

FINAL PROJECT REPORT

강화학습 기반 대화형 차량 추천 및 스케줄링 시스템

EMNLP "20 Questions" 프레임워크의
실제 자동차 시승 예약 도메인 확장 연구

김준호 (A71023)

서강대학교 AI SW 대학원
데이터사이언스 인공지능학과



GitHub: <https://github.com/jhboyo/drive-booking>



Live Demo: <https://driving-booking.streamlit.app>

Table of Contents

SECTION 01

연구 배경 및 동기

SECTION 03

실험 결과 및 성능 평가

SECTION 02

시스템 설계 및 구조

SECTION 04

결론 및 향후 연구

프로젝트 개요

강화학습 기반 대화형 차량 추천 및 스케줄링 시스템

연구 목표 (Research Goal)

사용자의 불확실한 선호도를 파악하여 최소한의 질문으로 최적 차량을 추천하고 차량 센터 가용성을 고려한 일정 자동 배정 시스템 구현

접근 방법 (Methodology)

Two-Phase Reinforcement Learning

- Q-Learning 기반의 대화형 추천 에이전트
- DQN 기반의 스케줄링 에이전트
- End-to-End 통합 학습 및 Synergy Bonus 설계

주요 기여 (Contribution)

기존 정적 룰 베이스(Rule-based) 예약 시스템의 한계를 극복하고, "20 Questions" 게임 프레임워크를 실제 자동차 비즈니스 도메인으로 확장 적용

연구 동기: "20 Questions" 게임

핵심 참조 논문

본 연구는 Hu et al. (EMNLP 2018)의 "Playing 20 Question Game with Policy-Based Reinforcement Learning"에서 제안된 프레임워크를 기반으로 함.

문제 정의

객체 추론 과정을 순차적 의사결정 문제(Sequential Decision Making)로 정의함. 에이전트는 질문(Action)을 통해 정보 이득(Information Gain)을 최대화하고, 최소한의 턴(Turn)으로 정답을 맞추는 정책(Policy)을 학습함.

주요 학술적 기여

1. Robustness: 사용자의 노이즈 섞인 답변(Noisy Answers)에 강건한 추론 모델 제시
2. Independence: 정적 지식 그래프(Knowledge Graph) 의존성을 제거하고 데이터 기반 학습 수행
3. Efficiency: 질문 효율성을 보상 함수(Reward Function)에 반영하여 최적화

연구 동기: 실제 도메인으로의 확장

대상 도메인

본 연구는 단순 퀴즈 게임 환경을 넘어, 실제 자동차 시승 예약 프로세스에 강화학습을 적용함.
실제 차량 데이터베이스(DB)와 시승 센터의 스케줄링 제약조건을 포함하는 복잡한 업무 환경을 모델링함

다목적 최적화

사용자 경험(UX) 향상을 위한 질문 수 최소화와 비즈니스 성과를 위한 예약 성사율 극대화를 동시에 달성하는 보상 함수(Reward Function)를 설계함.

End-to-End 프레임워크

정보 수집(Information Gathering) → 차량 추천(Recommendation) → 일정 스케줄링(Scheduling)으로 이어지는 전체 파이프라인을 통합 최적화하는 Two-Phase Architecture를 제안함.

문제 정의

1. High Interaction Cost

현재 제공 되는 서비스에서 사용자는 예약 완료까지 평균 5~7단계의 수동 입력 과정을 거쳐야 함. 모델 선택, 옵션 비교, 개인정보 입력 등 반복적인 상호작용이 요구됨

2. Decision Fatigue

수십 개의 차량 모델과 복잡한 트림/옵션 조합 중에서 적합한 차량을 직접 탐색해야 하며, 이는 사용자에게 높은 인지적 부하를 유발함

3. Information Asymmetry

시승 센터별 가용 스케줄이 실시간으로 투명하게 제공되지 않아, 사용자가 일일이 빈 시간대를 확인해야 하는 비효율성 존재

제안 솔루션 : RL Agent

Approach: Two-Phase Optimization

기존의 복잡한 사용자 입력 과정을 강화학습 기반의 자동화 파이프라인으로 대체함. 사용자는 최소한의 질문에 응답하기만 하면, 에이전트가 차량 추천과 일정 배정을 수행함

