| IP header | 160 bits |
| DATA frame | 8000 bits + PHY header + MAC header |
| RTS | 160 bits + PHY header |
| CTS, ACK | 112 bits + PHY header |
| Initial contention window size | 31 |
| Maximum backoff stages | 5 |
| Maximum retransmission limit | 7 |

TABLE II: DEEP LEARNING PARAMETERS

| | |
|---|---|
| # Input | 16 |
| # output | 2 |
| # Hidden layers | 11 |
| # Neurons per hidden layer | 512 |
| # Epochs | 1500 |
| Dataset size | 2390 |
| Train, test size | 2151, 239 |
| Optimizing algorithm | Stochastic gradient descent |
| Learning rate | 0.0006 |
| Momentum | 0.5 |
| Loss function | Mean absolute error (MAE) |
| Activation function | Rectified Linear Unit (ReLU) |

model can be further improved as the data set scales.

## V. RELATED WORK

The concept of Age of Information (AoI) was recently proposed in a seminal work [1], which was inspired by research on vehicle safety messaging over a CSMA network that was started in [16] and [17]. The work indicates the requirement for timely updating is not equivalent to maximizing queue utilization or minimizing the delay. While delivering updates as quickly as feasible can maximize utilization, a monitor will instead receive updates that were delayed due to the communication system backlog. In this situation, lowering the update rate may enhance the timeliness of status information at the receiver. However, slowing down the update rate will also result in a monitor's status information being excessively outdated due to a lack of updates.

Age in elementary queues is a popular research orientation. Paper [2] examines the AoI of the multi-source shared-queue system and applies an analysis technique called stochastic hybrid systems (SHS) [18] to reduce the evaluation complexity. AoI in single-server queues with various service disciplines is demonstrated in [19]. AoI analysis in IoT systems with multiple sources has been investigated in [5] under different preemptive queueing disciplines.

Many studies also focus on the AoI optimization in random access networks. For example, [7] devises age-dependent random access (ADRA) that fairly manages the channel access to everyone. ADRA is claimed to be a distributed protocol but

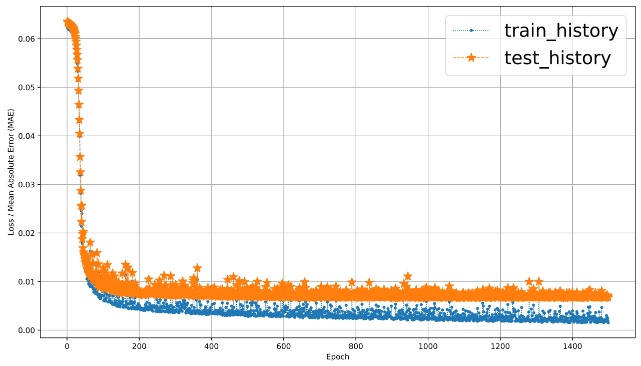


Fig. 9: The MAE vs. Training epochs.

seems centralized to some extent and impotent to meet various users' demands. The AoI performance of a decentralized system specifically under the context of slotted ALOHA is analyzed in [8], [20]. Reference [9], [21]–[25] examine packet management strategies that discard outdated packets and enhance AoI in a variety of operational regimes.

## VI. Conclusion

This paper proposes a solution for a single user with limited computational power to quickly adjust its updating rate for an optimal average AoI under a heterogeneous WiFi network. In particular, we develop a concise yet efficient framework and train a deep learning model to facilitate the AoI minimization process. The simulation results demonstrate that our deep learning model has outstanding prediction performance, and our framework is efficient and accurate in complex WiFi environments.

## VII. Acknowledgment

This work was supported in part by the NSF of USA under Grants CNS-2008092 and CNS-1816908.

