AutoBS: Autonomous Base Station Deployment with Reinforcement Learning and Digital Network Twins

Ju-Hyung Lee [1] **Andreas F. Molisch** [2]

Abstract

This paper introduces *AutoBS*, a reinforcement learning (RL)-based framework for optimal base station (BS) deployment in 6G radio access networks (RAN). AutoBS leverages the Proximal Policy Optimization (PPO) algorithm and fast, site-specific pathloss predictions from PMNet—a generative model for digital network twins (DNT). By efficiently learning deployment strategies that balance coverage and capacity, AutoBS achieves about 95% of the capacity of exhaustive search in single BS scenarios (and in 90% for multiple BSs), while cutting inference time from hours to milliseconds, making it highly suitable for real-time applications (*e.g.*, ad-hoc deployments). AutoBS therefore provides a scalable, automated solution for large-scale 6G networks, meeting the demands of dynamic environments with minimal computational overhead.

1. Introduction

The rollout of 6G networks demands higher base station (BS) density due to requirements for higher area spectral efficiency and the use of higher frequencies like millimeter-wave (mmWave). The latter offer enhanced bandwidth and low latency but suffer from severe signal attenuation and limited propagation range, particularly in complex urban environments. As a result, dense BS deployment becomes essential to maintain reliable coverage and capacity. Furthermore, cell-free massive MIMO, where a significant number of access points with one of a few antennas each are distributed over an area (Interdonato et al., 2019) is anticipated to be widely used in 6G.

However, optimizing BS placement in such environments presents significant challenges due to highly site-specific conditions. Traditional BS deployment methods often rely on manual planning, based on real-world measurement of the propagation conditions (Molisch, 2023), and/or computationally intensive ray-tracing (RT) simulations using tools like *WirelessInsite* (REMCOM) or SionnaRT (Aoudia et al., 2025; Hoydis et al., 2022). The fact that these approaches are time- and labor- consuming makes them less suited for the dense deployment in 6G. Furthermore, they are not suited for real-time adaptation of network topology, *e.g.*, dynamic addition and placement of (mobile) BSs in reaction to change of user density, *e.g.*, at special events or for ad-hoc deployment by emergency responders or military applications.

The network planning challenge can also be viewed within the framework of digital network twins (DNT), virtual replicas of physical environments that enable real-time simulation and optimization of network performance under site-specific conditions. Even within DNT frameworks, manually optimizing BS placement remains computationally prohibitive.

Hence, effective real-time optimization of large-scale networks requires the integration of machine learning (ML) approaches to automate and accelerate network planning, particularly for radio access network (RAN) deployments.

To address these challenges, we propose AutoBS—an autonomous BS deployment framework utilizing deep reinforcement learning (DRL) in conjunction with a generative DNT model for site-specific optimization. AutoBS integrates PMNet (Lee & Molisch, 2024; Lee et al., 2023), a fast and accurate ML-based pathloss predictor, to efficiently compute rewards based on coverage and capacity metrics. By autonomously learning optimal BS placement strategies, AutoBS significantly reduces computational complexity and deployment time compared to conventional methods, making it highly suitable for real-time optimization in large-scale 6G RAN scenarios.

Related Works. Recent studies on DNT have focused primarily on developing realistic simulation platforms. (Pegurri et al., 2025a) integrated the `ns-3` network simulator with the differentiable ray tracer `SionnaRT`, creating a scalable

[1]Nokia Technologies, Sunnyvale, CA, USA [2]Department of Electrical and Computer Engineering, University of Southern California, Los Angeles, CA, USA. Correspondence to: Ju-Hyung Lee <juhyung.lee@nokia.com>.

Proceedings of the 42^{nd} International Conference on Machine Learning, Vancouver, Canada. PMLR 267, 2025.

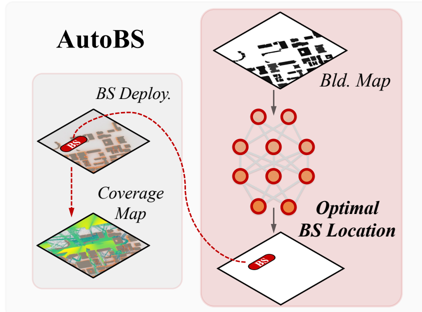


Figure 1. Overview of the AutoBS framework, where the DRL agent leverages PMNet's generative pathloss map (DNT of the site) to evaluate coverage and determine optimal BS locations.

full-stack DNT framework validated in dynamic urban vehicular scenarios(Pegurri et al., 2025b). Additionally, learning-based physical-layer solutions have emerged, such as the neural multi-user MIMO receiver by (Cammerer et al., 2023), and the real-time neural receiver fine-tuned with site-specific propagation data by (Wiesmayr et al., 2024). In contrast to these works—centered around simulation fidelity or neural receiver design—**AutoBS** leverages a generative DNT model (PMNet) combined with a PPO agent to automate BS deployment decisions, shifting ML-driven DNT applications from device-level adaptations to network-level topology optimization.

