

Energy-Efficient AI Using Stochastic Magnetic Tunnel Junctions

SEMICON EUROPA 2024

Nicolas Alder

Chair for Artificial Intelligence and Sustainability

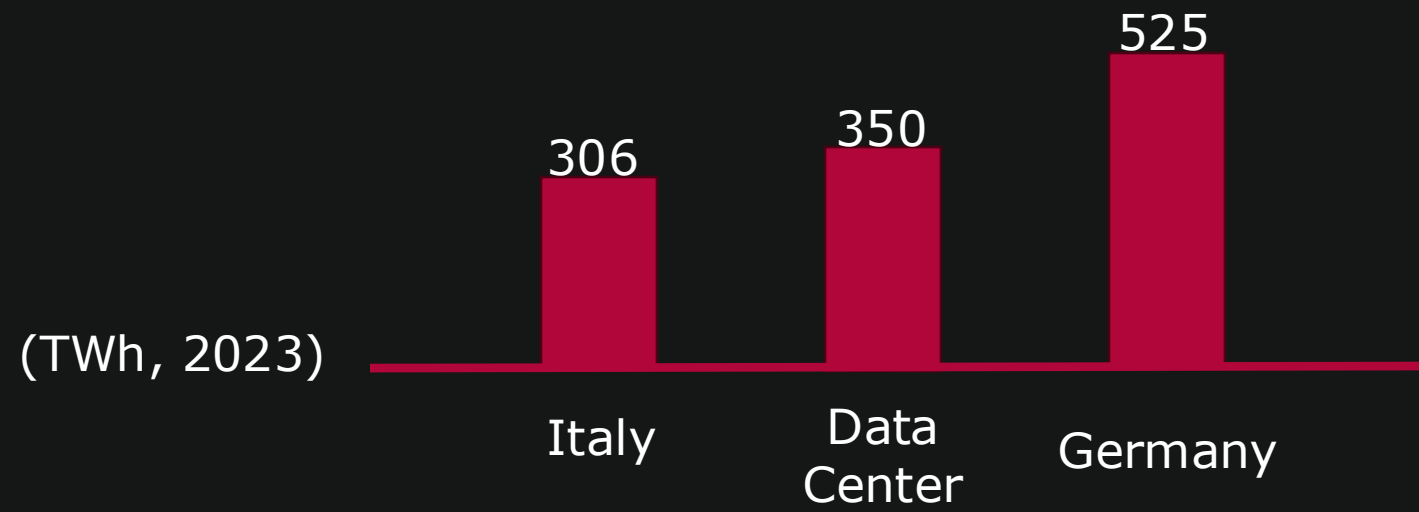
Hasso Plattner Institute

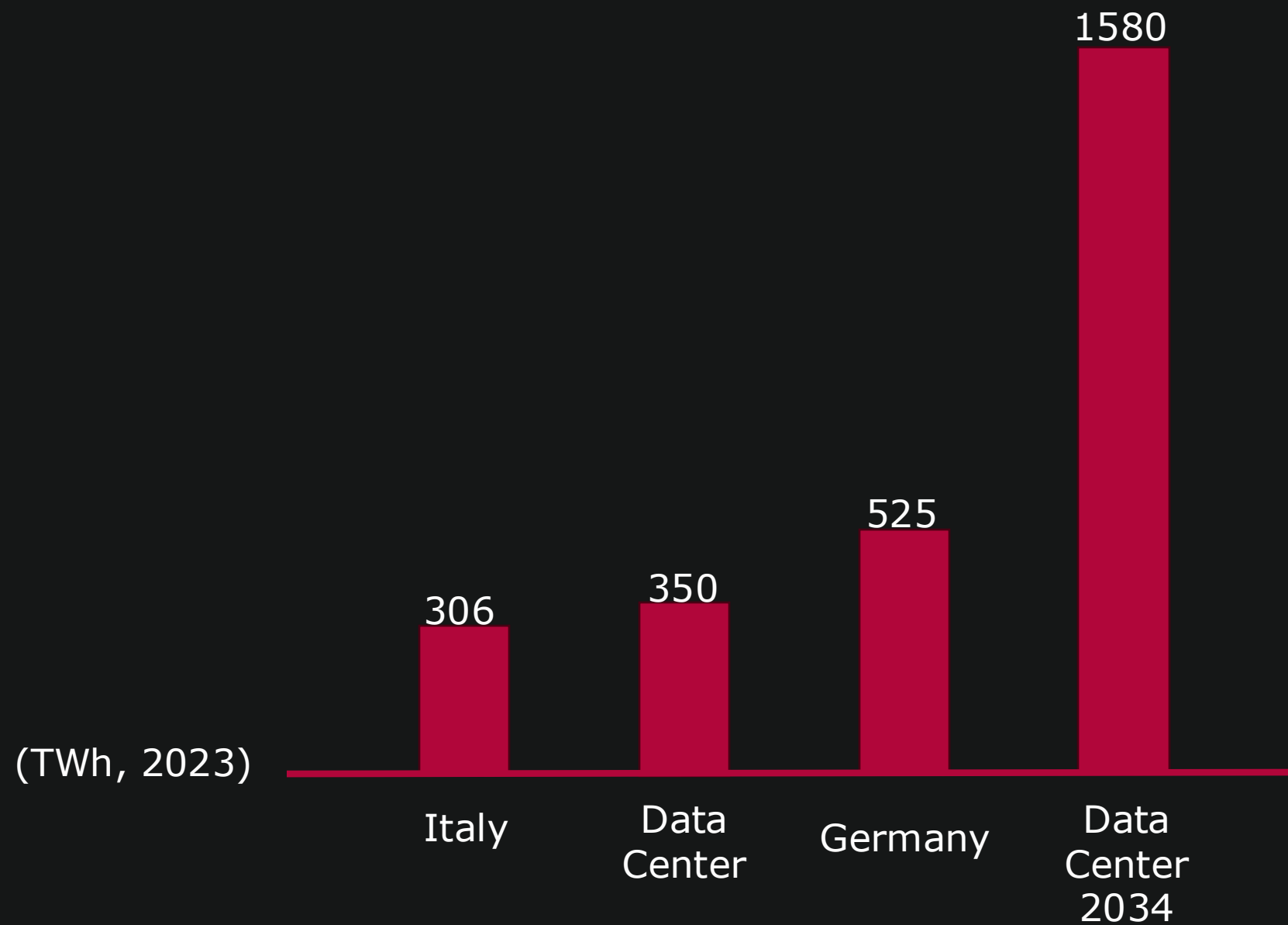
**Design IT.
Create Knowledge.**

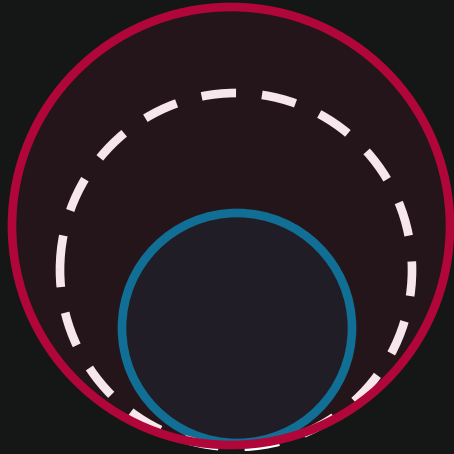
www.hpi.de



**Data center doubled
to 7,000 since 2015.**







Ireland

Planned Consumption (2034)

Renewable Energy Supply (2022)

Live Consumption (2022)

BBC

[Home](#) [News](#) [US Election](#) [Sport](#) [Business](#) [Innovation](#) [Culture](#) [Arts](#) [Travel](#) [Earth](#) [Video](#) [Live](#)

Google turns to nuclear to power AI data centres

7 days ago

Share  Save 

João da Silva
Business reporter

Taiwan eyes new nuclear energy plans due to demands of the AI boom

News

By [Mark Tyson](#) published 18 hours ago

Chipmakers assured us that current electricity production methods will be able to cope until 2030.

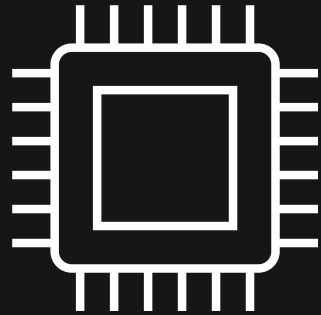
Nuclear power for AI data centers: Microsoft has Three Mile Island reactivated

