

AI-Driven Media Access Control (MAC) and Link Layer Intelligence

Lecture 5

By Dr. Tarfa Yaseen Hamed

Apr. 2026

Outline

- ▶ Introduction – The Challenges of Layer 2
- ▶ The Shift from Rule-Based to Learning-Based MAC
- ▶ The Problem: The "Exponential Backoff" Collapse
- ▶ Deep Reinforcement Learning for Channel Access
- ▶ The DRL Solution: Intelligent Channel Access
- ▶ Case Study 1: Predictive ARQ (Automatic Repeat Request)
- ▶ Case Study 2: Intelligent Slicing in 5G/6G

Introduction: The Challenges of Layer 2

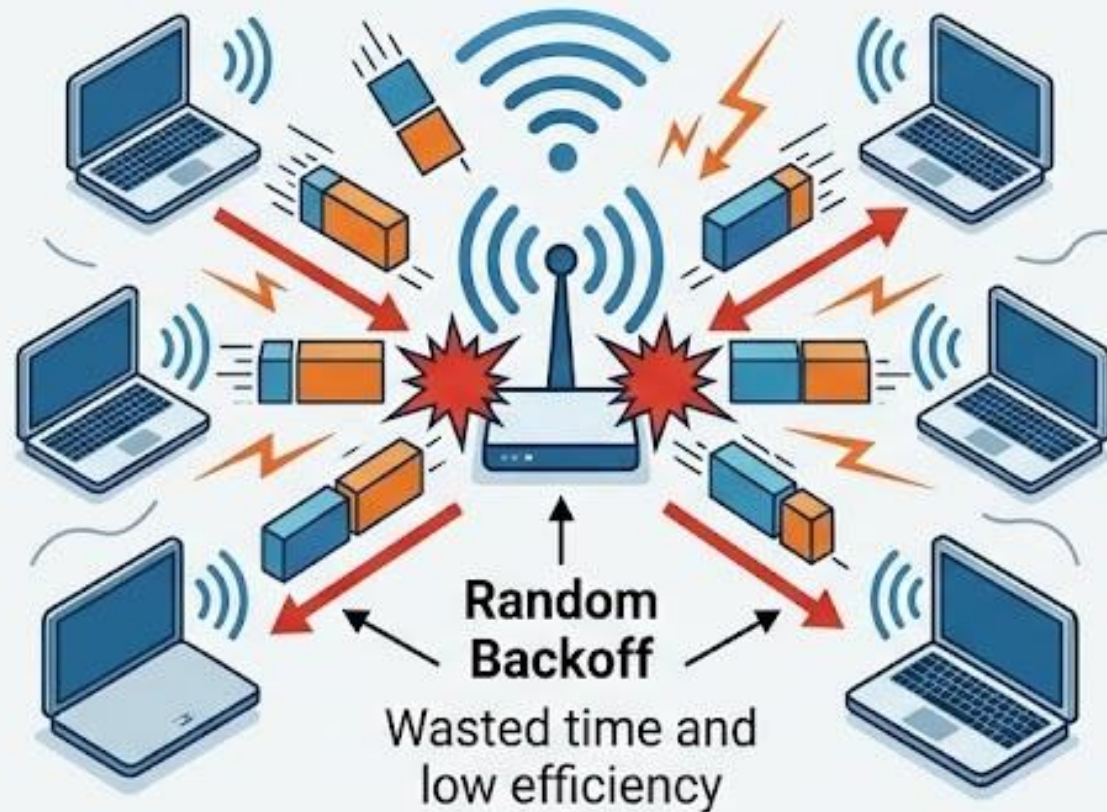
- The Data Link Layer is responsible for moving data between two nodes on the same network.
- Traditional protocols (like CSMA/CA in Wi-Fi or Scheduling in 5G) use "fixed rules."
- The Problem: Fixed rules cannot handle the "Ultra-Dense" environments of 2026, where thousands of IoT devices compete for the same millisecond of airtime.

Where AI fits in:

- **Resource Allocation:** Deciding who talks and when.
- **Error Control:** Predicting packet loss before it happens.
- **Congestion Control:** Managing the "buffer" to prevent latency spikes.