Contributions. This paper presents AutoBS, an autonomous BS deployment framework utilizing DRL and a generative DNT model (*i.e.*, PMNet). Our key contributions are summarized as follows:

- **AutoBS Framework Design:** We present a novel integration of deep reinforcement learning (PPO) with PMNet, a generative DNT model that produces fast and accurate pathloss maps. This coupling enables efficient, environment-aware reward computation during training and supports scalable BS placement optimization. To the best of our knowledge, AutoBS is the first to embed a learned DNT directly within a closed-loop RL training pipeline for deployment strategy learning.[1] Details on reward design appear in **Table 4** in Sec. A.
- **Sequential Multi-BS Deployment Strategy:** AutoBS handles both static single BS and asynchronous (sequential) multi-BS deployments, providing flexibility across diverse deployment scenarios (see **Fig. 5** in Sec. 4).
- **Fast and Near-Optimal Deployment:** Simulations demonstrate that AutoBS reduces inference time from hours to milliseconds compared to exhaustive methods, particularly in multi-BS deployments, making it practical for real-time optimization (see **Table 3** in Sec. 4).

2. Methodology

2.1. Problem Formulation and DRL Approach

The BS deployment problem can be mathematically formulated as a discrete optimization problem over a site-specific environment. Let the environment be described by a 2D grid-based site map $\mathcal{S}$, where each coordinate (i,j) corresponds to a spatial location annotated with physical features such as buildings, obstacles, or terrain. Each BS is placed at a location $(i,j) \in \mathcal{S}$ drawn from the set of allowable deployment positions $\mathcal{B} \subseteq \mathcal{S}$, such as rooftops or other candidate sites.

Given a total budget of M BSs to deploy, the goal is to find a placement policy that maximizes the aggregate network utility, defined as a weighted sum of coverage and capacity over successive deployments. The optimization objective is:

$$\max_{(i,j)kk=1^M} \sum_{k=1}^{M} \left(V_k + \nu C_k\right), \quad \text{s.t. } (i,j)_k \in \mathcal{B}, \tag{1}$$

where V_k and C_k denote the incremental coverage and capacity gains achieved after the k-th BS is placed (given the existing deployed set), and ν is a scalar hyperparameter governing the trade-off between the two objectives.

AutoBS treats this problem within the framework of DRL. At each time step, an agent observes the current environment state—comprising the site map and deployment history—and selects a placement action $(i,j) \in \mathcal{B}$. Through episodic interaction and reward feedback based on V_k and C_k, the policy is progressively refined to produce efficient BS deployment strategies tailored to site-specific characteristics.

[1] Selected results and codes are available at github.com/abman23/autobs.

2.2. Reward Calculation via Digital Network Twins

In DRL, efficient reward calculation is essential for guiding the learning process, where the agent explores numerous actions throughout training. However, calculating network performance metrics, such as coverage and capacity, based on BS placements—key components of the reward function—is computationally intensive. Traditional RT simulations, which can take tens of minutes to generate a single pathloss map, are impractical for real-time reinforcement learning.

To overcome this challenge, AutoBS integrates *PMNet* (Lee & Molisch, 2024)– a generative DNT model – for rapid reward computation. PMNet can predict a site-specific pathloss map within milliseconds (with an RMSE on the order of 10^{-2}), enabling the PPO algorithm (Schulman et al., 2017) to immediately evaluate network performance and obtain a reward after each deployment decision. This rapid feedback loop allows the agent to efficiently simulate multi-BS deployment scenarios, significantly accelerating training by exploring many more placement options. In summary, PMNet enables fast and scalable reward evaluation, allowing AutoBS to operate in large, diverse network environments.

3. AutoBS: An Autonomous Base Station Deployment Framework

3.1. Architecture

AutoBS uses a deep RL agent based on the PPO algorithm to automate optimal BS placement in 6G networks. PPO is well-suited to this task because it balances exploration (testing new deployment locations) and exploitation (refining placement strategies). By using a clipped objective function, PPO also ensures stable policy updates and prevents overly large, destabilizing changes during training.

