Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning

Enzo Charolois–Pasqua
Advanced Statistical Inference
EURECOM
enzo.charolois-pasqua@eurecom.fr

1 Introduction

Deep learning models are very powerful tools used accross many domains, but they often lack the ability to to capture uncertainty in their predictions, a limitation that could be critical in sensitive domains like autonomous driving. Traditional neural networks typically provide point estimates without indicating how confident the model is. In 2016, Gal and Ghahramani addressed this issue by showing that dropout, a common regularization technique, can be interpreted as approximate Bayesian inference, enabling uncertainty estimation through standard architectures with minimal extra cost[1].

Their approach, known as Monte Carlo (MC) Dropout, uses stochastic forward passes at test time to estimate predictive distributions, by computing the predictive mean and variance.

In this project, we reproduce key results from the paper through four experiments: an initial test to understand how MC dropout works, followed by experiments in regression, classification, reinforcement learning. Our goal is to implement the theory in practice and assess the value of MC Dropout for reliable predictions.

The implementation of the experiments is available at `https://github.com/elea-vellard/ASI-project_VELLARD_CHAROLOIS.git`.

2 Explanation of the paper

The paper aims to establish how dropout can serve as an approximate Bayesian inference method to represent uncertainty in deep learning models. Bayesian neural networks attempt to model the posterior distribution over the network weights given the observed data, $p(\theta|X, Y)$ but exact inference is computationally intractable due to the high dimensionality of the parameter space. The paper shows that dropout approximates this posterior by minimizing the Kullback-Leibler (KL) divergence between an approximate distribution $q(\theta)$ and the true posterior.

2.1 Dropout as Variational Inference

Dropout is a method in which we randomly deactivate a fraction of neurons during training, effectively creating multiple smaller neural networks by randomly removing units during training. The paper formalizes this as optimizing the following objective:

$$\mathcal{L}_{\text{dropout}} = \frac{1}{N} \sum_{i=1}^{N} \ell(y_i, f(x_i; \hat{\theta})) + \lambda \sum_j \|\theta_j\|^2, \tag{1}$$

[1] Gal, Y. and Ghahramani, Z. (2016). Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. *Proceedings of the 33rd International Conference on Machine Learning*, 1050–1059.

Preprint. Under review.

where ℓ is the loss function (e.g., the Mean Squared Error (MSE) for regression), $f(x_i; \hat{\theta})$ is the model's output with dropout applied on the parameters $\hat{\theta}$, and λ is a weight decay parameter. This objective is equivalent to minimizing the KL divergence between $q(\theta)$ and $p(\theta|X, Y)$.

Specifically, dropout defines $q(\theta)$ as a distribution composed of multiple networks with randomly deactivated units. The paper shows theoretically that this approximation is equivalent to performing inference in a deep Gaussian process, and that it can provide a formal explanation for the regularization effect of dropout.

2.2 MC Dropout for Uncertainty Estimation

While dropout is normally turned off at test time, the paper shows that keeping it active during inference allows us to estimate model uncertainty, which can be very useful in many applications. Indeed, MC Dropout performs T stochastic forward passes with dropout active to approximate the predictive distribution. The predictive mean $\mathbb{E}$ and variance *Var* are computed as:

$$\mathbb{E}[y^*|x^*] \approx \frac{1}{T}\sum_{t=1}^{T} f(x^*; \hat{\theta}_t), \tag{2}$$

$$\text{Var}[y^*|x^*] \approx \tau^{-1} I + \frac{1}{T}\sum_{t=1}^{T} f(x^*; \hat{\theta}_t)^2 - \left(\frac{1}{T}\sum_{t=1}^{T} f(x^*; \hat{\theta}_t)\right)^2, \tag{3}$$

where τ is the model precision (inverse variance of the observation noise), and $f(x^*; \hat{\theta}_t)$ is the output of the t-th forward pass with dropout mask $\hat{\theta}_t$. The term $\tau^{-1} I$ represents the noise in the data (we we can't reduce), and the variation between predictions shows how unsure the model is (what we can reduce by training with more data). This formulation allows MC Dropout to provide uncertainty estimates, differencing regions with high and low confidence.