Core Components

1. Phase 1 (Recommendation): Q-Learning을 통해 정보 이득이 높은 질문을 우선 선택하여, 불필요한 문답을 줄이고 정확한 차량을 추천
2. Phase 2 (Scheduling): DQN을 통해 센터의 가용 슬롯과 사용자 선호도를 매칭, 예약 실패 확률을 최소화

PROCESS EFFICIENCY

User Steps 7 steps → **2 steps**

Decision Type Manual → **AI-Driven**

Avg. Time ~8.5m → **~0.5m**

시스템 아키텍처: Two-Phase Framework

1. 차량 추천(Phase 1)

Q-Learning 기반 에이전트가 고객과 상호작용하며 Hidden Preferences를 추론.
최대 20개였던 질문 수를 4개(필수)로 단축하여 효율성을 극대화함

2. 스케줄링 최적화 (Phase 2)

추천된 차량에 대해 DQN 에이전트가 시승 센터의 Availability Constraints(가용성 제약)와 고객의 선호 시간대를 매칭하여 최적 슬롯을 배정

3. End-to-End Integration

두 단계의 보상을 통합한 Total Reward 함수를 설계. 'Synergy Bonus'를 도입하여 추천 품질과 예약 성사율의 전역 최적점(Global Optimum)을 탐색

데이터와 환경 세팅

Dataset

Vehicle Inventory: 23종 (Sedan, SUV, EV 등)

Attributes: 8개 핵심 속성 (가격, 연비, 용도, 승차인원 등)

기타: 24개의 차량 센터, 총 8개의 질문 목록

Environment

강화학습: Gymnasium(RL 환경 구성)

DQN 신경망: PyTorch

모바일 앱: Streamlit

데이터: Numpy, Pandas

패키지 관리: UV

Phase 1: Vehicle Recommendation

Core Objective

불확실한 상태에서 시작하여, 질문을 통해 엔트로피를 감소시키고 최소한의 턴 (Turn) 내에 최적 차량(Target Item)을 추천하는 정책 학습

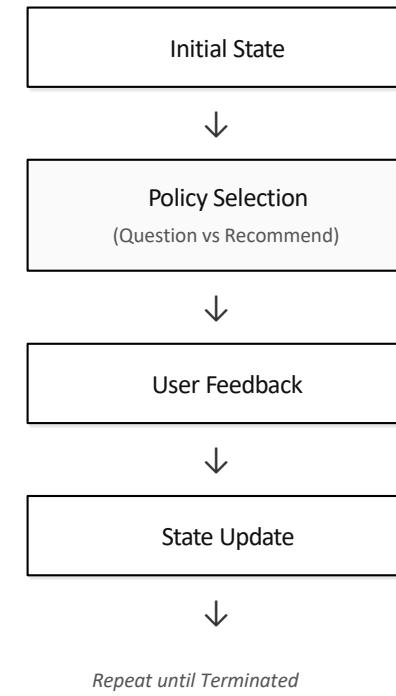
Exploration-Exploitation Trade-off

Exploration: 질문을 통해 State를 구체화하고 정보 이득을 최대화
Exploitation: 충분한 확신 도달 시, 즉시 추천하여 보상 획득

Constraints & Termination

- Max Questions: 8회 (초과 시 Truncated)
- Recommendation: Top1 ~ Top4 추천 시 즉시 Terminated

Episode Flow



Phase 1: State 구성

총 69차원 벡터

Index	Feature	Dim	Description
[0 : 5]	고객 프로필	5	나이, 성별, 내국인, 직장, 관심차량 * 나이 정규화: $(\text{age}-45)/25$, 범주형: ± 1.0 매핑
[5 : 45]	질문 응답 이력	40	8개 질문 \times 5개 선택지 (One-hot) * 미응답 상태 포함, 희소 벡터 (Sparse Vector) 형태
[45]	진행률	1	현재 질문 수 / 최대 허용 질문 수 * 0.0 ~ 1.0 사이 값으로 정규화
[46 : 69]	차량 매칭 점수	23	차량별 실시간 적합도 점수 * Hidden Preference 기반 계산, [-1, 1] 범위

NOTE: 신경망(DQN) 학습의 안정성을 위해 모든 입력값을 [-1.0, 1.0] 범위로 정규화함
특히 차량 매칭 점수를 State에 포함시킴으로써 에이전트가 현재 추천 품질을 인지할 수 있도록 설계