The framework supports two deployment modes: (1) *Static single BS deployment*, where one BS is optimally placed in a fixed environment; and (2) *Asynchronous multi-BS deployment*, where multiple BSs are deployed asynchronously.

3.1.1. SINGLE (STATIC) BS DEPLOYMENT

In the *static* single BS scenario, the environment and network conditions remain fixed (time-invariant). The input to the agent is a site-specific map $\mathcal{S}$ containing details about buildings, terrain, and obstacles. The PPO agent processes this map and outputs the optimal coordinates (i, j) for BS placement to maximize coverage and capacity.

3.1.2. MULTI (ASYNCHRONOUS) BS DEPLOYMENT

In the *asynchronous* multi-BS deployment scenario, BSs are deployed incrementally over time. After each deployment, the environment is updated to reflect the changes in network conditions, such as coverage, capacity, and user distribution. The PPO agent adjusts its strategy after each deployment based on real-time feedback, refining its decisions progressively. This scenario effectively models real-world situations where networks are expanded gradually (one BS at a time) due to evolving coverage/capacity demands or budget constraints. By deploying BSs sequentially and updating the environment each time, the agent can mimic practical network densification and optimize new BS locations while keeping previously deployed BSs fixed.

3.2. Evaluation Metrics

The effectiveness of a BS deployment strategy is assessed using two primary performance metrics: *Coverage* and *Capacity*. These capture distinct but complementary aspects of network quality: geographic service footprint and aggregate throughput.

Coverage. Coverage quantifies the fraction of the service region where users receive sufficient signal power. Let $\mathcal{R} \subseteq \mathcal{S}$ denote the set of user-accessible locations (e.g., non-building areas). For each point $(i, j) \in \mathcal{R}$, let $P_{i,j}$ be the received power from the deployed BSs, defined as the maximum signal strength over all BSs at that location:

$$P_{i,j} = \max_{k=1,\ldots,M} P_{i,j}^{(k)}, \tag{2}$$

where $P_{i,j}^{(k)}$ is the received power from the k-th BS at pixel (i, j). A location is considered covered if its received power exceeds a predefined threshold thr. The binary coverage indicator $v_{i,j}$ is defined as:

$$v_{i,j} = \begin{cases} 1, & P_{i,j} \geq \mathrm{thr}, \\ 0, & P_{i,j} < \mathrm{thr}. \end{cases} \tag{3}$$

The total coverage score V is then computed as:

$$V = \sum_{(i,j)\in\mathcal{R}} v_{i,j}. \tag{4}$$

The threshold thr is the minimum required received power, typically set to -90 dBm (i.e., $\approx 10^{-12}$ W), corresponding to a common planning threshold for acceptable Quality of Service (QoS). This value can be adapted to reflect more stringent or relaxed link requirements.

Capacity. Capacity measures the total theoretical data throughput over the service region, determined by the signal-to-noise ratio (SNR) experienced at each location. It is defined as:

$$C = \sum_{(i,j)\in\mathcal{R}} \log_2\left(1 + \mathrm{SNR}i,j\right), \tag{5}$$

where $\mathrm{SNR}i,j$ is the SNR at location (i,j), computed as:

$$\mathrm{SNR}i,j = \frac{Pi,j}{\sigma^2}. \tag{6}$$

Here, $P_{i,j}$ is the strongest received power as defined above, and σ^2 is the noise variance. To ensure a minimum SNR of at least 6 dB at the coverage threshold, we set $\sigma^2 = \mathrm{thr}/4$ in simulations. [2]

Both metrics are evaluated over the same user region $\mathcal{R}$. While coverage and capacity are often correlated (e.g., higher coverage generally implies improved SNR), they emphasize different deployment priorities. Hence, both are included in the reward to promote balanced network behavior.

3.3. Training

The AutoBS framework models the base station (BS) deployment problem as a Markov decision process (MDP), enabling the agent to interact with its environment to make optimal BS placement decisions.

3.3.1. MDP Design

Environment. The environment for the PPO agent is defined by the interactions within a site-specific building map, where each deployed BS impacts the network performance metrics such as coverage and capacity. At each time step t, the agent interacts with this environment by observing a state $s_t \in \mathcal{S}$, which includes details on buildings, obstacles, and terrain specific to the deployment site. The agent then takes an action $a_t \in \mathcal{A}$, selecting a BS deployment location based on its learned policy π. After the action is executed, the environment transitions to a new state s_{t+1}, representing the updated network layout and performance with the new BS in place. The agent receives a reward r_t based on the effectiveness of its decision, guiding it toward an optimal deployment policy π^*.

