PowRL : A Reinforcement Learning Framework for Robust Management of Power Networks

Motivating Example

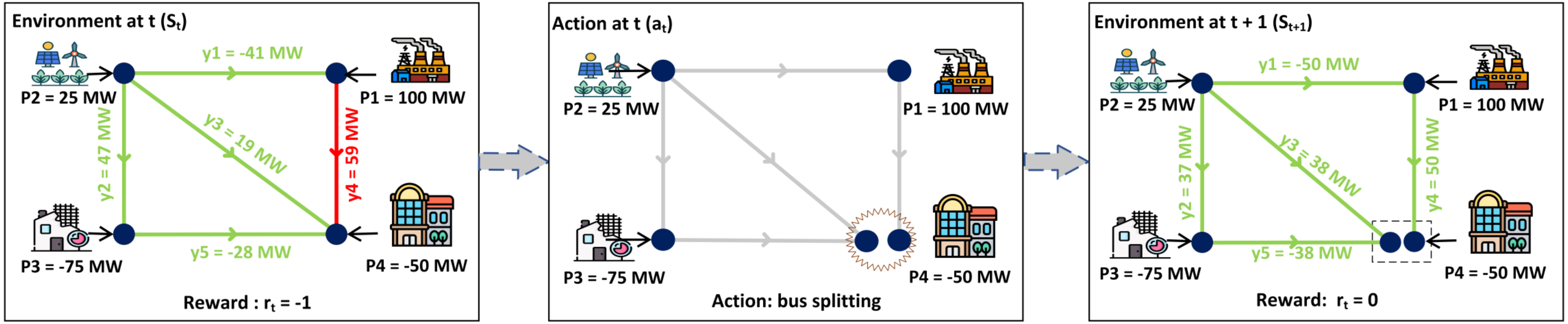


Figure 1: Reliable operation through bus-splitting: instead of disconnecting the overloaded line, a simpler and effective solution is to alter the underlying network topology

Grid2Op Environment

- Offers flexibility to work with realistic scenarios, execute contingency events.
- The challenge scenario consists of industry standard synthetic IEEE-118 network with 36 substations, 59 lines and 22 generators. The remaining part of the IEEE-118 network is represented as an inter-connected load.
- Challenge dataset equipped with realistic production and consumption scenarios, as well as adversarial attacks.
- Demand-Supply balance should be maintained else episode terminates.