Phase 1: Action 구성

Index	Feature	Description
[0 : 7]	질문 선택	용도, 연료, 가족, 예산, 우선순위, 크기, 차체, 색상
[8 : 11]	차량 추천	1순위, 2순위, 3순위, 대체 차량 추천

Phase 1: Transition & Termination Condition

State Transition (질문 수행 시)

1. 고객 응답 시뮬레이션 및 정보 획득
2. 차량 매칭 점수 업데이트
3. 질문 횟수 증가

Terminal State (추천 수행 시)

바로 추천 선택 시: `terminated = True` 즉시 에피소드 종료 및 최종 보상 계산

Truncated (강제 종료)

최대 질문 수(5회) 초과 시: `truncated = True` 효율성을 강제하기 위한 제약 조건



Reward Function

Step Penalty (Efficiency)

모든 질문 행동에 대해 -0.2의 고정 페널티를 부여. 최소한의 턴(Turn)으로 정보를 수집하도록 유도하는 Time Cost 역할

Logic Constraint (Redundancy)

이미 수행한 질문을 반복할 경우 -0.5의 강한 페널티. State 내의 질문 이력 (History)을 인지하고 회피하도록 학습

Information Gain (Discrimination)

질문 후 후보 차량들의 매칭 점수 분산(Variance) 증가량에 비례한 보상
후보군을 효과적으로 좁히는 유의미한 질문임

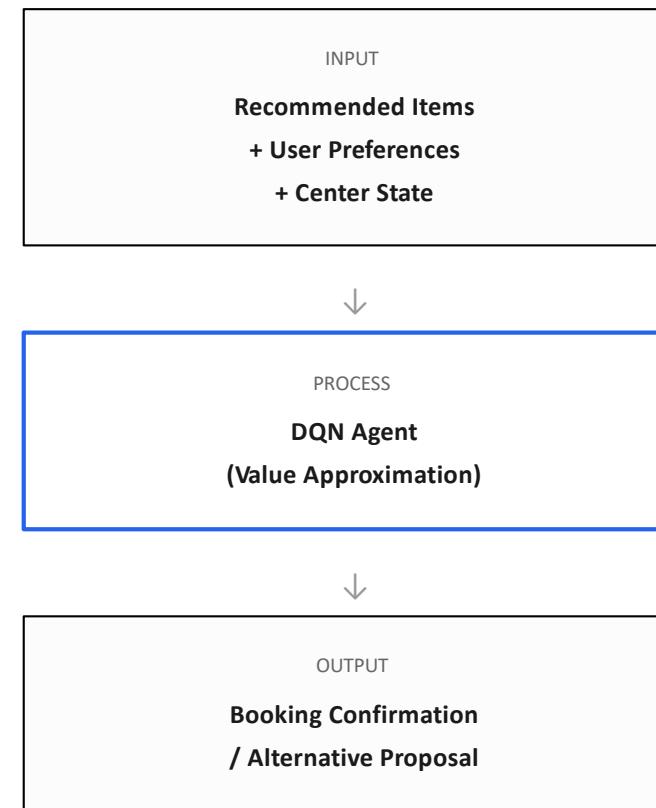
Phase2: Scheduling System

RESEARCH OBJECTIVE

추천된 차량(Item)에 대해 고객의 선호도(Preference)와 센터의 가용성(Availability)을 동시에 만족하는 최적의 시승 슬롯(Slot)을 배정하는 문제

PROBLEM CONTEXT

물리적 자원(차량, 스태프)의 한계 내에서 일정을 배정해야 함
실시간 예약 요청 및 취소로 인해 상태 공간이 지속적으로 변화함
고객 만족(선호 시간) vs 운영 효율(유휴 시간 최소화) 간의 균형 필요



