Q-Learning for Battery Storage Optimization in Energy Markets

Volterra Energy Solutions -- Optimizing Grid-Scale Battery Dispatch with Reinforcement Learning

---

Section 1: The Business Problem

Volterra Energy Solutions operates a fleet of 14 grid-scale lithium-ion battery storage systems across three regional electricity markets in the United States. Each battery installation has a capacity of 50 MWh and a power rating of 25 MW, meaning each unit can discharge at full power for two hours.

The core business model is energy arbitrage: buy electricity when prices are low (typically overnight and during mild weather), store it in the batteries, and sell it back to the grid when prices are high (typically during peak demand periods in the afternoon). The profit margin depends entirely on the spread between buy and sell prices.

The Decision-Making Challenge

Every 15 minutes, the dispatch controller must decide for each battery: **charge, discharge, or hold**. This decision depends on several factors:

- **Current electricity price**: Real-time prices from the wholesale market, which can fluctuate from \$15/MWh to over \$200/MWh within a single day.
- **Current state of charge (SoC)**: The battery's energy level, ranging from 10% (minimum for longevity) to 90% (maximum for safety).
- **Time of day**: Prices exhibit strong daily patterns, but these patterns shift with seasons and weather.
- **Grid conditions**: Demand forecasts, renewable generation levels, and transmission constraints.

Before adopting reinforcement learning, Volterra used a rule-based dispatch system designed by energy market analysts. The rules were straightforward:

Rule	Condition	Action
1	Price < \$30/MWh AND SoC < 80%	Charge
2	Price > \$80/MWh AND SoC > 30%	Discharge

Rule	Condition	Action
3	Otherwise	Hold

This rule-based system generated an average annual revenue of \$2.1 million per battery installation, with a fleet-wide revenue of \$29.4 million. However, internal analysis estimated that perfect-hindsight trading (if they could predict prices exactly) would yield \$4.8 million per battery -- meaning the rule-based system captured only 44% of the theoretical maximum.

Limitations of the Rule-Based System

The rule-based system had three critical limitations:

1. **Static thresholds**: The \$30 and \$80 price thresholds were set manually and did not adapt to changing market conditions. During the 2024 summer heat wave, wholesale prices exceeded \$150/MWh for extended periods, and the \$80 threshold was far too conservative.

2. **No look-ahead**: The system made decisions based purely on the current price, with no consideration of whether prices might go higher in the next hour. This led to premature discharging -- selling at \$85/MWh when prices would reach \$180/MWh two hours later.

3. **No state-awareness**: The system did not account for the battery's current SoC in a nuanced way. A battery at 85% SoC was treated the same as one at 50% SoC, even though their optimal strategies differ significantly.

The engineering team at Volterra recognized that the dispatch problem is fundamentally a sequential decision-making problem under uncertainty -- exactly the domain where reinforcement learning excels.

Section 2: The Technical Approach

Framing as a Markov Decision Process

The first step was formulating the battery dispatch problem as a Markov Decision Process (MDP):

State space *S*: Each state is a tuple *(SoC, Price, Hour, DayType)* where: - *SoC \in {0.10, 0.15, 0.20, \ldots, 0.90}* -- 17 discrete levels - *Price* -- discretized into 20 buckets from \$0 to \$300/MWh - *Hour \in {0, 1, \ldots, 23}* -- 24 hours - *DayType \in {weekday, weekend}* -- 2 types

Total state space: $17 \times 20 \times 24 \times 2 = 16{,}320$ states.

Action space *A*: Three discrete actions: - *a = 0*: Charge (buy electricity, increase SoC) - *a = 1*: Hold (do nothing) - *a = 2*: Discharge (sell electricity, decrease SoC)

Reward function $R(s, a)$: - Discharge at price p: $R = p \times P_{rated} \times \Delta t - C_{deg}$ (revenue minus degradation cost) - Charge at price p: $R = -p \times P_{rated} \times \Delta t - C_{deg}$ (negative because buying) - Hold: $R = 0$ - Where $P_{rated} = 25$ MW, $\Delta t = 0.25$ hours (15 minutes), and $C_{deg} = \$0.50$ per cycle-equivalent

Transition dynamics: The SoC transitions deterministically based on the action (charge adds ~3.7% SoC, discharge removes ~3.7% per step). Prices transition stochastically based on historical patterns.

Q-Learning Implementation

The team implemented tabular Q-Learning with the following hyperparameters:

Parameter	**Value**	**Rationale**
Learning rate α	0.05	Moderate pace: fast enough to learn, slow enough for stability
Discount factor γ	0.95	High because future revenue matters (discharge at peak requires earlier charging)
Initial ϵ	1.0	Start fully exploratory
ϵ decay	0.9995 per episode	Slow decay over ~10,000 episodes
Minimum ϵ	0.02	Always maintain 2% exploration for market shifts

The Q-Learning update rule:

$$Q(s, a) \leftarrow Q(s, a) + \alpha \left[r + \gamma \max_{a'} Q(s', a') - Q(s, a) \right]$$

Training used 3 years of historical price data (2021-2023) from the ERCOT market in Texas, with each year providing approximately 35,000 decision points per battery. The agent was trained over 50,000 episodes, where each episode was a randomly sampled 24-hour period from the historical data.

Key Design Decisions

1. **State discretization**: Continuous SoC and price values were discretized into bins. The team experimented with bin sizes and found that 17 SoC levels and 20 price levels provided a good balance between granularity and learning speed.