State. The state s_t at time step t provides key information for decision-making about BS placement. Formally, the state is expressed as:

$$s_t = \{\mathcal{S}\}, \tag{7}$$

where $\mathcal{S}$ represents the site-specific building map. This map encompasses critical details about building locations, obstacles, and terrain, all of which significantly influence signal propagation (*i.e.*, site-specific channel).

Action. The action space $\mathcal{A}$ consists of potential BS deployment locations. At each time step t, the agent selects an action $a_t \in \mathcal{A}$, which corresponds to placing a BS at a specific geographical coordinate (i,j) on the site-specific map. The action is represented as:

$$a_t = (i,j), \quad \{i,j\} \in \mathcal{B}, \tag{8}$$

where $\mathcal{B}$ is the set of all permissible deployment locations within the map. The agent deploys BSs sequentially, choosing new locations at each time step based on current network needs. This approach is computationally efficient, allowing the agent to adapt its strategy in real-time as it learns from previous decisions.

[2] While actual noise variance depends on factors such as system bandwidth, antenna gain, and receiver noise figure, we adopt a normalized value to match common cell-edge SNR planning assumptions.

Reward. The reward function r_t incentivizes the agent to improve network performance by maximizing both coverage and capacity. The reward at time step t is defined as:

$$r_t = \nu_1 V_t + \nu_2 C_t + \nu_3 P_t, \tag{9}$$

where V_t and C_t represent the improvements in coverage and capacity, respectively, and $P_t = \sum_{\{i,j\}\in\mathcal{R}} P_{i,j}$ denotes the total pathgain, for the t-th deployment. While the term P_t is not explicitly included in the objective function in (1), it strongly correlates with both coverage and capacity, providing a denser learning signal that accelerates training convergence. The weighting parameter ν adjusts the relative importance of coverage, capacity, and pathgain in the overall network optimization. For further details regarding the reward design, please refer to Appendix A. Note also that for more general DNT applications (*e.g.*, incorporating user demands), only the reward function needs to be adapted, while the remaining framework of AutoBS stays the same.

As mentioned in Sec. 2.2, the reward must be recalculated after each BS deployment to reflect the updated network conditions. This typically requires coverage and capacity evaluation from pathloss map prediction, which are computationally expensive. To overcome this challenge, AutoBS framework integrates the *PMNet* model, which provides fast and accurate pathloss predictions, enabling efficient reward calculations. PMNet allows the agent to quickly assess the network performance after each BS placement, ensuring that the training process is both realistic and computationally feasible.

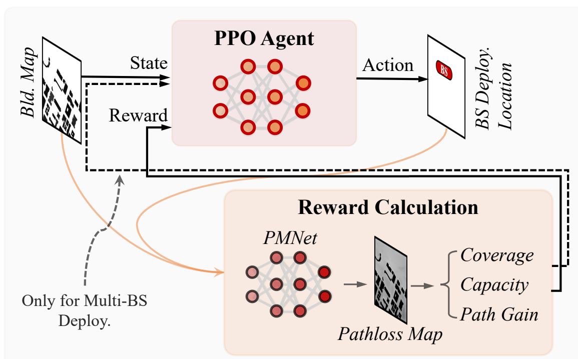


Figure 2. Training process for AutoBS framework, with PMNet providing generative pathloss predictions to compute the reward at each step.

3.3.2. Training Process

The PPO agent is trained using the PPO algorithm (Schulman et al., 2017), which allows the agent to iteratively interact with a simulated environment. The agent selects BS deployment actions and receives rewards based on improvements in network coverage and capacity, progressively refining its policy $\pi_\theta(a_t|s_t)$ over time.

At each time step t, the agent updates its policy to maximize the cumulative reward:

$$J(\theta) = \mathbb{E}\left[\sum_{t=0}^{T} \gamma^t r_t\right], \tag{10}$$

where γ is the discount factor, r_t is the reward received at time step t, and T is the time horizon over which the agent aims to optimize its decisions. The discount factor γ balances immediate and future rewards, promoting long-term planning in BS placement.

The PPO agent optimizes the following objective function:

$$\mathcal{L}(\theta) = \mathbb{E}_t\left[\min\left(r_t(\theta)\hat{A}_t,\, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t\right)\right], \tag{11}$$

where $r_t(\theta)$ is the probability ratio between the updated and previous policies, ϵ is a clipping parameter to limit policy changes for stability, and $\hat{A}_t$ represents the advantage function, which quantifies how much better the current action is compared to the baseline.

For clarity and space reasons, we omit detailed descriptions of the PPO algorithm, and instead present a summary of the training process in Fig 2, which outlines the key steps involved in training the PPO agent.