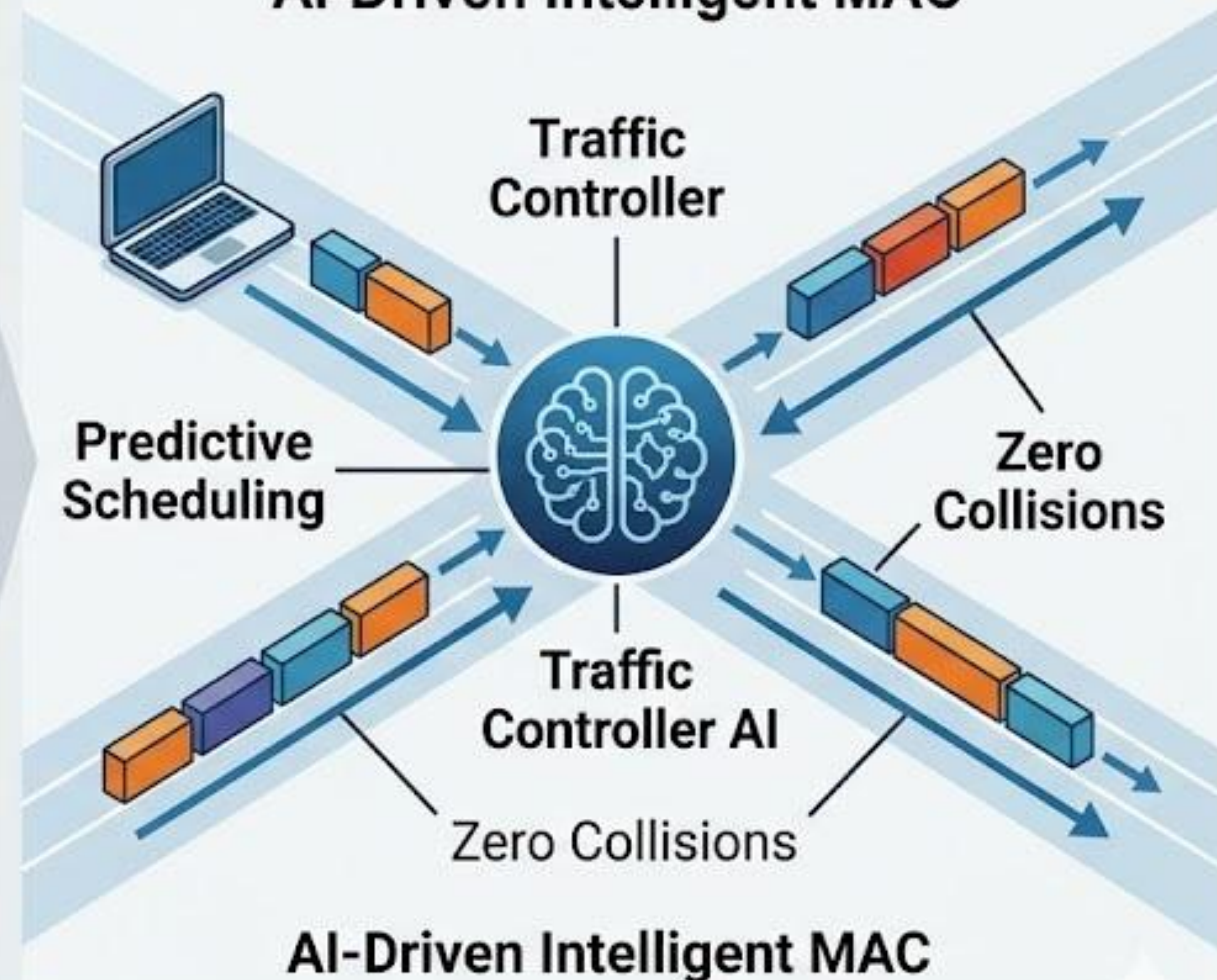
Figure 1: Traditional vs. AI-Driven MAC Layer

Traditional MAC (e.g., Wi-Fi 6 CSMA/CA)



Traditional MAC (e.g., Wi-Fi 6 CSMA/CA)

AI-Driven Intelligent MAC



AI-Driven Intelligent MAC

Figure 1: Traditional vs. AI-Driven MAC Layer

The Shift from Rule-Based to Learning-Based **MAC**

Feature	Traditional MAC (e.g., 802.11ax)	AI-Enabled MAC
Strategy	Fixed Backoff / Request-to-Send	Adaptive Learning (RL)
Input	Current Collision	Historical Traffic Patterns
Efficiency	Drops in High Density	Scales via Prediction

The Problem: The "Exponential Backoff" Collapse

- In traditional Wi-Fi, if two devices collide, they both pick a random "Backoff" time from a Contention Window (CW).
- The Contention Window is a range of random time slots from which a device chooses a waiting time before attempting to transmit data.
- The Rule: If they collide again, they double the window size (31 -> 63 -> 127 ...).
- The Failure: In a stadium or a dense apartment with 200+ devices, this "random guessing" leads to a collision storm.
- Throughput drops to nearly zero because devices spend all their time "backing off" and waiting.