Phase2: State 정의

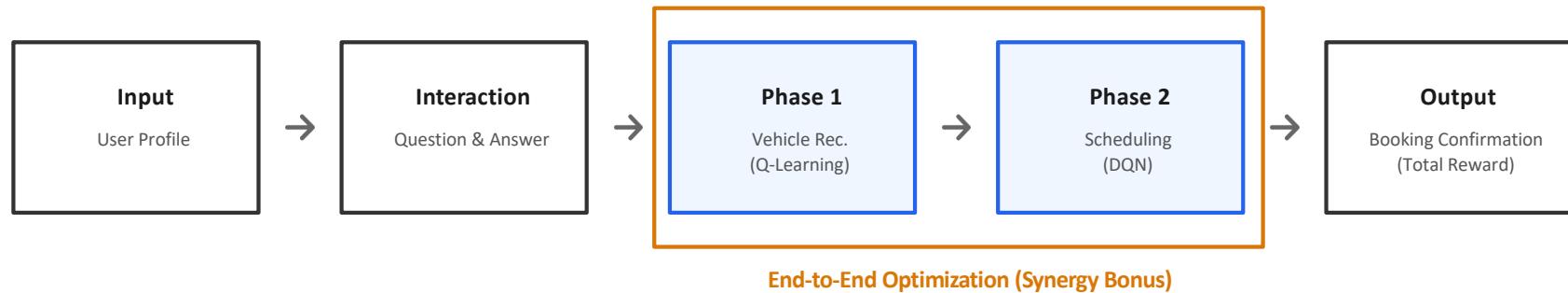
State Component	Variable	Description
추천 차량	recommended_cars	Phase 1에서 도출된 Top-k 추천 차량 목록 및 매칭 점수
고객 선호	preferences	선호 날짜, 시간대, 유연성, 긴급도
센터 가용성	center_state	차량별 보유 수량, 현재 예약 현황 매트릭스, 스태프 가용 여부
컨텍스트	context	현재 시점, 전체 예약 대기열 길이 (부하량)

Phase 2: Action Space

DQN 에이전트는 단순 슬롯 할당(Assignment)이 아닌, 고객과 협상(Negotiation)하는 전략을 학습

Index	Action Name	Description	Strategic Goal
0	예약 확정	1순위 선호 일정으로 즉시 예약 확정	만족도 최적화
1	같은 날 대안	동일 날짜 내 다른 시간대 제안	일정 유지
2	다음 날 대안	다음 날 동일/유사 시간대 제안	시간대 유지
3	평일 대안	주말 대신 평일 저녁(비혼잡) 제안	부하 분산
4	다음 주 대안	일주일 뒤 동일 요일로 연기	장기 대안
5	인센티브	비선호 시간 + 할인/쿠폰 제안	타협 유도

Phase3: Integrated Pipeline Architecture



시스템은 순차적(Sequential)으로 동작하지만, 학습은 통합적(Integrated)으로 이루어짐
Phase 1의 추천 결과(Recommended Vehicle)가 즉시 Phase 2 스케줄러의 입력 상태(State)로 전이 됨
이를 통해 추천 에이전트는 단순히 '좋은 차'가 아니라, '예약 가능한 차'를 추천하도록 학습 됨

Total Reward Function

$$R_{\text{total}} = R_1 + R_2 + R_{\text{synergy}}$$

R₁ Vehicle Recommendation Reward (Phase 1) 고객 만족도(Satisfaction)와 질문 효율성 (Efficiency)의 합. 불필요한 질문에 대한 페널티 포함.

R₂ Scheduling Reward (Phase 2) 예약 성사 여부(Booking Success)와 선호 시간 매칭 (Preference Match). 대기 시간 최소화 목표.

R_{Synergy} Connection Bonus (Bridge) 현재 가능한 차량을 추천했을 때 부여되는 추가 보상. Recommender가 스케줄러의 상태(Availability)를 고려하도록 유도.

프로젝트 실행 방법

1. Online Demo (Recommended)

별도의 설치 과정 없이 웹 브라우저를 통해 즉시 접근 가능
Streamlit Cloud 환경에서 배포되어 있음

URL: <https://drive-booking-rl.streamlit.app/>

2. Local Environment Setup

로컬 실행을 위해서는 Python 3.10+ 환경이 필요
패키지 의존성 관리를 위해 uv를 사용

주요 의존성: gymnasium, torch, streamlit, numpy, pandas

1. Install 'uv' package manager

\$ pip install uv

2. Sync project dependencies

\$ uv sync

3. Run Streamlit application

\$ uv run streamlit run src/app/main.py

Access via browser:

<http://localhost:8501>

실험 설정: Phase1

파라미터	설정값	비고
Episodes	1,000	Evaluation: 100 eps
Learning Rate (α)	0.1	Fixed
Discount Factor (γ)	0.95	Future reward weighting
Exploration (ϵ)	1.0 → 0.05	Decay rate: 0.998
Random Seed	42	Reproducibility