4. Experimental Results

4.1. Simulation Setup

The simulation environment is based on site-specific maps of the University of Southern California (USC) campus, covering an area of 512×512 [m^2]. This environment reflects dense urban features, including buildings and terrain, providing a realistic scenario to evaluate AutoBS.

Simulations are executed on a high-performance machine with a Tesla T4 GPU, 12GiB of RAM, and an Intel Xeon CPU @ 2.20GHz. Developed in Python with PyTorch for model training, the framework leverages PMNet for pathloss predictions.

Table 5 in Appendix B summarizes the key simulation parameters, including network configurations, environmental factors, and training setup.

Benchmark. To evaluate the performance of the proposed AutoBS framework, we compare it against two baseline methods: a heuristic-based deployment and an exhaustive search strategy. These baselines serve as the lower and upper bounds for performance comparison.

- **Heuristic**: This baseline simply places BSs uniformly at random within the deployable area B, with no learning or optimization. Such a random-placement approach serves as a performance lower bound since it ignores all site-specific information. (Notably, random spatial deployment is a common assumption in stochastic geometry models (Haenggi, 2012).)
- **Exhaustive**: Exhaustive search evaluates every possible BS placement configuration on the map and thus provides a performance upper bound (it finds the globally optimal placement for a given metric). However, this approach is *computationally prohibitive* – especially for multiple BSs, where complexity grows exponentially with the number of BSs and the map size. For practicality, we limit the exhaustive search in our experiments to evaluating 50 representative placements (rather than the full search space) as the multi-BS baseline. Additionally, our exhaustive baseline places multiple BSs *synchronously* (optimizing all placements jointly), unlike AutoBS's sequential addition. One could conceive of an "asynchronous" greedy exhaustive approach to speed it up, but it would still be extremely costly and not guarantee optimal results. We therefore use the synchronous exhaustive method as the true upper bound for comparison.
- **AutoBS (Proposed)**: AutoBS uses a PPO-based agent to make near-optimal BS placement decisions in real time. The agent's policy network is a fully-connected neural network with four hidden layers of 128 nodes each, trained to jointly optimize coverage and capacity. As a result, AutoBS can output deployment decisions on the order of only a few milliseconds.

4.2. Performance Analysis

Scheme	*Coverage* (V)	*Capacity* (C)
Heuristic	43.37%	1.892
AutoBS	59.44%	2.893
Exhaustive for V	64.23%	2.981
Exhaustive for C	63.08%	3.036

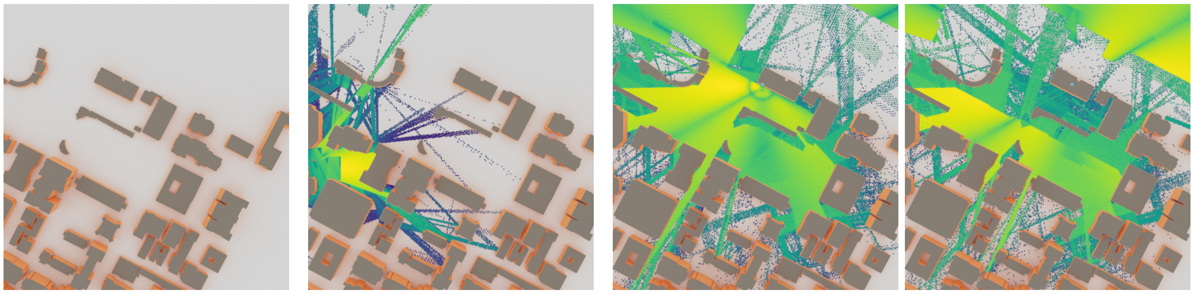
Table 1. Comparison results for Single (Static) BS deployment between exhaustive, random, and AutoBS deployment.

Comparison. Table 1 presents the performance of Heuristic, Exhaustive, and AutoBS deployments. AutoBS achieves significant improvements in both coverage and capacity compared to Heuristic deployment. Its performance closely approaches that of the Exhaustive method, which represents the global optimal solution, highlighting the effectiveness of policy-guided BS placement.

AutoBS focuses on coverage optimization through a reward function based on pathloss predictions from PMNet. Although capacity metrics are not explicitly included in the reward function (*i.e.*, $\nu_2 = 0$), the observed capacity difference between AutoBS and Exhaustive search remains minimal—which will be further discussed in Appendix A.

Coverage. Fig. 3 presents the coverage results for single BS deployment.[3] The building map, representing the input state, includes key geographical features such as buildings and obstacles, which significantly affect BS placement and signal propagation.