2. **Reward shaping**: A small penalty (\$0.50) was added for each charge/discharge action to account for battery degradation. This prevented the agent from cycling the battery unnecessarily.

3. **Episode structure**: Each training episode was a full 24-hour period (96 time steps at 15-minute intervals). This allowed the agent to learn the daily price pattern.

4. **Safety constraints**: The agent was prevented from charging above 90% SoC or discharging below 10% SoC. Attempting these actions resulted in a "hold" with a small negative reward as penalty.

Section 3: The Results

Training Dynamics

The Q-Learning agent converged after approximately 30,000 episodes. The figure below shows the training progression:

- **Episodes 0-5,000**: Highly exploratory phase. The agent's daily profit was negative because random actions often meant buying high and selling low.
- **Episodes 5,000-15,000**: Rapid improvement phase. The agent learned the basic arbitrage pattern -- charge at night, discharge during afternoon peak.
- **Episodes 15,000-30,000**: Fine-tuning phase. The agent learned to hold instead of discharging at moderately high prices when even higher prices were likely later.
- **Episodes 30,000+**: Converged. Performance plateaued at a stable level.

Performance Comparison

The Q-Learning agent was backtested on 6 months of out-of-sample data (January-June 2024) that was not used during training:

Metric	Rule-Based	Q-Learning Agent	Improvement
Daily revenue per battery	\$5,753	\$8,219	+42.9%
Monthly revenue per battery	\$172,600	\$246,575	+42.9%
Annual projected revenue per battery	\$2.10M	\$3.00M	+42.9%
Fleet annual projected revenue	\$29.4M	\$42.0M	+42.9%
% of theoretical maximum	44%	63%	+19 pp
Average discharge price	\$97/MWh	\$134/MWh	+38.1%
Average charge price	\$28/MWh	\$22/MWh	-21.4%
Daily charge/discharge cycles	1.8	1.4	-22.2%

Key Behavioral Differences

The Q-Learning agent exhibited several behaviors that the rule-based system did not:

1. **Patient discharging**: The agent learned to wait for peak prices rather than discharging at the first above-threshold price. On a typical summer day, the rule-based system would

start discharging at 2 PM when prices hit \$85/MWh, while the Q-Learning agent would hold until 4-5 PM when prices peaked at \$150-180/MWh.

2. **Opportunistic charging**: The agent learned that prices in the early morning (2-5 AM) are consistently lower than late evening (9-11 PM), shifting more charging to the pre-dawn hours.

3. **Fewer cycles**: By being more selective about when to charge and discharge, the agent completed fewer cycles per day (1.4 vs 1.8), reducing battery degradation and extending the expected battery lifespan by approximately 18 months.

4. **Weather-adaptive behavior**: Through the hour and day-type features, the agent implicitly learned seasonal patterns. During winter, it shifted its discharge window earlier (3-6 PM heating demand) compared to summer (4-7 PM cooling demand).

Production Deployment

Volterra deployed the Q-Learning agent on 4 of its 14 battery installations in a controlled rollout in July 2024. After 3 months of production operation:

- The Q-Learning batteries outperformed the rule-based batteries by 38% in live revenue.
- The system handled the August 2024 heat wave effectively, correctly identifying the sustained high-price period and maximizing discharge during the highest-price hours.
- Zero safety incidents or constraint violations were recorded.

The full fleet migration was completed by November 2024, with an estimated additional annual revenue of \$12.6 million fleet-wide compared to the previous rule-based approach.

Section 4: Lessons Learned and Extensions

What Worked Well

1. **Tabular Q-Learning was sufficient**: With 16,320 states and 3 actions, the Q-table had fewer than 50,000 entries. This was small enough that tabular methods converged reliably without the complexity of function approximation.

2. **Historical data as a simulator**: Using 3 years of historical price data as a training environment eliminated the need for a price forecasting model. The agent learned directly from real market patterns.

3. **Conservative deployment**: The phased rollout (4 batteries first, then full fleet) built confidence with the operations team and provided a controlled comparison.

Challenges Encountered

1. **State discretization trade-offs**: Too few price bins (10) caused the agent to treat \$40/MWh and \$60/MWh as the same state, leading to suboptimal decisions. Too many bins (50) slowed convergence because the state space was too large. The team settled on 20 bins after systematic experimentation.

2. **Non-stationarity**: Electricity markets evolve over time (new generators, policy changes, demand shifts). The Q-table trained on 2021-2023 data gradually became less optimal when deployed in 2024. The team addressed this by continuing online learning with a low epsilon (2%) during production.

3. **Reward scaling**: Initial experiments used raw revenue as the reward, which ranged from -\$1,500 to +\$1,500 per step. This caused numerical instability in Q-value updates. Normalizing rewards to the [-1, 1] range resolved the issue.

Future Extensions

1. **Deep Q-Network (DQN)**: The team is developing a DQN variant that takes raw price time series as input (last 96 price points) rather than discretized states. This eliminates the information loss from discretization and can capture richer temporal patterns.

2. **Multi-agent coordination**: Currently, each battery is controlled independently. A multi-agent approach could coordinate the fleet to avoid all batteries discharging simultaneously (which could depress local prices).

3. **Ancillary services**: Beyond arbitrage, batteries can provide frequency regulation and spinning reserves. Extending the action space to include these services could increase revenue by an estimated 15-25%.

4. **Model-based RL**: Combining Q-Learning with a learned price transition model (Dyna-Q) could improve sample efficiency and allow the agent to "imagine" rare but high-value scenarios (price spikes) that appear infrequently in historical data.