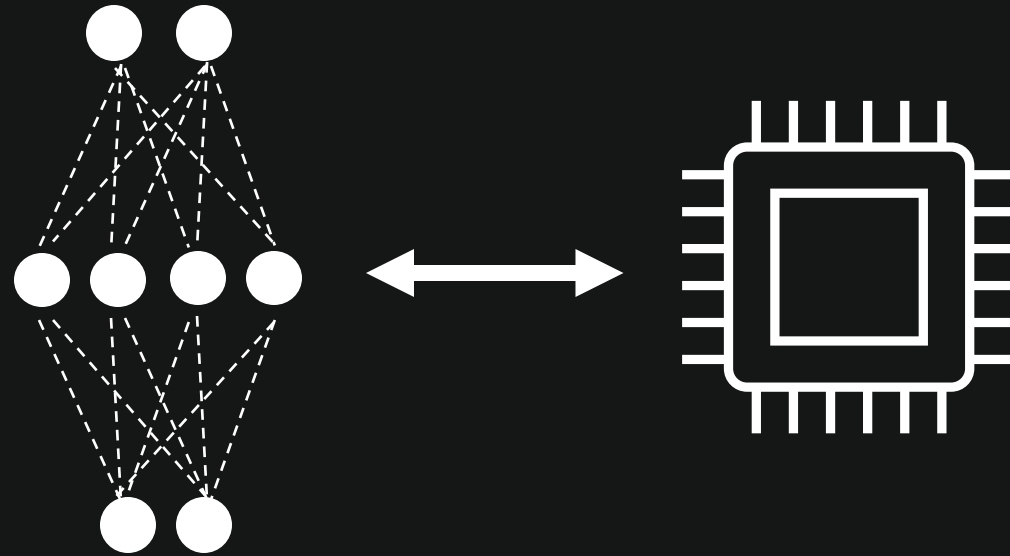
Microsoft urgently needs carbon-neutral electricity for its AI data centers. A decommissioned nuclear power plant in the USA is now to be reactivated.