[3]The plots are generated with SionnaRT purely because of its nicer graphical representation (the underlying numerical results are identical to those obtained with PMNet).



(a) Building Map (b) Heuristic (c) Exhaustive (d) AutoBS

Figure 3. Coverage comparison for Single (Static) BS deployment. Simulations are performed using *WirelessInsite* and visualized with *SionnaRT*. Light green areas indicate regions with higher received signal strength. Fig. 3a illustrates the building map $\mathcal{S}$ used in each state s_t.

In the *Heuristic* method, BSs are placed randomly across the deployable area, leading to inefficient coverage with noticeable gaps. This highlights the limitations of non-optimized deployments. The *Exhaustive* search method evaluates every possible BS placement option, achieving the best coverage by systematically considering all configurations, but at the price of much higher complexity.

The *AutoBS* method, leveraging the PPO algorithm and fast, accurate pathloss predictions from PMNet for efficient reward calculations, achieves coverage performance approaching that of exhaustive search, but with significantly reduced computational overhead.

Convergence. Fig. 4 shows the training convergence for the single-BS deployment scenario, comparing AutoBS with the baseline methods. (Note that the Heuristic and Exhaustive approaches do not involve iterative learning, so their performance is flat.) AutoBS's performance improves steadily and stabilizes after approximately 50–100 training episodes. Although a performance gap remains compared to the global optimum (Exhaustive), AutoBS achieves close-to-optimal performance efficiently, demonstrating effective learning and stable convergence.

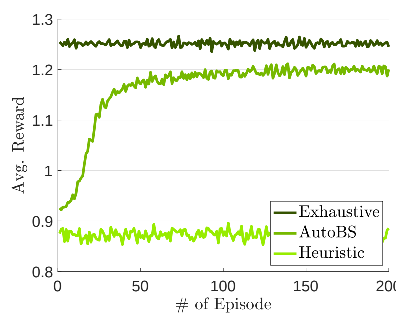


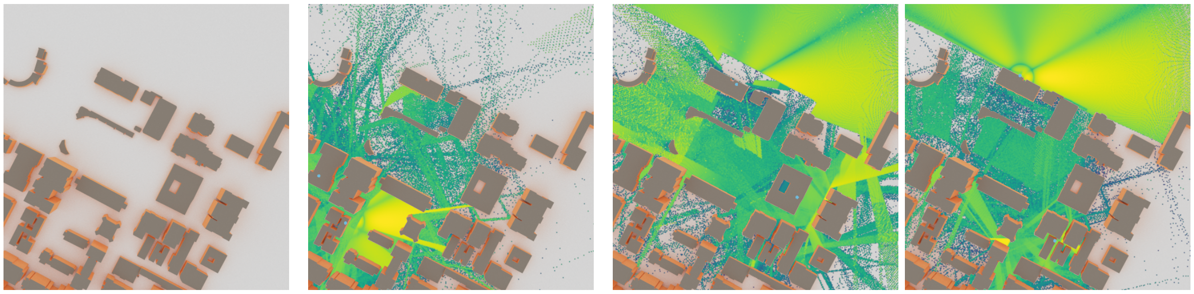
Figure 4. Convergence behavior for Single (Static) BS deployment for Heuristic, Exhaustive, and AutoBS deployments over 200 steps.

Multi-BS Deployment (Asynchronous). Table 2 and Fig. 5 summarize the results for the asynchronous multi-BS scenario. Note: In this setting, the reward at each step accounts only for the newly deployed BS's contribution (*e.g.*, the coverage, capacity, and/or path gain added by that BS). For example, when deploying a second BS, only the second BS's performance metrics are used to calculate the reward for that step.

Scheme	*Coverage* (V)	*Capacity* (C)
Heuristic	63.95%	2.966
AutoBS	71.65%	3.637
Exhaustive for V	81.06%	3.995
Exhaustive for C	75.36%	4.023

Table 2. Comparison results for Multi (Asynchronous) BS deployment between exhaustive, random, and AutoBS deployment.

In our multi-BS experiments, AutoBS follows an asynchronous (greedy) deployment strategy, adding one BS at a time. This sequential approach is practical for scenarios like gradually adding BSs to an existing network. However, in a greenfield deployment (with no initial infrastructure, as studied here), a purely sequential addition can yield worse results than an



(a) Building map (b) Heuristic (c) Exhaustive (d) AutoBS

Figure 5. Comparison results for Multi (Asynchronous) BS deployment in terms of coverage using *SionnaRT*. Light green areas indicate higher received signal strength. Fig. 5a shows a building map $\mathcal{S}$ used in each state s_t.

ideal joint (synchronous) optimization. Developing methods to iteratively fine-tune or reposition previously placed BSs could mitigate this, and will be addressed in future work.