Deep Reinforcement Learning for Channel Access

In modern Wi-Fi and 5G, the most significant breakthrough is replacing the "Random Backoff" (waiting a random time after a collision) with a Deep Q-Network (DQN).

- **The Agent:** The Wi-Fi Access Point (AP).
- **The Action:** Choosing the optimal "Contention Window" size or frequency slice.
- **The Reward:** Total Network Throughput vs. Fair Access for all users.
- **The Result:** AI learns the "rhythm" of the devices, reducing collisions by up to 40% in dense apartment buildings.

The DRL Solution: Intelligent Channel Access

Instead of a fixed mathematical rule, we use a Deep Q-Network (DQN) to control the MAC layer.

- **The State (s):** The agent (Access Point) observes the "Channel Busy" ratio, the number of successful transmissions, and the current collision rate.
- **The Action (α):** The AI dynamically adjusts the Contention Window size or assigns a specific Time Slot to a user. It doesn't just "guess"; it learns the pattern of when users need to talk.
- **The Reward (r):**
$$\text{Reward} = \alpha \cdot \text{Throughput} - \beta \cdot \text{Delay} - \chi \cdot \text{Fairness}.$$
- The AI is "paid" to keep the network fast but also "fined" if it ignores a specific user for too long.

Technical Implementation: "Listen and Learn"

How does the DQN actually "see" the network? It uses a **Sensing Window**.

1. The **AI** monitors the last K time slots.
2. It converts this history into a vector (e.g., [1, 0, 0, 1] where 1 is a collision and 0 is success).
3. The **Neural Network** processes this vector and predicts: "Based on the last 10 milliseconds, User 5 is about to send a large file. I should increase the Contention Window for others to give them a clear path."

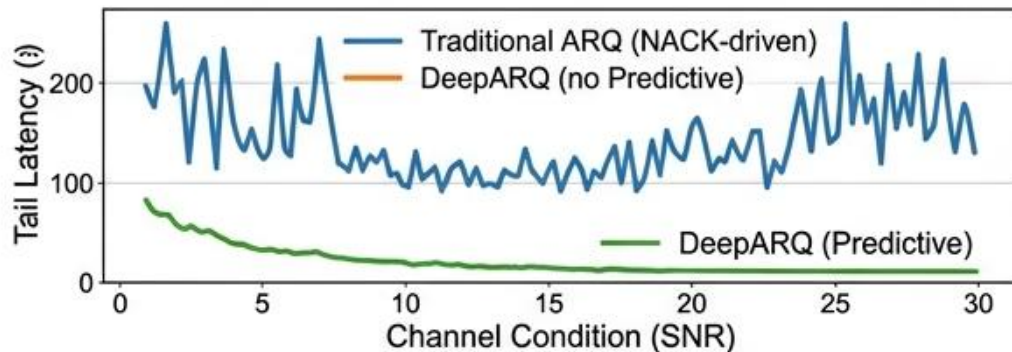
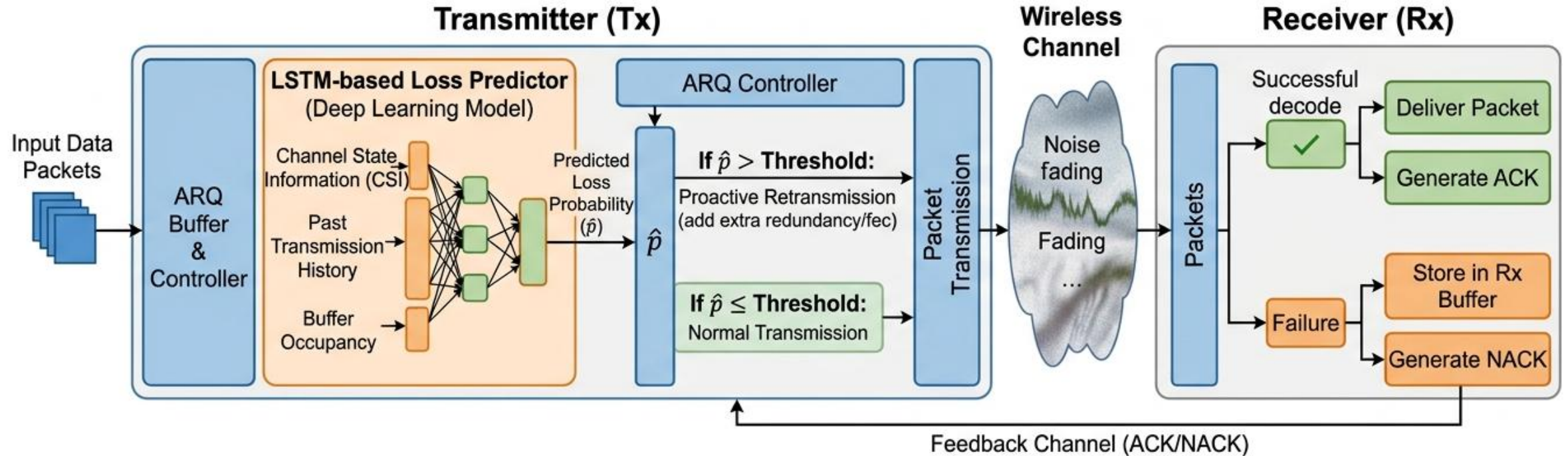
Case Study 1: Predictive ARQ (Automatic Repeat Request)