탐험 전략 (Exploration Strategy)

초기에는 무작위 탐색($\epsilon=1.0$)을 수행하여 상태 공간을 넓게 탐색하고, 학습이 진행됨에 따라 Greedy 정책의 비중을 점진적으로 높이는 Epsilon-Greedy Decay 방식 적용

환경 제약 (Constraints)

효율적인 추천을 유도하기 위해 최대 질문 수를 5회로 엄격히 제한하며, 중복 질문 발생 시 페널티(-0.5)를 부여하여 불필요한 상호작용을 억제함

결과 요약: Phase1

KEY OBSERVATION

Q-Learning 에이전트는 효율성 보너스를 극대화하기 위해 "질문 없이 즉시 추천"하는 정책으로 수렴함. 이는 보상 구조상 질문 비용(-0.2)보다 빠른 추천의 이득이 크기 때문임

8.254

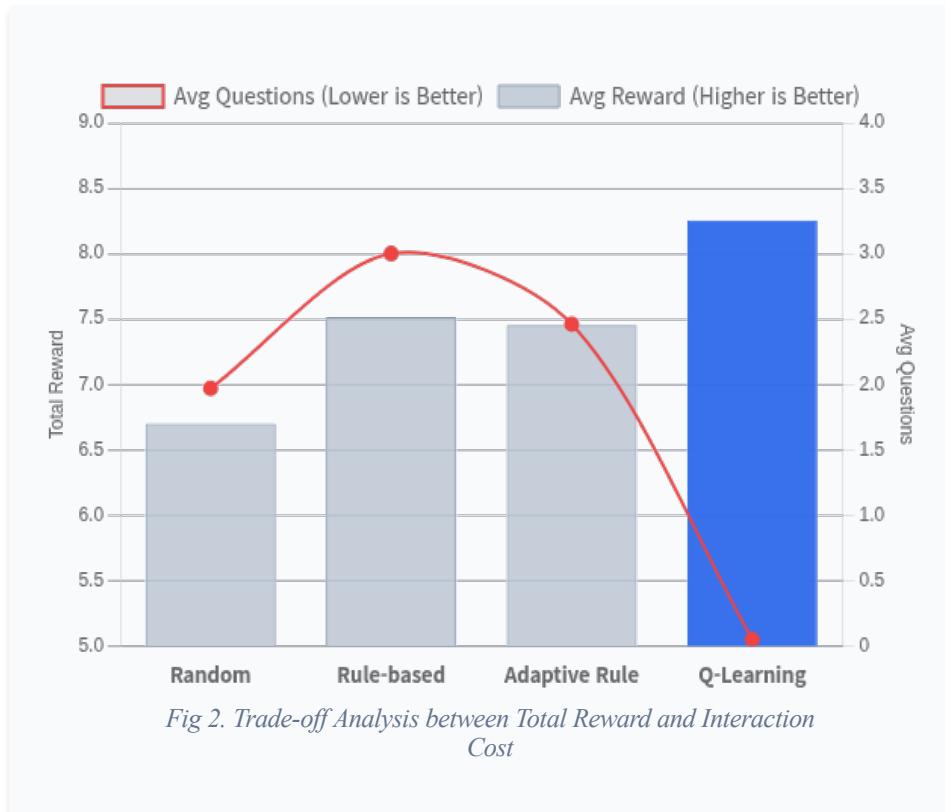
Max Average Reward (+23%)

0.00

Average Questions

Agent	Reward	Success Rate	Avg Questions
Random Baseline	6.699	89.0%	1.97
Rule-based	7.513	100.0%	3.00
Adaptive Rule	7.450	100.0%	2.46
Q-Learning (Proposed)	8.254	100.0%	0.00

알고리즘 비교: Phase1



Agent Type	Reward	Avg Qs	Success
Random Baseline	6.699	1.97	89.0%
Rule-based (Fixed)	7.513	3.00	100%
Adaptive Rule	7.450	2.46	100%
Q-Learning	8.254	0.00	100%