Overall, the multi-BS results align with the single-BS findings, but with a notable insight: using two randomly placed BSs (Heuristic) can perform no better – or even worse – than using just one well-placed BS (as AutoBS or Exhaustive would). In other words, simply deploying more BSs does not guarantee better performance without optimization. This highlights the critical importance of optimized deployment strategies. As seen by comparing Table 2 (Heuristic two-BS) with Table 1 (AutoBS/Exhaustive single-BS), a non-optimized approach can fall short even with additional infrastructure.

In summary, our multi-BS results highlight two key points: (1) AutoBS's sequential (asynchronous) strategy still achieves roughly 90% of the capacity of the global optimum (the exhaustive placement), and (2) by learning the site-specific propagation characteristics, AutoBS is able to deploy multiple BSs very efficiently (vastly outperforming unoptimized placements).

4.3. Complexity Analysis

Table 3 compares the computation time required by AutoBS versus an exhaustive search. The key takeaway is that AutoBS drastically reduces inference time — achieving decisions in milliseconds even for multi-BS deployments — whereas exhaustive search quickly becomes intractable as the number of BSs grows. The efficiency gains of AutoBS are especially dramatic in the multi-BS case (exhaustive search time explodes exponentially with each additional BS). AutoBS's sequential deployment strategy, on the other hand, scales easily with network size, demonstrating that it is well-suited for large networks and real-time optimization.

	One BS Deploy. [sec]	***Two BS Deploy.*** [sec]	***Three BS Deploy.*** [sec]
Exhaustive	3.2245 ($^{10000}/_{10000}$)	4062.1 ($^{1000000}/_{1000000}$)	13449 ($^{1000000}/_{1000000}$)
AutoBS	**0.0145** ($^{45}/_{10000}$)	**0.1608** ($^{39}/_{1000000}$)	**0.2128** ($^{15}/_{1000000}$)

Table 3. Time complexity analysis.

5. Conclusion

In this paper we presented AutoBS, a reinforcement-learning-based framework for optimal BS deployment in 6G RANs. By leveraging the PPO algorithm and integrating fast, accurate pathloss predictions from PMNet (a generative model for DNT), AutoBS efficiently learns deployment strategies that adapt to site-specific channel conditions while balancing coverage and capacity.

Our results show that AutoBS achieves *near-optimal* performance – up to $\sim 95\%$ of the capacity of exhaustive search – while *reducing inference time from hours to milliseconds*. This makes AutoBS a practical solution for real-time deployment. With its combination of near-optimal performance and efficient scalability, AutoBS stands out as a powerful automated solution tailored to the demands of large-scale 6G network optimization.

While we demonstrated AutoBS for 6G BS deployment, the underlying framework can be easily adapted to other network optimization tasks in DNT setting. By modifying the state representation and reward function, the same approach could tackle problems like mobility management, link adaptation, or BS sleep mode energy savings, aligning the optimization with specific application goals. Furthermore, AutoBS's flexible design makes it suitable for entirely different deployment scenarios – for example, optimal placement of Wi-Fi access points (APs) or O-RAN radio units (RUs).

Acknowledgements: Part of this work was supported by Nokia Technologies and NSF-AoF grant 2133655. The authors gratefully acknowledge Arjun Balamwar and Yanqing Lu for their contributions to the framework design and simulations.

References

Aoudia, F. A., Hoydis, J., Nimier-David, M., Cammerer, S., and Keller, A. Sionna RT: Technical report. *arXiv preprint arXiv:2504.21719 [cs.IT]*, 2025. URL https://arxiv.org/abs/2504.21719.

Cammerer, S., Aït Aoudia, F., Hoydis, J., Oeldemann, A., Roessler, A., Mayer, T., and Keller, A. A neural receiver for 5G NR multi-user MIMO. In *Proc. of IEEE GLOBECOM Workshops*, 2023.

Haenggi, M. *Stochastic geometry for wireless networks*. Cambridge University Press, 2012.

Hoydis, J., Cammerer, S., Ait Aoudia, F., and et al. Sionna: An open-source library for next-generation physical layer research. *arXiv preprint arXiv:2203.11854 [cs.IT]*, Mar. 2022. URL https://arxiv.org/abs/2203.11854.