2.3 Consequences of the paper

A key contribution of the paper is showing that model uncertainty can be estimated without changing the architecture or adding significant cost. By reusing standard dropout layers at test time, the method remains simple, efficient, and model-agnostic.

Its ability to quantify uncertainty has strong practical value, especially in high-stakes tasks like regression extrapolation or reinforcement learning, where confidence guides better decisions.

3 Experiments

3.1 Experiment 1: MC Dropout for Regression and Uncertainty Visualization

3.1.1 Motivation

This experiment is not a direct reproduction of the paper's results, but was a way for us to get used to use MC dropout for uncertainty, understand how to implement it and visualize results. To visualize this, we used a regression task using the California Housing dataset in order to visualize predictive uncertainty.

By performing multiple stochastic forward passes with dropout active at test time, the model provides both predictive means and uncertainties, crucial for applications where understanding model confidence is essential, such as financial forecasting or environmental modeling.

3.1.2 Implementation

We implemented a neural network with two hidden layers of 100 units each, using ReLU activations and a dropout rate of 0.1. The California Housing dataset was used, with standardized features and targets. The model was trained for 2,000 epochs using the Adam optimizer (learning rate 0.01, weight

decay 1e-2) and mean squared error loss. At test time, 100 stochastic forward passes were performed to compute the mean and standard deviation of predictions.

3.1.3 Results

The model achieved a root mean squared error (RMSE) of 0.58 on the test set, demonstrating strong predictive accuracy.

The first visualization (cf. figure 1) displays the predicted house prices along with a $\pm 2\sigma$ uncertainty band for 300 test samples (so less samples to improve readability). It shows the uncertainty across different samples, and the uncertainty bands widen in areas where the model is less certain.

The second plot (cf. figure 2) is a histogram of the predicted standard deviations, giving an overview of how uncertainty is distributed across the test set. It confirms that MC Dropout effectively captures model confidence: most predictions have low uncertainty, indicating high confidence, while a smaller number of samples show higher uncertainty, often corresponding to ambiguous or unfamiliar inputs.

3.2 Experiment 2: Model Uncertainty in regression tasks

3.2.1 Motivation

This experiment aims to show how MC Dropout can effectively model predictive uncertainty in a regression task, recreating the results implemented in the paper's Figure 2, that highlights the predictive mean and uncertainties on the Mauna Loa CO_2 concentrations dataset, for various models. Thus, we aim to compare a few models based on their predictions and uncertainty estimates when trying to estimate future and unseen values for this dataset. The compared models are:

- A standard dropout model used in deterministic mode for reference;
- A Gaussian Process with an RBF (Radial basis function) kernel, which is known for principled uncertainty estimation;
- Two MC Dropout models with different activation functions (ReLU and Tanh).

To compare their behavior, we run multiple stochastic forward passes with MC Dropout to visualize and assess each model's confidence, especially beyond the training data, and we will try to understand the impact of non-linearities on MC Dropout uncertainty.

3.3 Implementation

We trained three neural networks on the Mauna Loa CO_2 dataset: one with standard dropout (used in evaluation mode, without uncertainty), one with ReLU activation, and one with Tanh activation, each with two hidden layers of 100 units and a dropout rate of 0.1. Both models were trained for 2,000 epochs using the Adam optimizer. Uncertainty was estimated with 100 stochastic forward passes. We also train a Gaussian Process with an RBF kernel as a reference, since it provides exact predictive uncertainty. Comparing it with three dropout-based models allows us to evaluate how well MC Dropout approximates Bayesian behavior in regression.

3.4 Results

Results are shown in figure 3. In chart (a), the standard dropout model extrapolates into regions with no training data, with no uncertainty provided. In contrast, the Gaussian Process in (b) shows smooth predictions with increasing uncertainty beyond the training range, which is an expected and desirable Bayesian behavior.

Charts (c) and (d) show MC Dropout with ReLU and Tanh activations, respectively. Both capture rising uncertainty in the extrapolation zone, consistent with the Gaussian Process. The ReLU model shows unbounded uncertainty growth, that increases as we move further away from the training data, highlighting the model's awareness of its own limitation. However, we can see that the Tanh model produces "flat" output. This is because the network has not learned to extrapolate beyond the data seen during training, and Tanh activation saturate quickly (around ± 1), so outside the training area, neurons produce almost constant values. Furthermore, because of its saturating nature, Tanh produces bounded uncertainty.