Key Findings

1. Efficiency Dominance: Q-Learning 에이전트는 평균 질문 수를 0으로 최소화하면서도 가장 높은 보상(8.254)을 달성함. 이는 불필요한 질문 비용을 절감하는 전략이 유효함을 입증함
2. Rule-based Limit: 규칙 기반 에이전트는 100% 성공률을 보였으나, 고정된 질문 패턴(평균 3.0회)으로 인해 비용 효율성 측면에서 강화학습 모델보다 열세임

실험 세팅: Phase2 & 3

Experimental Parameters

Category	Parameter	Value & Description
Training Episodes	Pre-training	1,000 eps (Phase 2 only)
	Integrated Training	1,000 eps (Phase 3 End-to-End)
DQN Architecture	Hidden Layers	FC(256) → ReLU → FC(256) → ReLU
	Network Size	Increased capacity for complex scheduling states
Hyperparameters	Optimization	LR: 5e-4, Batch: 64, Buffer: 10,000
Reward Function	Total Reward	$R_{\text{total}} = R_1 + R_2 + R_{\text{synergy}}$

통합 결과 Key Metrics

Total Reward Increase +34.1% vs Phase 3 Baseline	Synergy Bonus +165% Efficiency Gain	Success Rate +6.0%^p 83.0% → 89.0%	vs Random Agent +120.5% Overall Improvement
<hr/>			
Metric	Before (Baseline)	After (Integrated)	Change
Total Reward	23.84	31.98	▲ 34.1%
End-to-End Success	83.0%	89.0%	▲ 6.0%^p
Synergy Bonus	4.06	10.76	▲ 165%
Preference Match	46.0%	43.0%	▼ 3.0%^p

시스템이 Synergy Bonus (즉시 예약 가능 차량 추천)를 극대화하는 방향으로 학습됨. 이로 인해 고객 선호 시간 매칭률(Preference Match)은 소폭 감소했으나, 전체적인 예약 성공률과 총 보상은 크게 향상되는 Trade-off 관계가 관찰됨. 이는 예약 실패 비용이 선호 시간 이탈 비용보다 크다는 도메인 특성이 반영된 결과임

Hyperparameter Sensitivity – 설정과 범위

최적의 성능을 도출하기 위해 Q-Learning (Phase 1)과 DQN (Phase 2)의 핵심 하이퍼파라미터에 대한 Grid Search를 수행함

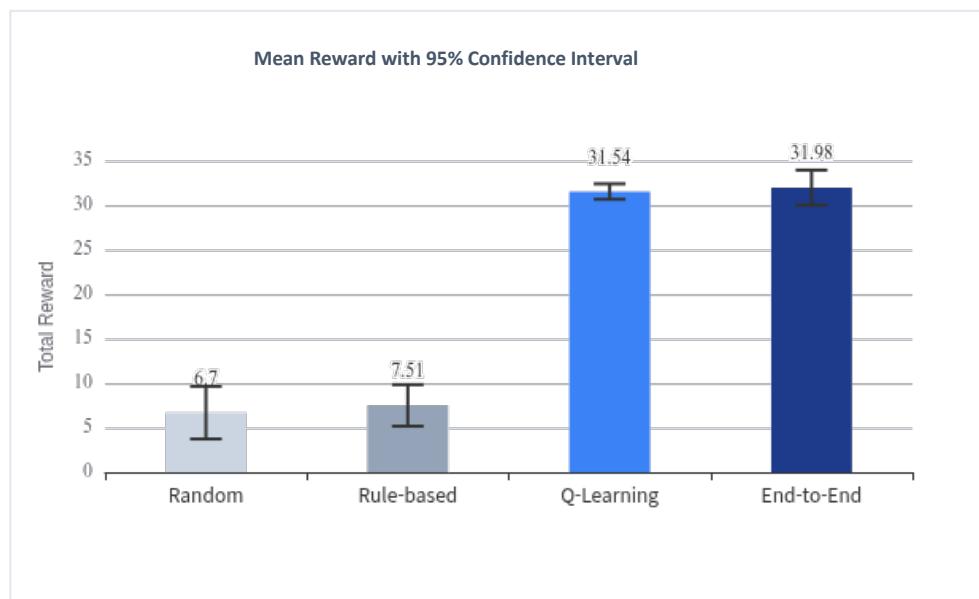
Algorithm	Parameter	Search Range (Grid)	Description / Fixed
Q-Learning(Phase 1)	Learning Rate (α)	{0.05, 0.10, 0.20}	학습 단계 크기 영향 분석
	Discount (γ)	{0.90, 0.95, 0.99}	미래 보상의 가중치
	ϵ -decay	{0.995, 0.998, 0.999}	Min ϵ = 0.05 고정, 1000 Episodes
DQN(Phase 2)	Learning Rate	{1e-4, 3e-4, 1e-3}	Adam Optimizer 사용
	Discount (γ)	{0.95, 0.99}	장기적 스케줄링 이익 고려
	Target Update	{100, 500, 1000}	Q-Target 네트워크 동기화 주기 (step)
	Batch Size	{32, 64, 128}	Replay Buffer 50k (Warmup 1k)