Traditional DLLs wait for a Negative Acknowledgement to retransmit a lost packet. This causes a "Round Trip Time" (RTT) delay.

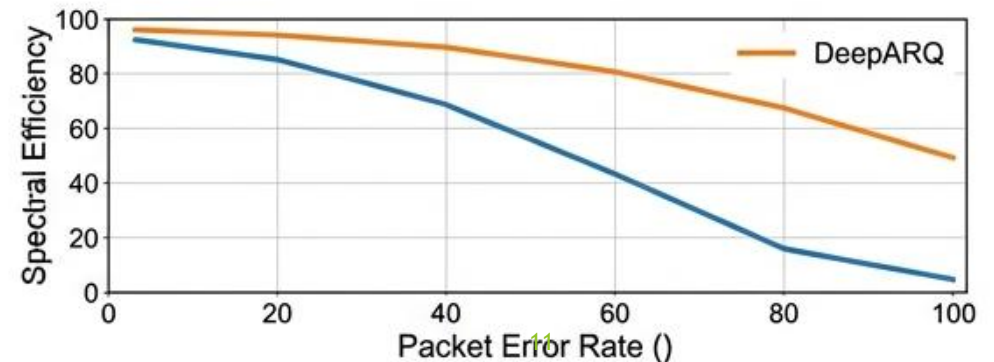
The AI Solution: DeepARQ Using an **LSTM (Long Short-Term Memory)** network, the transmitter predicts the probability of a packet failure *before* the Negative Acknowledgement even arrives.

- **If Probability > 80%:** The system proactively retransmits or adds extra error correction (FEC).
- **The Impact:** Reduces "Tail Latency," which is critical for Cloud Gaming and VR/AR.

DeepARQ: Predictive Error Control for Reduced Latency



Tail Latency (99th percentile) vs. Channel Condition (SNR)



Spectral Efficiency vs. Packet Error Rate

Case Study 2: Intelligent Slicing in 5G/6G

- Network Slicing allows one physical network to act as many virtual ones.
- The MAC layer must decide how much bandwidth to give to an **Ambulance (Ultra-Reliable)** vs. a **Smartphone (Mobile Broadband)**.
- Performance Comparison: Heuristic vs. ML Slicing

Scenario	Traditional (Proportional Fair)	ML-Based (DDPG Agent)
Throughput (Gbps)	1.2	1.8 (+50%)
SLA Violation Rate	5%	< 0.1%
Energy Overhead	Baseline	15% Lower (Smart Sleeping)

Technical Implementation: The "Cross-Layer" Feed

For the DLL to be "smart," it needs information from the PHY layer. This is called **Cross-Layer Design**.

- **Input to AI:** RSSI (Signal Strength), Doppler Spread, and Buffer Occupancy.
- **Output from AI:** Modulation Coding Scheme (MCS) index and Frame Size.