3.5 Differences with the Paper

Compared to the original paper, our experiment uses a simpler setup: we fixed the dropout probability to 0.1, used 4 hidden layers (instead of 5), and 100 Monte Carlo samples (instead of 1000) for uncertainty estimation. Additionally, we applied a single split (75/25) of the Mauna Loa CO_2 data instead of multiple runs or repeated experiments. Despite these simplifications, our results qualitatively match the paper's findings in terms of uncertainty behavior.

3.6 Experiment 3: Model uncertainty in classification tasks

3.6.1 Motivation

After exploring model uncertainty in regression tasks, we now turn to classification. Inspired by the paper's MNIST experiment, this study evaluates MC Dropout's ability to capture uncertainty in a convolutional neural network (CNN) for handwritten digit recognition. By performing multiple stochastic forward passes, we track how the model's confidence changes as an image of a digit is gradually rotated, making it increasingly difficult to classify.

3.6.2 Implementation

We trained a LeNet-style CNN on the MNIST dataset, applying dropout (probability 0.5) before the final fully connected layer, as described in the paper. The model was trained for 1,000,000 iterations using a learning rate of 0.0001 and a decay factor of 0.75. We evaluated uncertainty by performing 100 stochastic forward passes on a rotated digit "1" image, scattering the softmax inputs and outputs to visualize class uncertainty, following the paper's methodology. We observe the uncertainty evolution for 3 digit classes: 1, 5 and 7, as in the paper.

3.6.3 Results

The first plot (cf. figure 4) indicates which class is predominant in the classification estimate, while the second plot (cf. figure 5) shows how the uncertainty evolves across rotations, denser points indicating lower uncertainty, while very sparse points indicate a very high uncertainty.

As shown in those two graphs, the model correctly classified the digit '1' for initial rotations with high certainty, but showed increased uncertainty (larger variance in softmax outputs) as the rotation angle increased, misclassifying it as the digits '5' and '7' (graph a) with high uncertainty (graph b). This aligns with the paper's observation that MC Dropout captures uncertainty when softmax inputs for different classes overlap, indicating low confidence in predictions.

3.6.4 Differences with the Paper

Compared to the paper, we used PyTorch instead of Caffe, trained for fewer epochs, and evaluated on 12 rotated images with more MC samples. Despite these changes, we reproduced the key result: model uncertainty grows as the digit becomes harder to classify.

3.7 Experiment 4: MC Dropout in Reinforcement Learning

3.7.1 Motivation

The goal of this final experiment is to reproduce the reinforcement learning experiments of the paper, that shows MC Dropout's uncertainty estimate can help achieve a better balance between exploration and exploitation and allows for a faster and more efficient learning than baseline exploration techniques in RL set-ups. Indeed, MC dropout allows to gain an estimate of the uncertainty on the Q-values, allowing to explore in an intelligent way: uncertain actions are explored more often, while certain actions are chose, and the exploration becomes not arbitrary.

To demonstrate this, we compare two methods: Thompson Sampling with MC Dropout, which leverages model uncertainty to guide exploration, and Epsilon-Greedy, which explores randomly with a fixed probability. This experiment replicates the paper's approach in a simplified 2D reinforcement learning task to validate the benefits of uncertainty-aware exploration.

3.8 Implementation

We simplify the original RL setup by using a 20-arm Bernoulli bandit, keeping the core exploration–exploitation challenge without complex dynamics. Each arm has a fixed reward probability, and the agent must learn to favor the best ones. We use a 1-hidden-layer MLP with dropout to estimate success probabilities. Epsilon-Greedy explores randomly, while Thompson Sampling uses MC Dropout to guide exploration. Results are averaged over three runs on the same bandit. For visualization purposes, we plotted in figure 6 the 20-arm environment, indicating the reward probabilities of each arm.