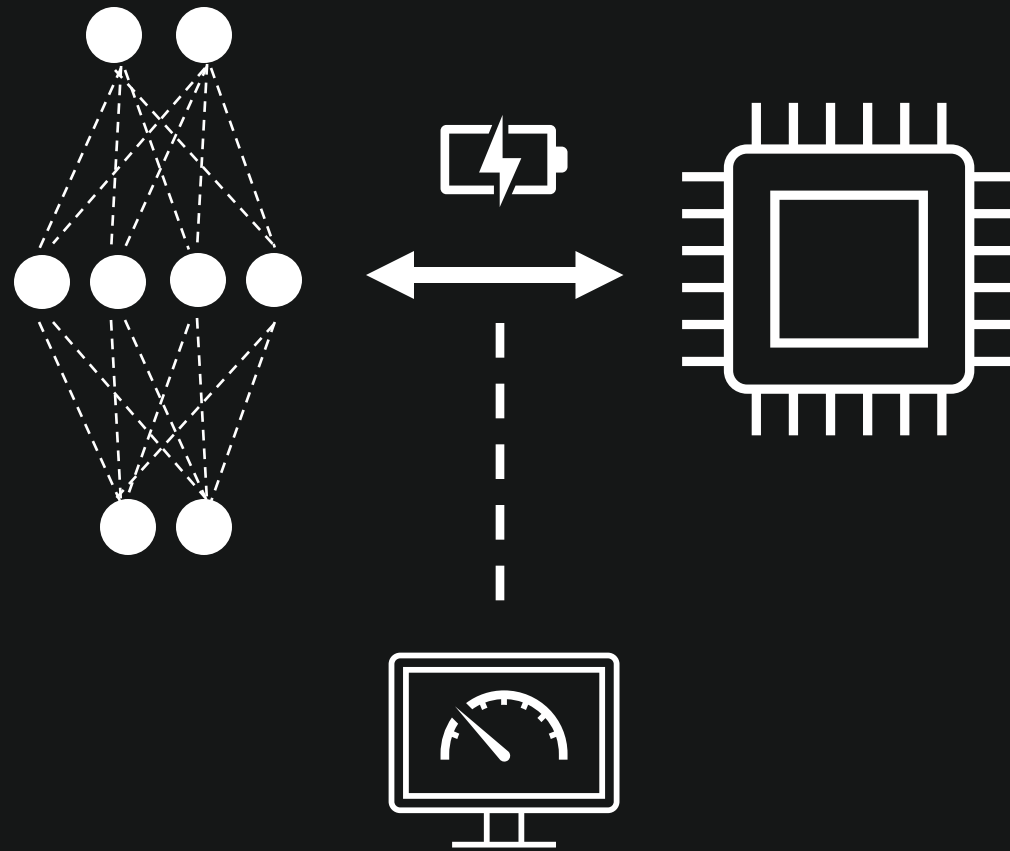


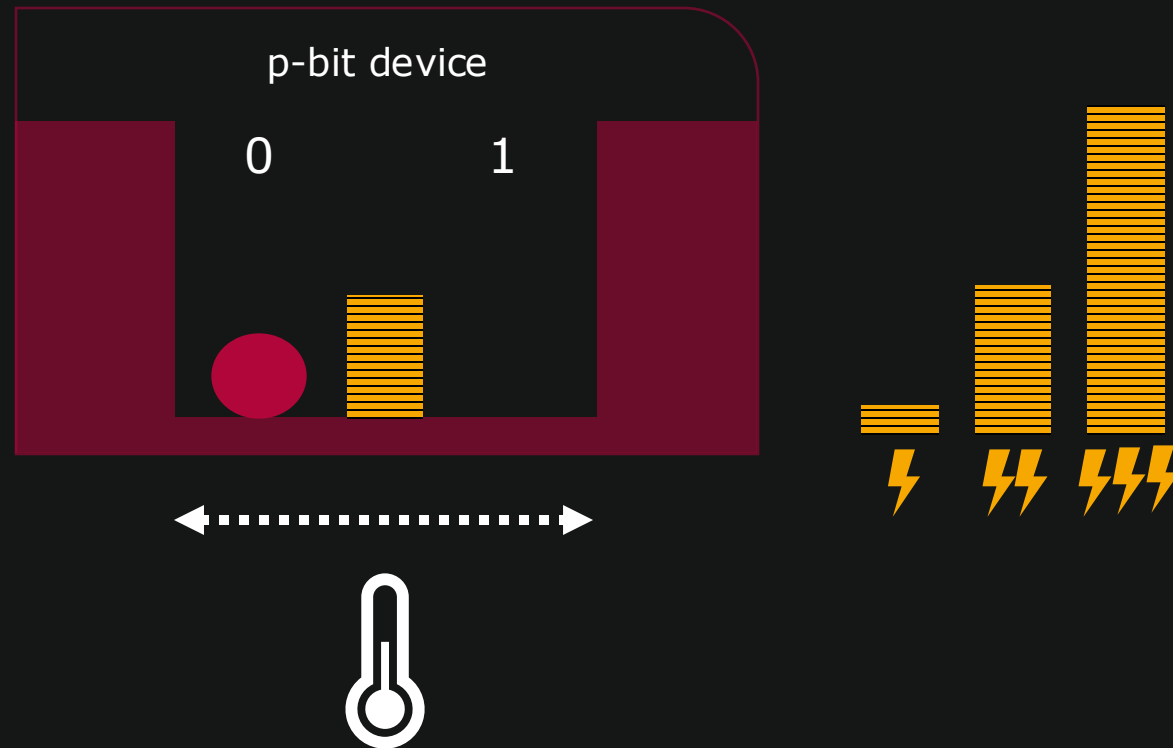
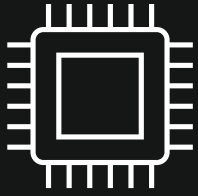
Forbes

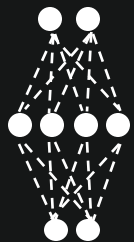
AI Writes Over 25% Of Code At Google—What Does The Future Look Like For Software Engineers?











Random Number Generation

Data Preparation

- **Data Shuffling:** Shuffling training data to improve the robustness of the model.
- **Data Splitting:** Randomly splitting data into training, validation, and test sets.
- **Data Augmentation:** Randomly altering data (e.g., flipping, cropping, rotating images).

Optimization and Training Algorithms

- **Stochastic Gradient Descent (SGD):** Using random mini-batches to train models, commonly used in neural networks.
- **Dropout Regularization:** Randomly "dropping out" neurons during training to prevent overfitting.
- **Bayesian Optimization:** Randomly sampling from prior distributions over hyperparameters to find the optimal configuration.

Probabilistic and Generative Models

- **Markov Chain Monte Carlo (MCMC):** Sampling from posterior distributions in Bayesian inference.
- **Variational Inference:** Using random samples for approximating posterior distributions, as in Variational Autoencoders (VAEs).
- **GANs (Generative Adversarial Networks):** Random noise generation in the generator network to create new, diverse samples.