Humongous action space

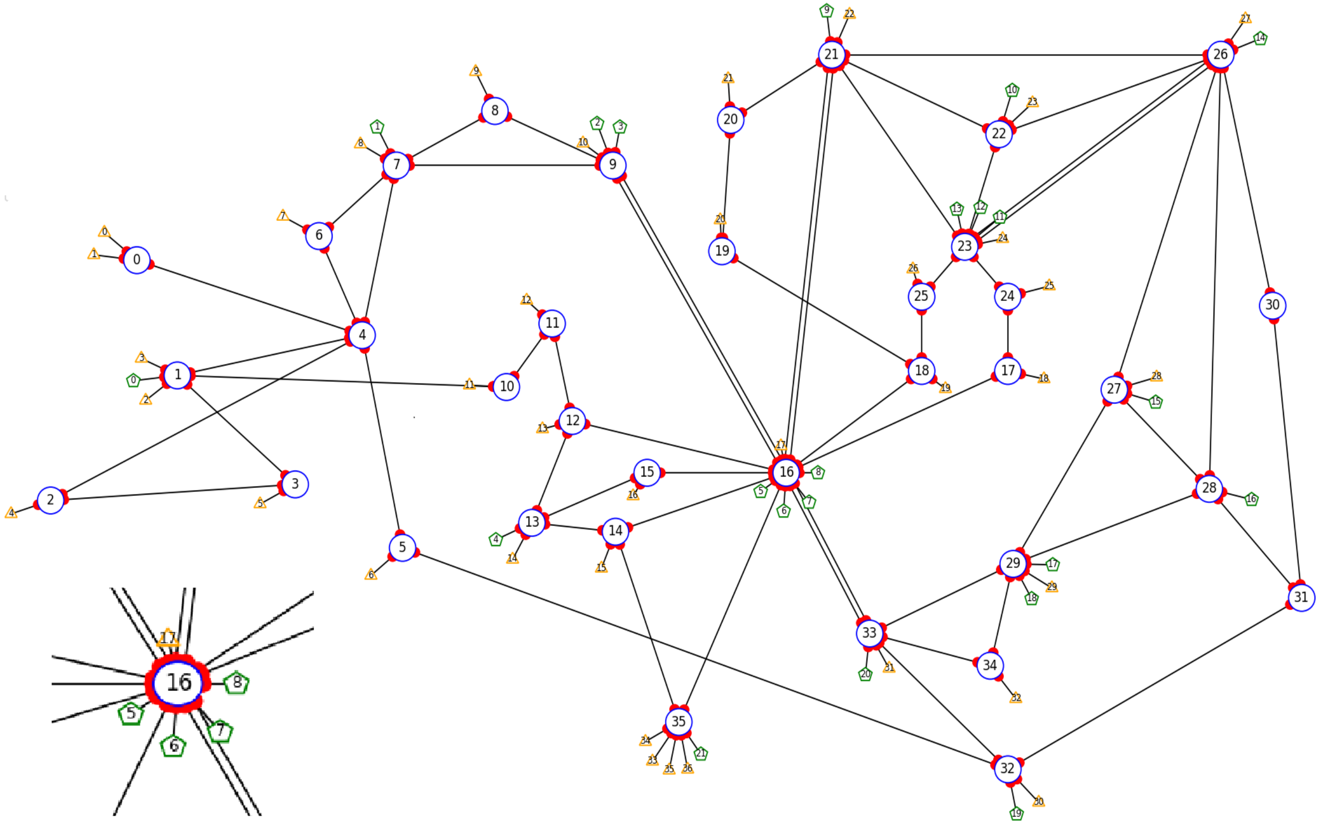


Figure 2: Synthetic IEEE-118 network consisting of 36 substations, 59 lines, 22 generators

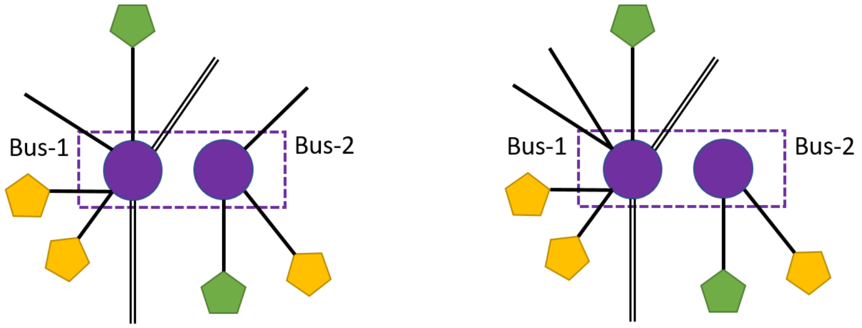


Figure 3: Examples of Valid (left) and invalid (right) topological configuration at a double busbar substation

The number of valid topological reconfigurations for a double busbar substation comprising of N_{line} lines, N_g generators, and N_{load} loads is $2^{N_{tot}-1} - 2^{N_g+N_{load}} + 1$

$$N_{tot} := N_{line} + N_g + N_{load}$$

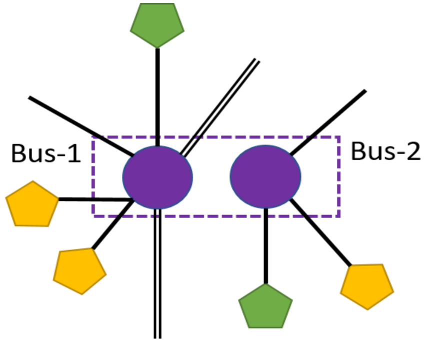
Challenges in Controlling Power Network

- Combinatorially many actions : The number of actions increases exponentially with the number of elements, and for substation 16 it counts nearly 65k.
- Uncertainty with renewables : Weather-related changes that significantly affect power generation may force the operator to take immediate remedial action in order to avoid transmission loss failures or eventual blackouts.
- Adversarial attacks : The L2RPN challenge operates with a heuristically designed opponent to mimic the $N - 1$ security criterion in power networks, it acts in an environment parallel to the RL agent and affects the power grid through forced contingencies.

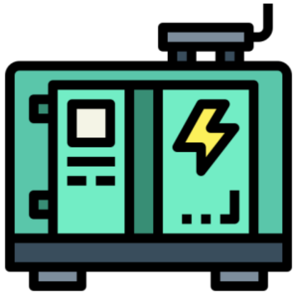
Action

- Do Nothing action
- Doesn't change the status of elements and survives nearly 24%

- Line action includes the line reconnection/disconnection
- Agent can only reconnect lines once the cooldown is over



- Switching elements bus at the substation.
- Flexibility offered is un-explored due very large action space



- Generator redispatch actions modify production to operate power network
- Results in high operating cost

Rules and Evaluation

Constraints:

- Demand-Supply balance should be maintained at any time without load shedding
- Electrical islands are not allowed
- Tripping of power plant not allowed
- Action on an element has certain cooldown to avoid degradation cost

Evaluation:

- In NeurIPS challenge data, agent is evaluated over 24 episodic weekly scenarios
- In WCCI challenge dataset, agent is evaluated over 10 different 3-day long scenarios
- Score is calculated based on operating cost and losses due to blackout
- Agent with less operating cost and less blackout will get higher score

PPO

- PPO trains a stochastic policy in an on-policy way.
- Actor-Critic objective could be written as, $J_\theta = E_t[\nabla_\theta \log \pi_\theta(a_t|s_t) A_t]$
- The new objective proposed by PPO is, $J_\theta = E_t\left[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)} A_t\right]$
- However, if we just start to blindly maximize this value, it may lead to a very large update to the policy weights. To limit the update, the clipped objective is used.

$$J_\theta^{clip} = \mathbb{E}_t[\min(r_t(\theta)A_t, clip(r_t(\theta), 1-\varepsilon, 1+\varepsilon)A_t)]$$

PPO

- There are two primary variants of **PPO**: PPO-Penalty and PPO-Clip.
- **PPO-Penalty** approximately solves a KL-constrained update like TRPO, but penalizes the KL-divergence in the objective function instead of making it a hard constraint, and automatically adjusts the penalty coefficient over the course of training so that it's scaled appropriately.
- **PPO-Clip** doesn't have a KL-divergence term in the objective and doesn't have a constraint at all. Instead relies on specialized clipping in the objective function to remove incentives for the new policy to get far from the old policy.