3.9 Results

The plot in figure 7 in the appendix shows that Thompson Sampling with MC Dropout (blue) outperforms Epsilon-Greedy (green) at the beginning of the learning, allowing to reach higher average rewards more quickly (around 1000 steps before the Epsilon-Greedy policy). This indicates that leveraging uncertainty helps the agent explore more effectively, focusing on less certain actions and learning faster. While both methods eventually converge to similar performance, MC Dropout enables faster learning in the first phases by making exploration adaptive rather than random.

3.10 Differences with the Paper

The main difference lies in the experimental setup: the paper uses a custom 2D RL environment from Karpathy et al. (2014–2015)[2], while we opted for a simpler but conceptually similar setup to reduce complexity.

4 Conclusion

Our experiments reproduced key aspects of Gal and Ghahramani, demonstrating that MC Dropout provides effective uncertainty estimates various tasks, and enhances exploration in reinforcement learning. While the experiments we conducted were not exactly similar to those exploited in the paper, they allowed us to both induce and understand this method, leaving us with a powerful tool that could be used in many real-world applications.

Future work could explore larger models, which we avoided due to computational constraints, additional datasets, more complex environments (especially in reinforcement learning), or advanced dropout variants such as Gaussian dropout, in order to further validate and extend the paper's findings.

5 Appendix

[2]Karpathy, A and authors. A Javascript implementation of neural networks. `https://github.com/karpathy/convnetjs`, 2014–2015.

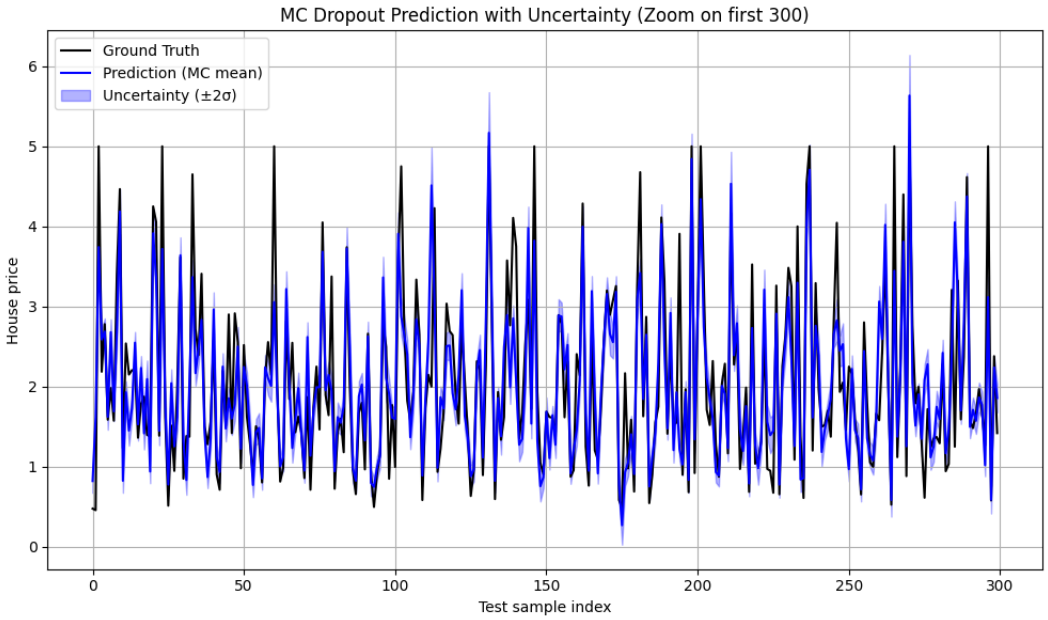

Figure 1: Comparison between ground truth house prices (black) and MC Dropout predictions (blue), with uncertainty represented by a $\pm 2\sigma$ confidence band. The model closely follows true values while adapting the uncertainty level to local prediction confidence.