Interdonato, G., Björnson, E., Quoc Ngo, H., Frenger, P., and Larsson, E. G. Ubiquitous cell-free massive MIMO communications. *EURASIP Journal on Wireless Communications and Networking*, 2019(1):1–13, 2019.

Lee, J.-H. and Molisch, A. F. A scalable and generalizable pathloss map prediction. *IEEE Trans. on Wireless Commun.*, pp. 1–14, 2024. doi: 10.1109/TWC.2024.3457431.

Lee, J.-H., Molisch, A. F., and et al. PMNet: Robust pathloss map prediction via supervised learning. In *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, 2023.

Molisch, A. F. *Wireless Communications: From Fundamentals to Beyond 5G*. Wiley-IEEE Press, 3rd ed. edition, 2023. ISBN 978-1-119-11720-9.

Pegurri, R., Linsalata, F., Moro, E., Hoydis, J., and Spagnolini, U. Toward digital network twins: Integrating sionna RT in ns-3 for 6G multi-RAT networks simulations. *arXiv preprint arXiv:2501.00372 [cs.NI]*, arXiv:2501.00372, 2025a. URL https://arxiv.org/abs/2501.00372.

Pegurri, R., Moro, E., Linsalata, F., Hoydis, J., and Spagnolini, U. Towards digital network twins: Full-stack and multi-stack solution for 6G simulations. In *Proc. of IEEE WCNC (Demo Track)*, 2025b. doi: 10.1109/WCNC61545.2025.10978768.

REMCOM. Wireless Insite. https://www.remcom.com/wirelessinsite.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347 [cs.LG]*, 2017. URL https://arxiv.org/abs/1707.06347.

Wiesmayr, R., Cammerer, S., Aït Aoudia, F., Hoydis, J., Zakrzewski, J., and Keller, A. Design of a standard-compliant real-time neural receiver for 5G NR. *arXiv preprint arXiv:2409.02912 [cs.IT]*, arXiv:2409.02912, 2024. URL https://arxiv.org/abs/2409.02912.

A. Reward Design

Reward	***Coverage*** (V)	***Capacity*** (C)
Capacity Only	57.642%	2.862
Coverage + Capacity	57.833%	2.831
Pathgain + Capacity	58.747%	2.883
Coverage Only	**59.537**$\%$	2.884
Pathgain + Coverage	59.438%	**2.893**

Table 4. Reward design variations for single (static) BS deployment.

This section presents empirical findings from experiments with different reward designs in AutoBS framework. The pathgain is defined as $P = \sum_{i,j\in\mathcal{R}} P_{i,j}$, and Table 4 summarizes coverage and capacity outcomes for each reward configuration. Although other transformations, such as logarithmic scaling (*e.g.*, $\log V$), were tested, they are omitted here for brevity. These findings are empirical and sensitive to training hyperparameters (*e.g.*, learning rate) and reward design parameters (*e.g.*, ν).

The results provide valuable insights. As expected, the Coverage Only reward achieves the highest coverage, aligning with its direct optimization objective (*e.g.*, $\nu_2, \nu_3 = 0$). Interestingly, in terms of capacity, the combination of Pathgain and Coverage (*i.e.*, $\nu_2 = 0$) outperforms the Capacity Only reward. Notably, Coverage Only also surpasses Capacity Only in capacity, highlighting complex dynamics within the DRL training process. These findings suggest that site-specific channel characteristics are implicitly embedded in rewards derived from PMNet's pathloss maps. While Capacity Only smooths variations through logarithmic scaling, Pathgain captures these fluctuations on a linear scale, providing more granular feedback for effective learning.

Throughout this work, the Pathgain + Coverage reward configuration is primarily used.

B. Setup

For all experiments, we used the same simulation configuration. Table 5 below summarizes the key parameters of this setup, divided into two parts: the wireless network environment settings and the training hyperparameters for AutoBS.

Network Config.	
Area	512×512 [m^2]
Carrier frequency	2.5 [GHz]
Effective bandwidth	1 [MHz]
Antenna	Isotropic (vertical)
Input power	0 [dBm]
Noise figure	3 [dB]
Coverage threshold	-90.015 [dBm]
AutoBS Train Config.	
Generative digital twin model	PMNet$_{\text{usc}}$ (Lee & Molisch, 2024)
DRL algorithm	PPO (Schulman et al., 2017)
Reward function	Pathgain + Coverage ($\nu_2 = 0$)
Learning rate	1.0×10^{-5}
Gamma	0.1
SGD minibatch size	256
# of samples for train (test)	1100 (50)
# of episode	5

Table 5. Simulation Parameters.