Ref : https://spinningup.openai.com/en/latest/algorithms/ppo.html?highlight=PPO#documentation-pytorch-version

PPO Algorithm

Algorithm 1 PPO-Clip

1: Input: initial policy parameters θ_0, initial value function parameters ϕ_0

2: **for** $k = 0, 1, 2, ...$ **do**

3: Collect set of trajectories $\mathcal{D}_k = \{\tau_i\}$ by running policy $\pi_k = \pi(\theta_k)$ in the environment.

4: Compute rewards-to-go $\hat{R}_t$.

5: Compute advantage estimates, $\hat{A}_t$ (using any method of advantage estimation) based on the current value function V_{ϕ_k}.

6: Update the policy by maximizing the PPO-Clip objective:

$$\theta_{k+1} = \arg\max_\theta \frac{1}{|\mathcal{D}_k|T} \sum_{\tau \in \mathcal{D}_k} \sum_{t=0}^{T} \min\left(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_k}(a_t|s_t)} A^{\pi_{\theta_k}}(s_t, a_t), \quad g(\epsilon, A^{\pi_{\theta_k}}(s_t, a_t)) \right),$$

typically via stochastic gradient ascent with Adam.

7: Fit value function by regression on mean-squared error:

$$\phi_{k+1} = \arg\min_\phi \frac{1}{|\mathcal{D}_k|T} \sum_{\tau \in \mathcal{D}_k} \sum_{t=0}^{T} \left(V_\phi(s_t) - \hat{R}_t \right)^2,$$

typically via some gradient descent algorithm.

8: **end for**

Ref : https://spinningup.openai.com/en/latest/algorithms/ppo.html?highlight=PPO#documentation-pytorch-version

State

- Time-step information (t) : month, date, hour, minute and day of week
- Generator features (F_{gen}) : P_{gen} , Q_{gen} , V_{gen}
- Load features (F_{load}) : P_{load} , Q_{load} , V_{load}
- Line features (F_{line}) : P_{or} , Q_{or} , V_{or} , a_{or} , P_{ex} , Q_{ex} , V_{ex} , a_{ex}
- Other features are line status (l_1), line rho value (ρ), topology vector of the power grid (I_{topo_vect}), timestep overflow (t_{of}), cooldown period of line (t_l) and substations (t_s) and the time of next planned maintenance t_{nm} and its duration $t_{d.}$
- St := [t , F_{gen} , F_{load} , F_{line} , l_1 , ρ , I_{topo_vect} , t_{of} , t_l , t_s , t_{nm} , t_d]

Actions and Reward

- Action
  - The action space is a set of 240 topological actions.
  - These action are picked based on their impact on network control through extensive simulation.
- Reward
  - The step reward is designed to incur additional penalty when the maximum line rho ρ_{max} exceeds the $\rho_{safety\ threshold}$ of 0.95,

$$r = \begin{cases} 2 - \rho_{\max}, & \text{if } \rho_{\max} < 0.95, \\ 2 - 2\rho_{\max}, & \text{else.} \end{cases}$$

Heuristic-guided topological actions

- Disengage RL agent during ‘acceptable’ grid operation
- Manually disconnect a line during sustained periods of overflow in order to avoid permanent damage, as well as allowing PowRL to reconnect the line back soon after the cooldown period ends.
- Any network reconfigurations are restored to original state as soon as the contingency ends
- Reconnect the line back soon the scheduled maintenance period is over
- Do not disconnect lines that result in network bifurcation

Optimal Action Selection

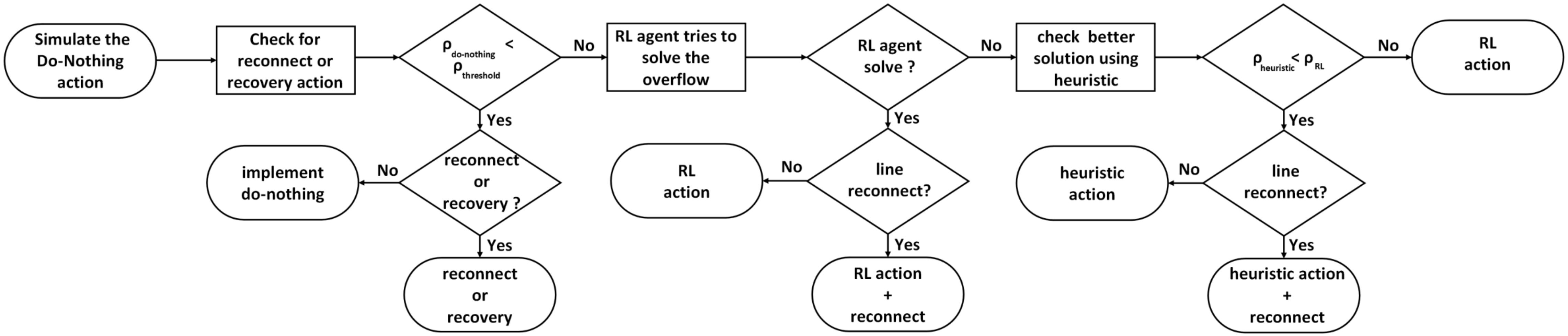


Figure 4: Schematic of PowRL: PowRL combines RL-agent with a threshold based heuristic scheme