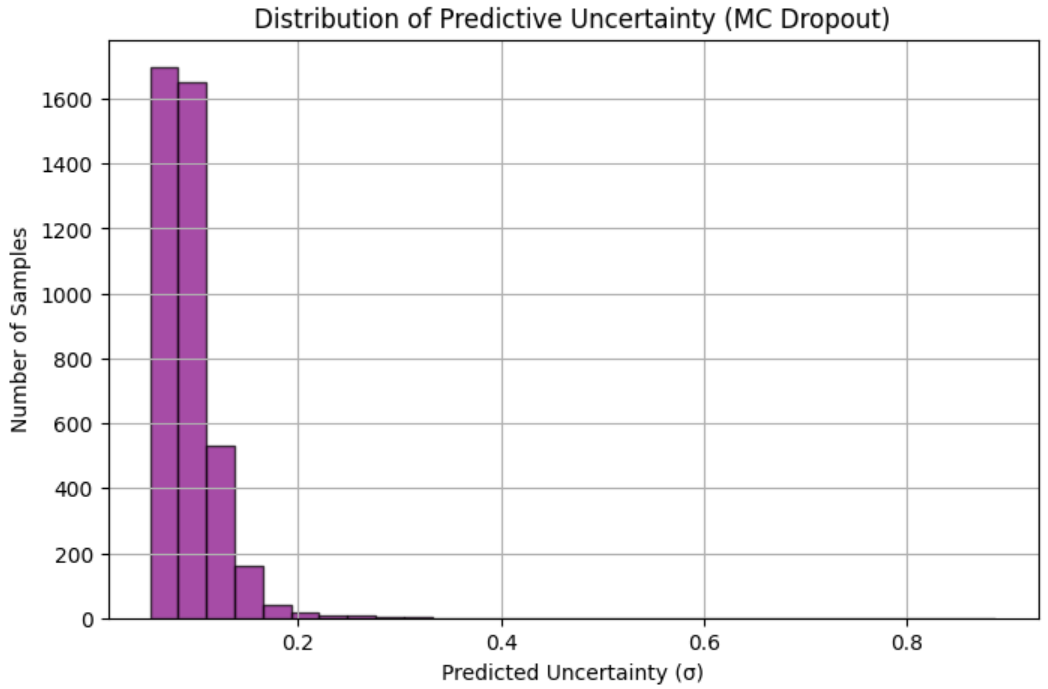

Figure 2: Histogram of predicted standard deviations (σ across test samples, showing that most predictions have low uncertainty. A small number of samples have higher uncertainty, indicating cases where the model is less confident.