Ensemble Methods

- **Bagging (e.g., Random Forests):** Randomly sampling data with replacement to create multiple models.
- **Random Subspace Methods:** Randomly selecting subsets of features to train models.
- **Boosting (e.g., AdaBoost):** Randomly selecting data to assign higher weights to misclassified samples.

Sampling-Based Methods

- **Bootstrap Sampling:** Generating samples with replacement from a dataset to estimate statistics or perform cross-validation.
- **Reservoir Sampling:** Random sampling from a potentially unbounded data stream.
- **Reinforcement Learning:** Generating random exploration actions or policy sampling in approaches like Q-learning.

Dimensionality Reduction

- **Stochastic Neighbor Embedding (SNE):** Random initialization and sampling to preserve pairwise similarities in dimensionality reduction tasks.
- **Principal Component Analysis (PCA):** Often uses random initialization for approximate PCA calculations in large datasets.

Initialization Techniques

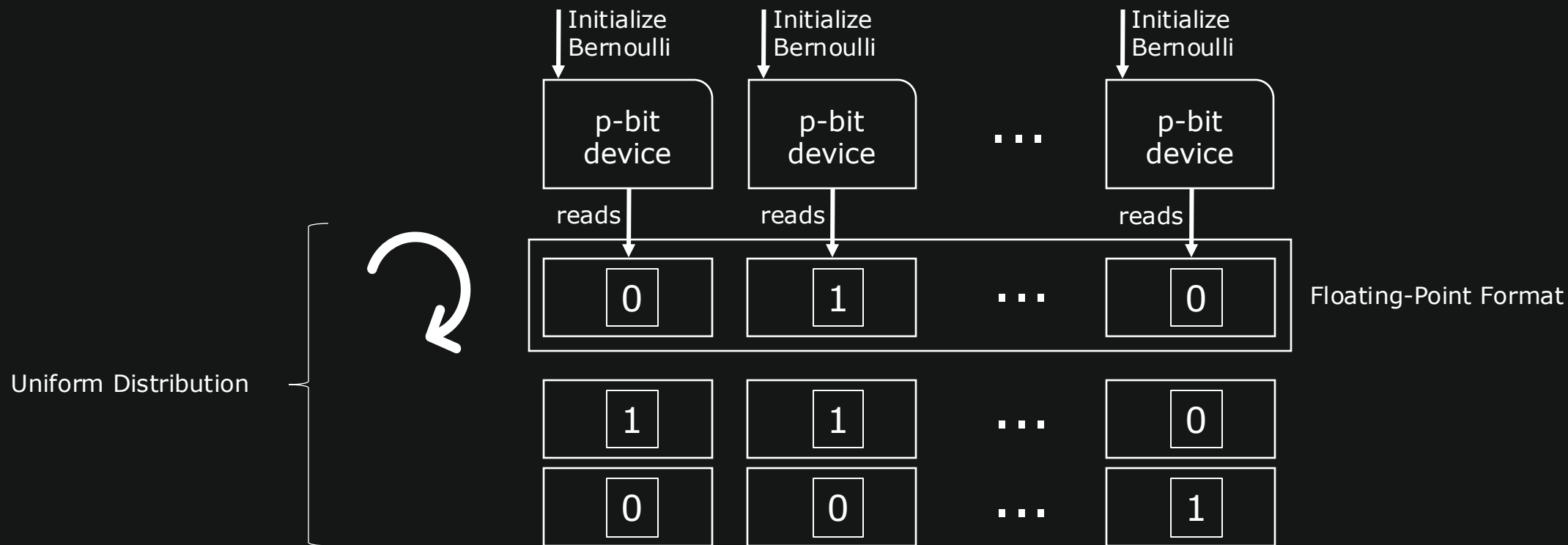
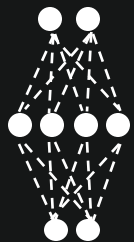
- **Weight Initialization for Neural Networks:** Random initialization (e.g., Xavier or He initialization) of weights to break symmetry.
- **Cluster Initialization in K-Means:** Randomly choosing initial cluster centroids or leveraging techniques like K-Means++ for a more informed start.