Experiments

- PowRL runs on Grid2Op, which emulates the sequential decision-making in the power system, where each episode is divided into a list of states each corresponding to a time step of 5 minutes.
- Lightsim2grid backend to the Grid2Op platform in order to accelerate the computation.
- Agent is evaluated over three different datasets;
  I. NeurIPS 2020 L2RPN Online data
  II. NeurIPS 2020 L2RPN Offline data
  III. WCCI 2020 L2RPN offline data

Results

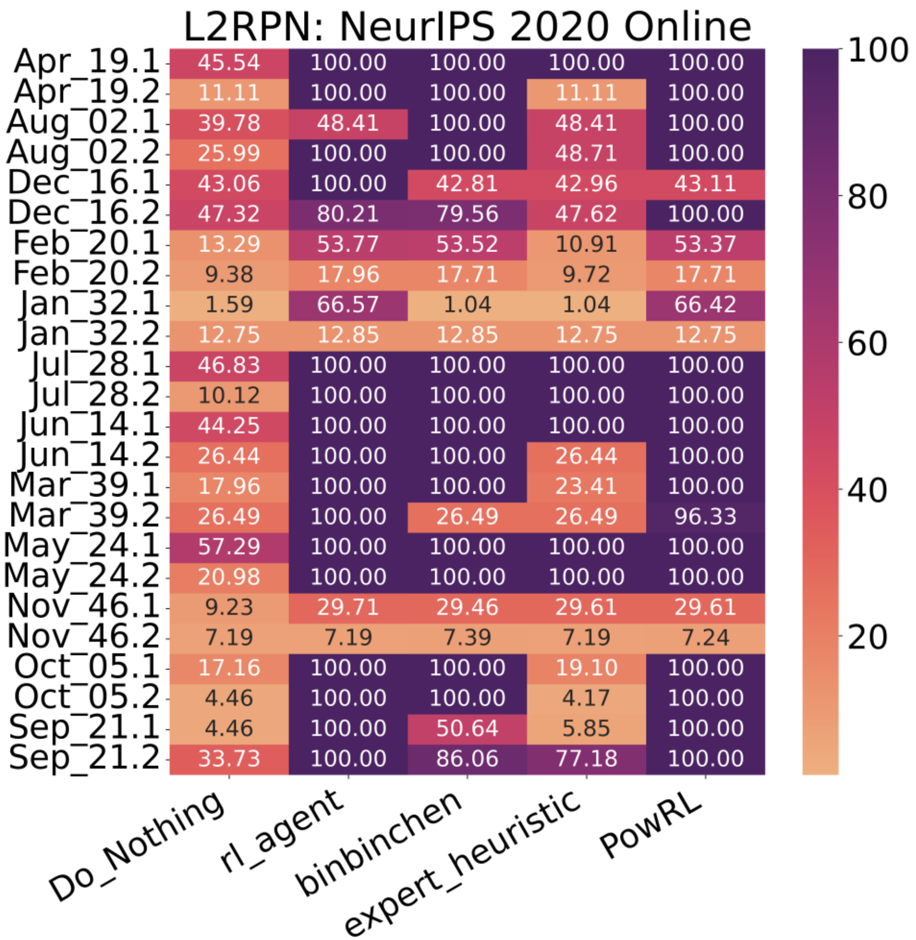


Figure 5: Survival percentages of various agents on the L2RPN NeurIPS 2020 challenge Online dataset