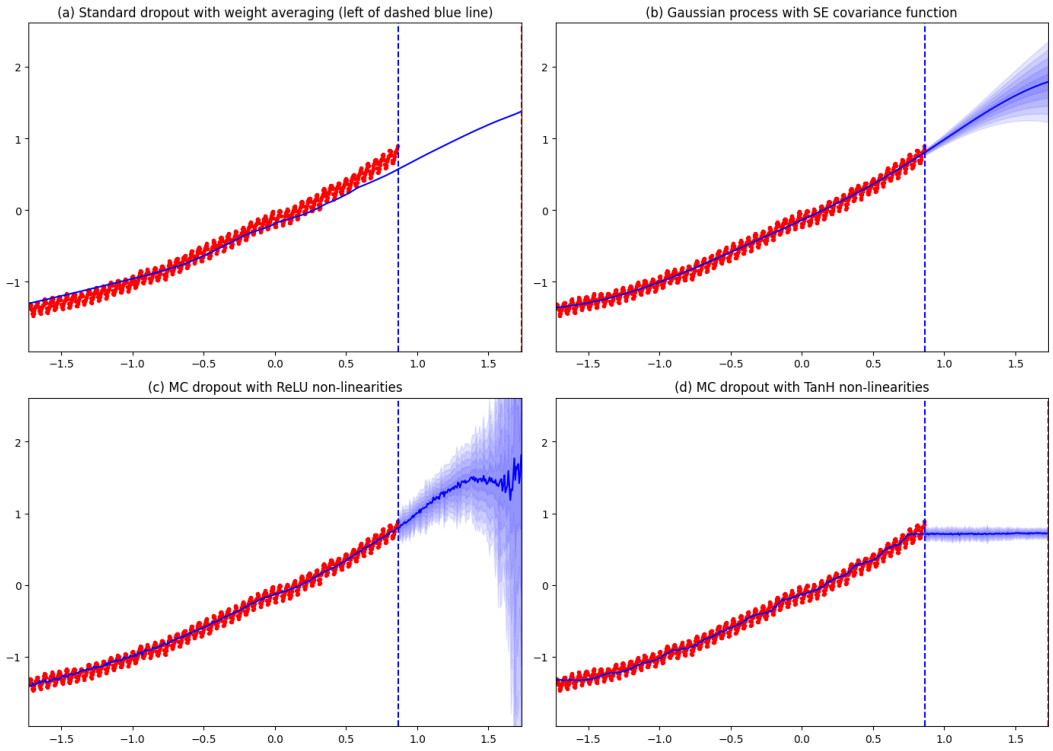

Figure 3: Comparison of predictive uncertainty across models in a regression task. Standard dropout (a) does not show uncertainty beyond the training range, while the Gaussian Process (b) and MC Dropout with ReLU (c) and Tanh (d) capture rising uncertainty in unseen regions, each showing different extrapolation behaviors.

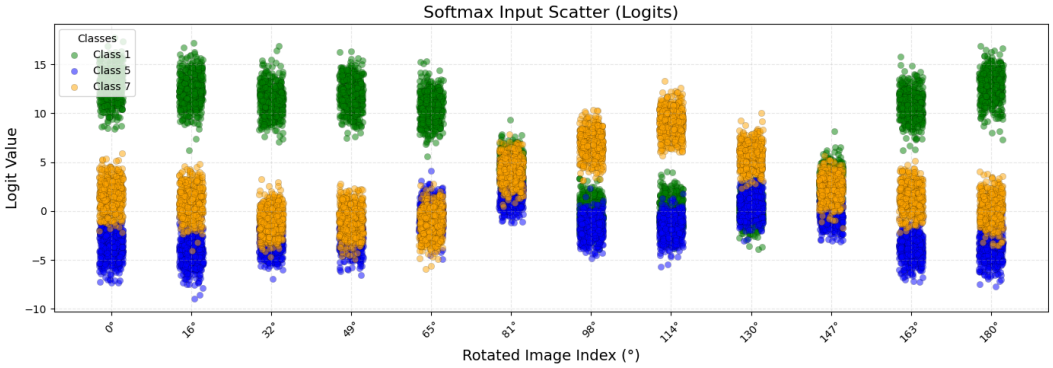

Figure 4: Softmax input scatter across rotated inputs (logits): Distribution of raw class scores (logits) for classes 1, 5, and 7 as the input image is progressively rotated. High and separated logits indicate confident predictions, while overlapping logits reflect uncertainty, that rises with rotation.

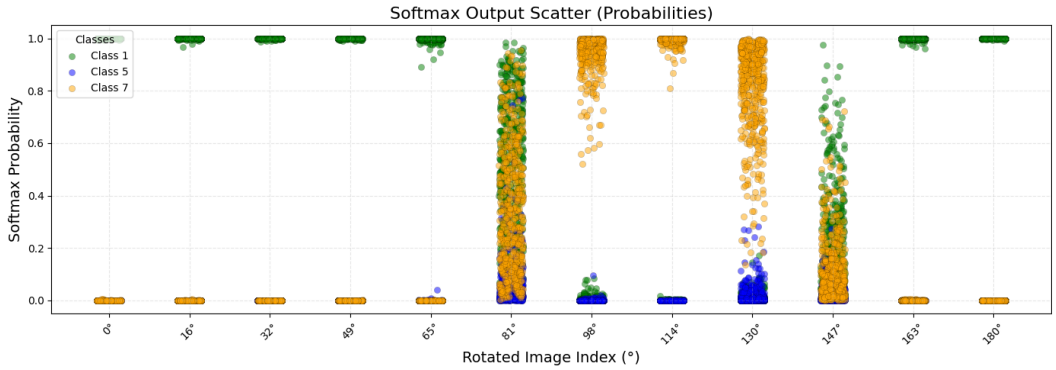

Figure 5: Softmax output scatter across rotated inputs. Softmax probabilities for classes 1, 5, and 7 across increasing rotation angles of a digit image of 1. High confidence is shown by concentrated points near 1.0, while scattered outputs reflect model uncertainty.