각 파라미터 조합(Combination)에 대해 3회 반복 실험(3 Runs)을 수행하여 평균 성능(Total Reward, Success Rate)을 기록
모든 실험은 동일한 차량 데이터셋(23종)과 사용자 페르소나(100명) 환경에서 진행됨

Random Seed 실험 및 신뢰구 간분석

실험 조건: 10개의 서로 다른 Random Seed (0~9)를 사용하여 독립적인 학습 및 평가 수행

통계 방법: 각 알고리즘의 Total Reward에 대해 평균과 표준편차를 계산하고, t-분포 기반 95% 신뢰구간을 산출함



Q-Learning은 가장 좁은 신뢰구간을 보여,
초기화 조건에 관계없이 매우 안정적인 수렴성을 입증

반면, End-to-End (DQN 결합) 모델은 가장 높은 평균 성능
(31.98)을 달성했으나,
표준편차가 상대적으로 큼을 확인할 수 있음

이는 신경망 기반 근사(DQN)가 초기 가중치나 탐험
(Exploration)의 무작위성에 더 민감하게 반응함을 시사하며,
실제 배포 시에는 Ensemble 기법 등을 통해 분산을 줄이는
추가 연구가 필요할 수 있음

모바일 데모 - Overview

Implementation

Streamlit 기반의 경량화된 웹 애플리케이션으로 구현. 모바일 및 데스크탑 환경 모두에서 접근 가능한 반응형 인터페이스 제공

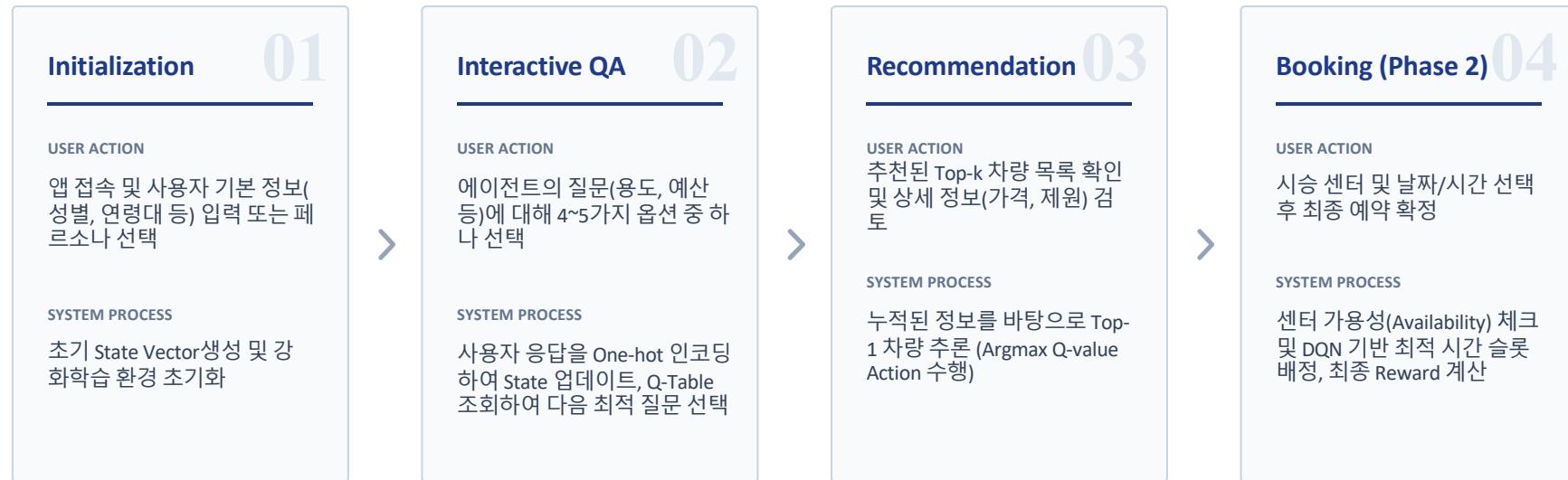
Real-time MDP Visualization

강화학습 에이전트의 내부 상태(Internal State)를 투명하게 시각화. 사용자는 현재 Step에서의 Q-value 기반 의사결정 과정을 실시간으로 확인 가능