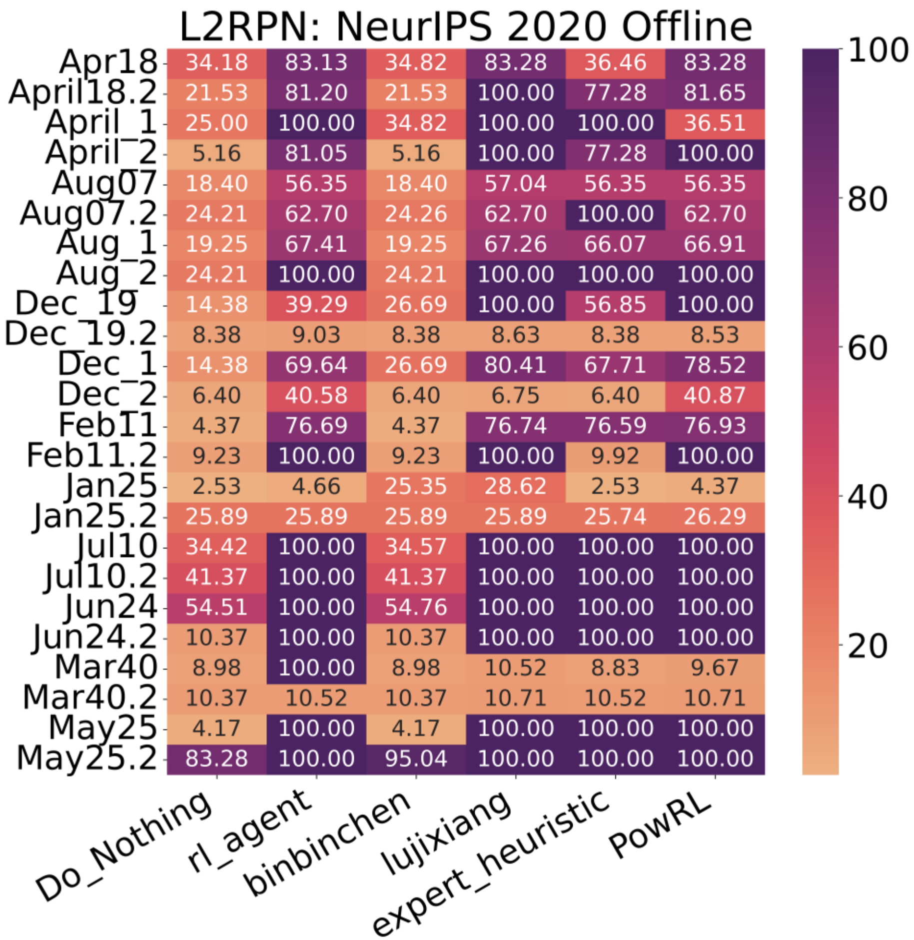


Figure 6: Survival percentages of various agents on the L2RPN NeurIPS 2020 challenge Offline dataset

Results

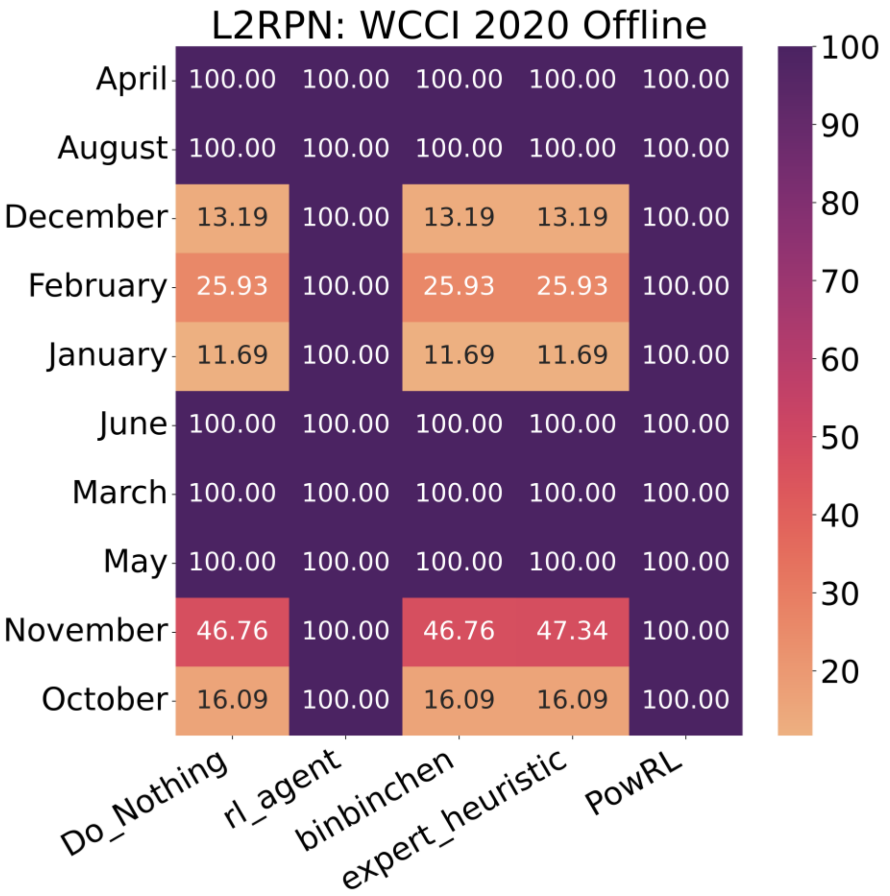


Figure 7: Survival percentages of various agents on the L2RPN WCCI 2020 challenge (Offline)