## References

[1] S. Kaul, R. Yates, and M. Gruteser, "Real-time status: How often should one update?" in *Proc. of IEEE INFOCOM*, 2012, pp. 2731–2735.
[2] R. D. Yates and S. K. Kaul, "The age of information: Real-time status updating by multiple sources," *IEEE Transactions on Information Theory*, vol. 65, no. 3, pp. 1807–1827, 2019.
[3] A. Kosta, N. Pappas, A. Ephremides, and V. Angelakis, "The age of information in a discrete time queue: Stationary distribution and non-linear age mean analysis," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 5, pp. 1352–1364, 2021.
[4] D. Deng, Z. Chen, Y. Jia, L. Liang, S. Fang, and M. Wang, "Age of information in a multiple stream m/g/1/1 non-preemptive queue," in *Proc. of IEEE International Conference on Communications*, 2021, pp. 1–6.
[5] N. Akar and O. Dogan, "Discrete-time queueing model of age of information with multiple information sources," *IEEE Internet of Things Journal*, vol. 8, no. 19, pp. 14 531–14 542, 2021.
[6] J. Xu and N. Gautam, "Peak age of information in priority queuing systems," *IEEE Transactions on Information Theory*, vol. 67, no. 1, pp. 373–390, 2021.
[7] H. Chen, Y. Gu, and S.-C. Liew, "Age-of-information dependent random access for massive iot networks," in *Proc. of IEEE INFOCOM WORKSHOPS)*, 2020, pp. 930–935.
[8] X. Chen, K. Gatsis, H. Hassani, and S. S. Bidokhti, "Age of information in random access channels," in *Proc. of IEEE International Symposium on Information Theory*, 2020, pp. 1770–1775.
[9] A. Kosta, N. Pappas, A. Ephremides, and V. Angelakis, "Age of information performance of multiaccess strategies with packet management," *Journal of Communications and Networks*, vol. 21, no. 3, pp. 244–255, 2019.
[10] A. Maatouk, M. Assaad, and A. Ephremides, "On the age of information in a csma environment," *IEEE/ACM Transactions on Networking*, vol. 28, no. 2, pp. 818–831, 2020.
[11] N. Lu, B. Ji, and B. Li, "Age-based scheduling: Improving data freshness for wireless real-time traffic," in *Proc. of the ACM International Symposium on Mobile Ad Hoc Networking and Computing*, 2018, p. 191–200.
[12] I. Kadota, A. Sinha, and E. Modiano, "Optimizing age of information in wireless networks with throughput constraints," in *Proc. of IEEE INFOCOM*, 2018, pp. 1844–1852.
[13] H. Zhai, Y. Kwon, and Y. Fang, "Performance analysis of ieee 802.11 mac protocols in wireless lans," *Wireless communications and mobile computing*, vol. 4, no. 8, pp. 917–931, 2004.
[14] D. Choi, "Frame alignment in a digital carrier system-a tutorial," *IEEE Communications Magazine*, vol. 28, no. 2, pp. 47–54, 1990.
[15] Y. Cheng, B. Yin, and S. Zhang, "Deep learning for wireless networking: The next frontier," *IEEE Wireless Communications*, vol. 28, no. 6, pp. 176–183, 2021.
[16] S. Kaul, M. Gruteser, V. Rai, and J. Kenney, "Minimizing age of information in vehicular networks," in *Proc. of 8th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks*, 2011, pp. 350–358.
[17] S. Kaul, R. Yates, and M. Gruteser, "On piggybacking in vehicular networks," in *Proc. of IEEE Global Telecommunications Conference*, 2011, pp. 1–5.
[18] J. P. Hespanha, "Modelling and analysis of stochastic hybrid systems," *IEE Proceedings-Control Theory and Applications*, vol. 153, no. 5, pp. 520–535, 2006.
[19] Y. Inoue, H. Masuyama, T. Takine, and T. Tanaka, "A general formula for the stationary distribution of the age of information and its application to single-server queues," *IEEE Transactions on Information Theory*, vol. 65, no. 12, pp. 8305–8324, 2019.
[20] X. Chen, K. Gatsis, H. Hassani, and S. S. Bidokhti, "Age of information in random access channels," *IEEE Transactions on Information Theory*, pp. 1–1, 2022.
[21] M. Moltafet, M. Leinonen, and M. Codreanu, "Average age of information for a multi-source m/m/1 queueing model with packet management," in *Proc. of IEEE International Symposium on Information Theory*, 2020, pp. 1765–1769.
[22] M. Costa, M. Codreanu, and A. Ephremides, "On the age of information in status update systems with packet management," *IEEE Transactions on Information Theory*, vol. 62, no. 4, pp. 1897–1910, 2016.
[23] P. Zou, O. Ozel, and S. Subramaniam, "Waiting before serving: A companion to packet management in status update systems," *IEEE Transactions on Information Theory*, vol. 66, no. 6, pp. 3864–3877, 2020.
[24] M. Moltafet, M. Leinonen, and M. Codreanu, "Average aoi in multi-source systems with source-aware packet management," *IEEE Transactions on Communications*, vol. 69, no. 2, pp. 1121–1133, 2021.
[25] M. Costa, M. Codreanu, and A. Ephremides, "Age of information with packet management," in *Proc. of IEEE International Symposium on Information Theory*, 2014, pp. 1583–1587.