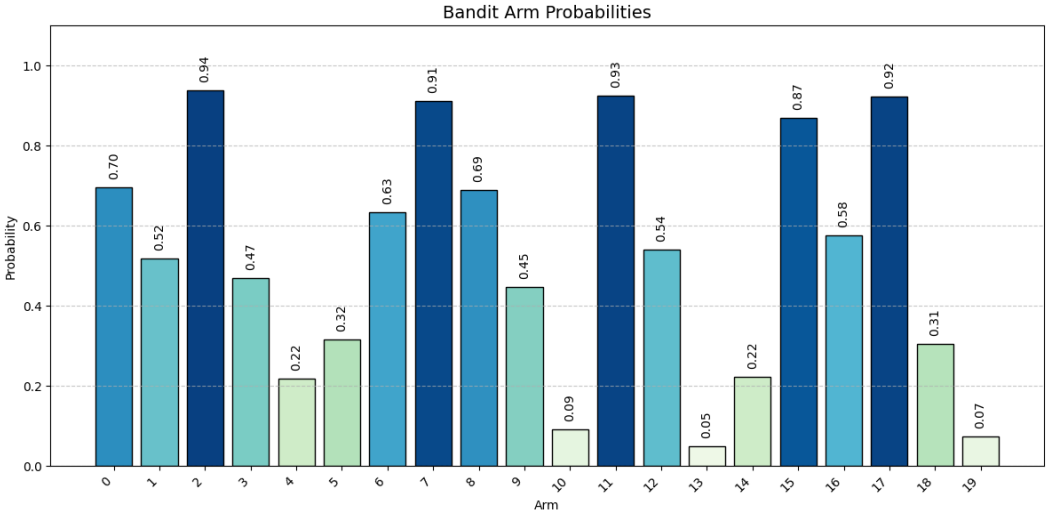

Figure 6: Experiment environment: reward probabilities for each bandit arm. Higher bars indicate arms with higher chances of yielding a reward and thus form the target of optimal exploration strategies.

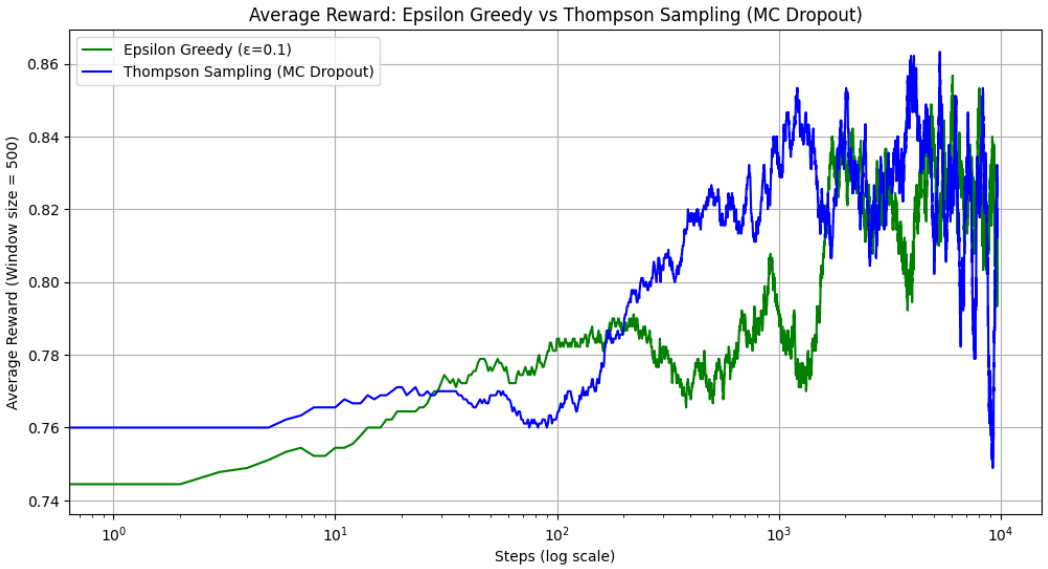

Figure 7: Comparison of average reward (over a window of 500 steps) between Epsilon-Greedy (green) and Thompson Sampling with MC Dropout (blue) in a 20-arm bandit environment. The log-scale x-axis shows that Thompson Sampling leads to faster early learning and reaches higher rewards more quickly, highlighting the interest of uncertainty-aware exploration.