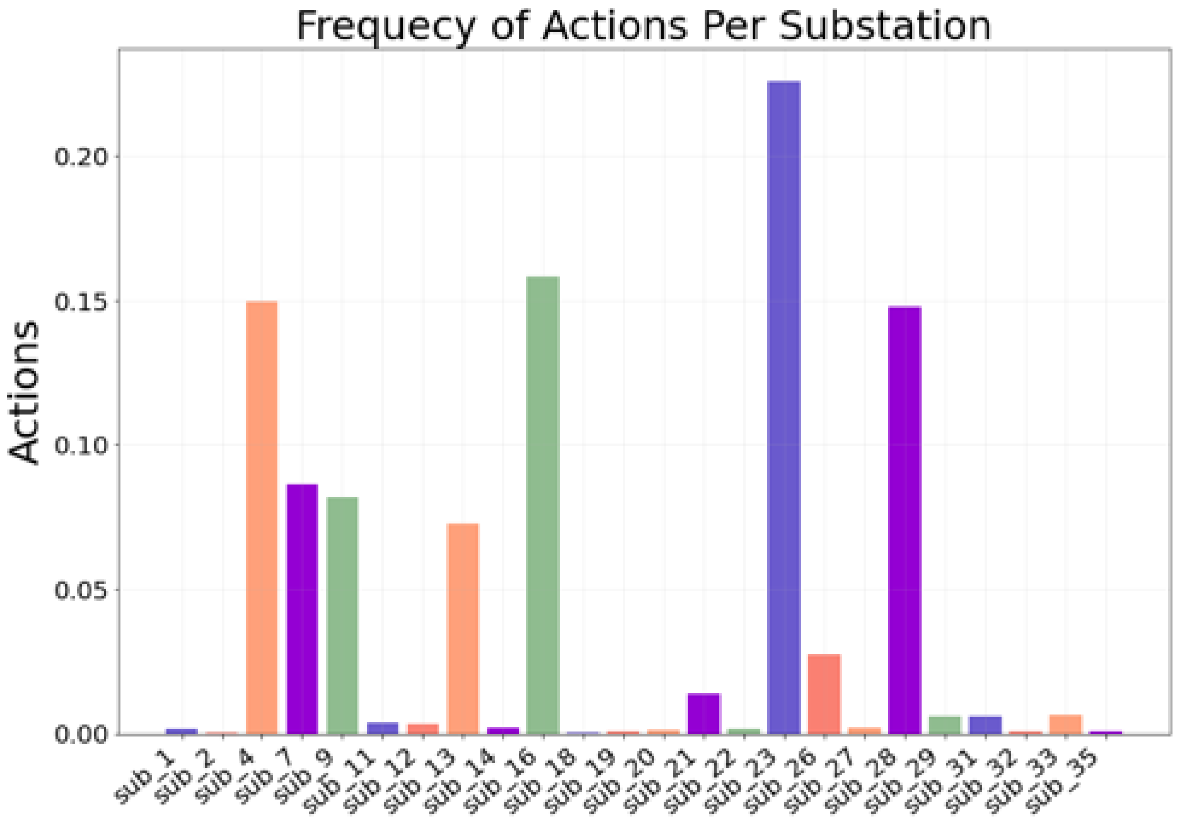


Figure 8: Diversity of remedial actions distributed across multiple substations.