Interaction Flow

1. Initialization: 사용자 프로필 설정
2. Phase 1 (Dialog): 필수 질문(4개)을 통한 State 업데이트
3. Recommendation: Top-k 차량 제안
4. Phase 2 (Booking): 센터 스케줄링 및 최종 확정

모바일 데모 - Process



모바일 데모



Live Demo: <https://driving-booking.streamlit.app>

The image displays three screenshots of a mobile application interface for car rental bookings. The interface is designed to look like a chat conversation between a user and a virtual assistant.

Screenshot 1 (Left): Shows the initial greeting and system status. It includes a Q-Learning summary table with columns for Action, 대기 중 (Waiting), Reward, Step, Policy, and 대기 (Waiting). The reward is +0.0, step is 0, policy is Policy, and waiting time is 대기 중 (Waiting). Below the table, it says "추가 질문 - 1 | 다른 차량 - 5 | 예약 확정 +15". A "시작하기" button is at the bottom.

Screenshot 2 (Middle): Shows the AI asking about the number of passengers. The user responds with "3~4명" (3-4 people). The AI then asks about the location, with the user responding with "송파/강동" (Songpa/Gangdong). The AI continues to ask about vehicle type, with the user responding with "3000-4500만원" (3000-4500万won).

Screenshot 3 (Right): Shows the AI confirming the booking details. It lists: 연료: gasoline, 좌석: 5인승, 가격: 3,800만원 ~ 4,300만원. It also asks if the user wants to proceed with the booking. The user confirms with "시승 예약할게요!". The AI then informs the user that the booking is successful, providing the date and time: 2025년 12월 07일 09:00. It also provides a summary of the booking details: 차량: 이번때 N, 차종: n, 연료: 가솔린, 좌석: 5인승, 가격대: 3,800~4,300만원, 장소: 송파 시승센터 (서울 송파구 올림픽로 300), 날짜: 2025년 12월 07일, 시간: 09:00. A note at the bottom states: "예약 확인 문자가 발송될 예정입니다. 시승 당일 운전면허증을 지참해 주세요. 감사합니다! 🙏".

연구 결론

Research Contributions

1. Domain Extension

"20 Questions" 게임 프레임워크를 실제 자동차 비즈니스 도메인으로 확장 적용

2. End-to-End Optimization

Two-Phase framework를 통해 통합 차량 추천(Q-Learning)과 일정 예약 (DQN)의 시너지 보너스를 보상으로 채택

3. Efficiency Improvement

기존의 7~8개의 단계별 예약 시스템을 4~5단계 대화형 인터페이스 예약 추천으로 변경함으로써 고객의 시승 예약 성공률을 높임

Key Results

+34.1%

TOTAL REWARD
vs. Naive Integration

89.0%

SUCCESS RATE
End-to-End Conversion

+120%

PERFORMANCE GAIN

향후 연구 과제

1. Reward Function Refinement

현재의 Rule-based 보상을 넘어, 실제 사용자의 만족도 설문 결과와 연동된 Data-driven Reward Shaping이 필요함. 또한, 최소 질문 수 보장을 위한 Explicit Constraint를 보상 함수에 추가하여 안정성 확보

2. Algorithm Scalability

복잡한 대화 문맥(Context) 처리를 위해 현재의 Tabular/Linear Approximation에서 벗어나 정책 신경망 도입 고려. 이를 통해 상태 공간의 차원이 늘어나도 강건한 학습 가능

3. Continuous Learning Pipeline

데모 운영 중 수집되는 로그 데이터를 활용한 Off-policy Learning 파이프라인 구축. 주기적인 모델 업데이트와 A/B 테스트를 통해 실제 서비스 수준의 성능 검증 필요