Results

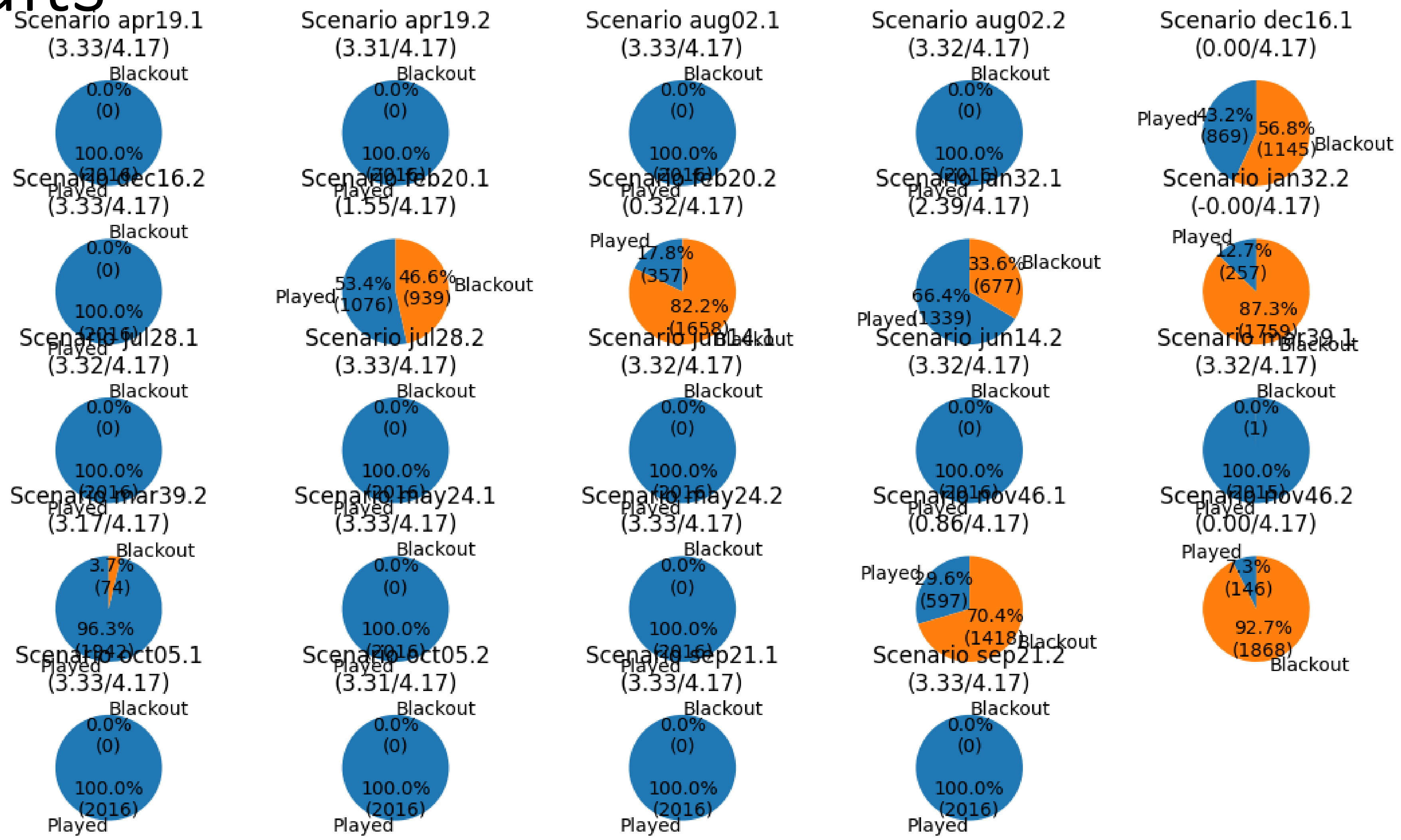


Figure 9: Scenario-wise survival percentage and scores on the L2RPN NeurIPS 2020 challenge data (Online)

Results

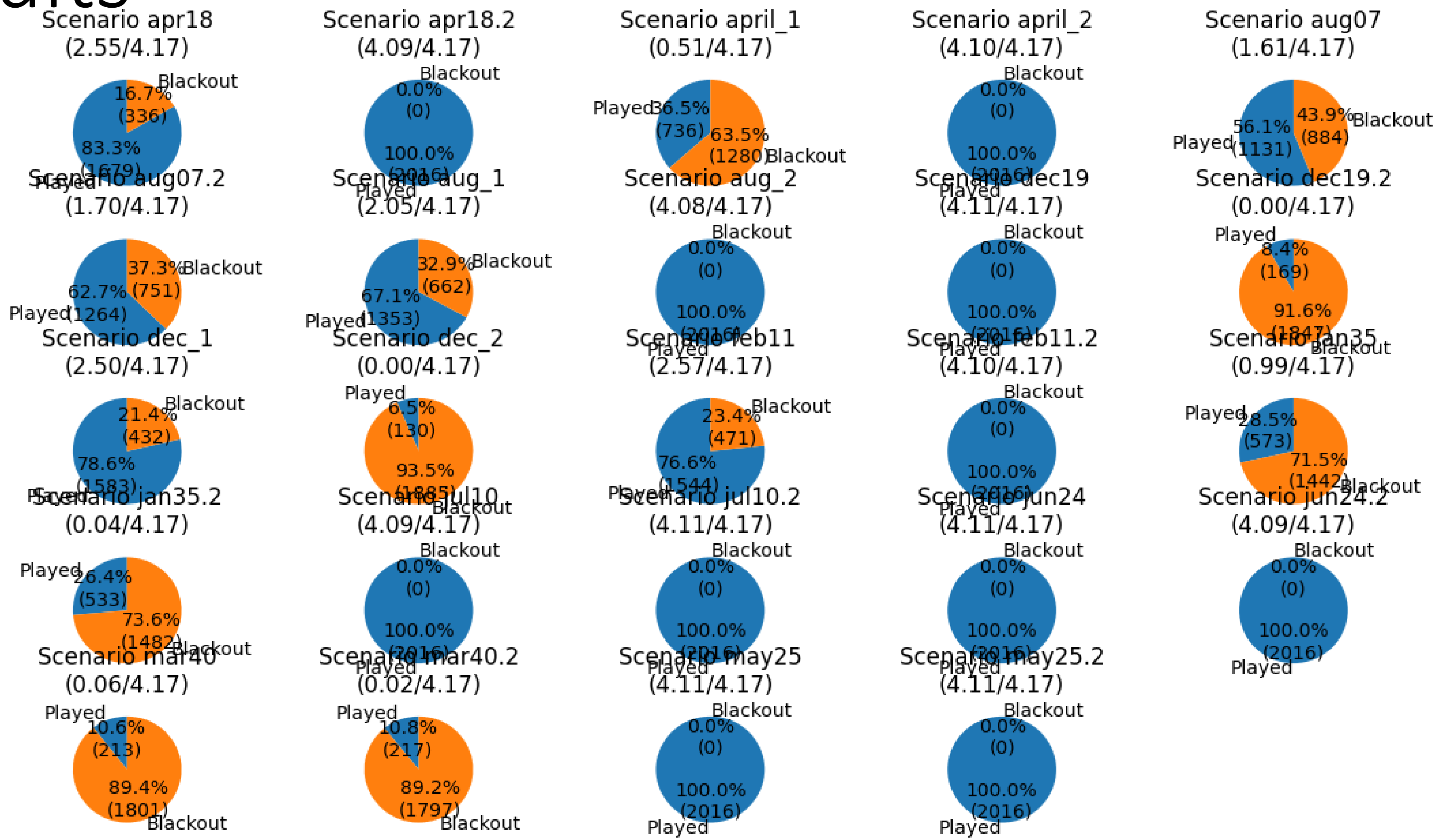


Figure 10: Scenario-wise survival percentage and scores on the L2RPN NeurIPS 2020 challenge data (Offline)

Results

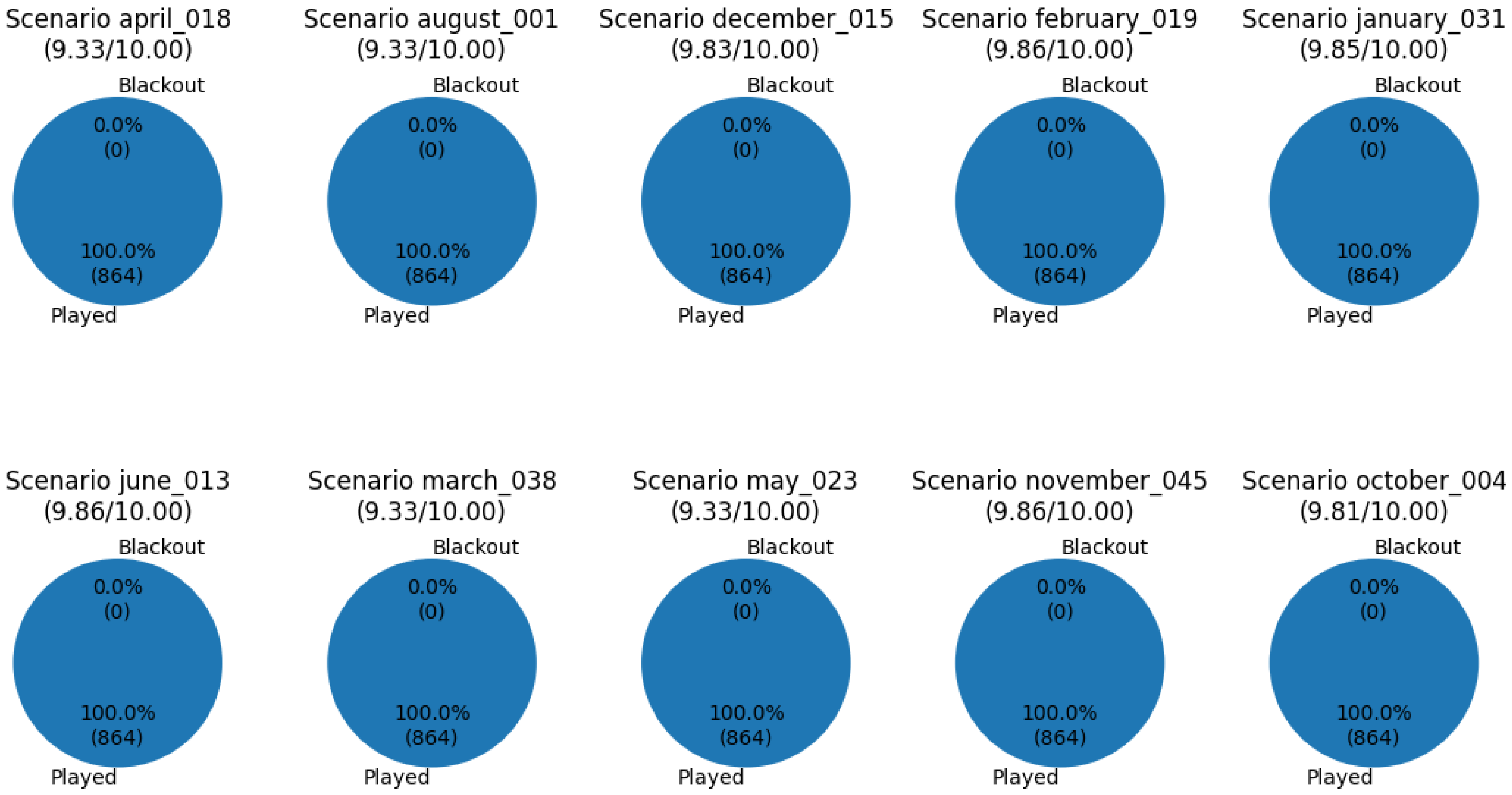


Figure 11: Scenario-wise survival percentage and scores on the L2RPN WCCI 2020 challenge data (Offline)

Summary

Method	NeurIPS (Online)	NeurIPS (Offline)	WCCI	# actions	Run-time (s)
Do_Nothing	0	0.34	50.09	NA	0.54
expert_heuristic	22.53	51.44	50.1	NA	9.76
lujixiang	45	53.96	NA	885	824.58
binbinchen	52.42	3.96	51.1	208 + 1255	503.88
rl_agent	61.05	**61.06**	**96.4**	232 + 500 + redispatch	358.82
PowRL (Ours)	**61.48**	59.69	**96.4**	**206 + 34**	**353.36**

Table 1: Performance on L2RPN Challenge Datasets