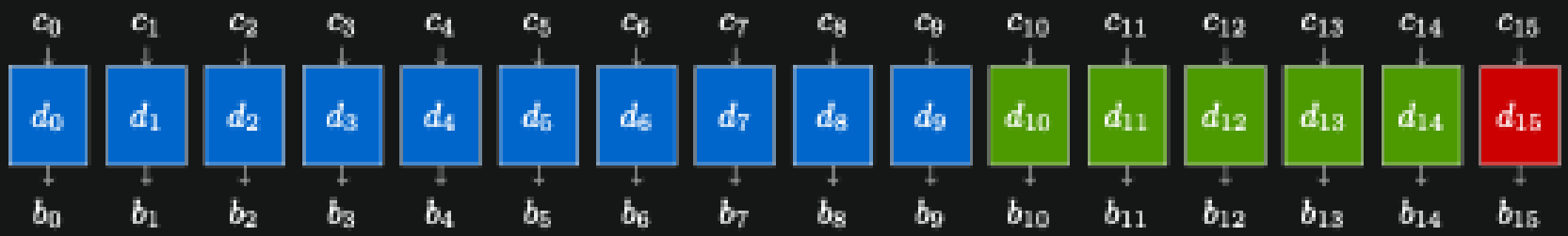
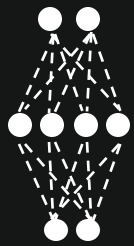
Reinforcement Learning Algorithms

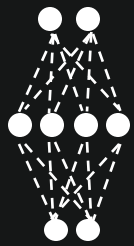
- **Policy Gradient Methods:** Using random samples from policies to compute gradients.
- **Q-Learning:** Random exploration in action selection with epsilon-greedy strategies.
- **Monte Carlo Tree Search (MCTS):** Random simulations to explore possible outcomes.

Testing and Evaluation

- **Cross-Validation:** Using random sampling for different validation splits.
- **A/B Testing:** Randomly assigning users to control and test groups.
- **Random Seed Setting:** Repeatedly generating consistent random states to ensure reproducibility across experiments.





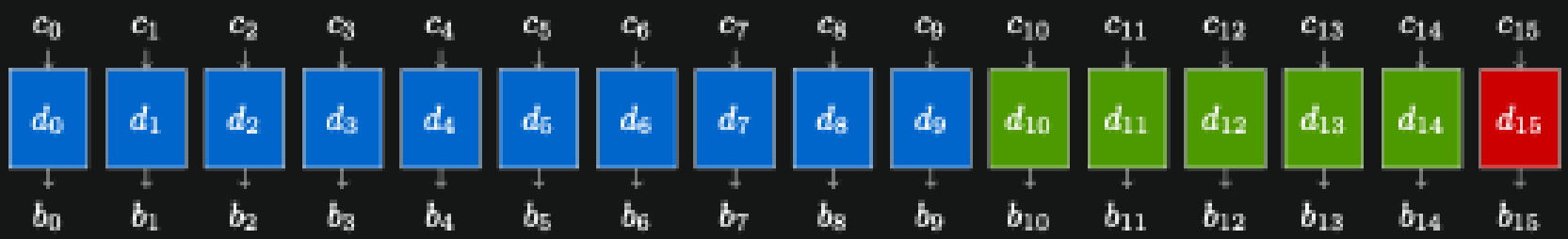


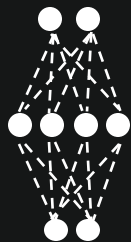
$$C = \{(b_i, p_i) \mid p_i \in [0, 1], b_i \in \{b_0, \dots, b_{15}\}, \text{ where} \quad (1)$$

$$p_i = \begin{cases} \frac{o_i - 0.5}{2^{(2^e)-1}} & \text{if } i \in \{10, \dots, 14\}, \\ 0.5 & \text{otherwise} \end{cases}, \text{ and} \quad (2)$$

$$o_i = 2^{2^{i-1}} + \sum_{k=1}^{c-1} 2^{2^{i-1}+k} \cdot \mathbf{1}_{\{i>1\}} + \sum_{k=1}^{z-1} 2^{2^{i-1}+2^i \cdot k} + \sum_{k=1}^{z-1} \sum_{g=1}^{c-1} 2^{2^{i-1}+2^i \cdot k+g} \cdot \mathbf{1}_{\{z>1\}}, \text{ and} \quad (3)$$

$$z = 2^{-i+e}, c = 2^{i-1}, e = 5. \quad (4)$$



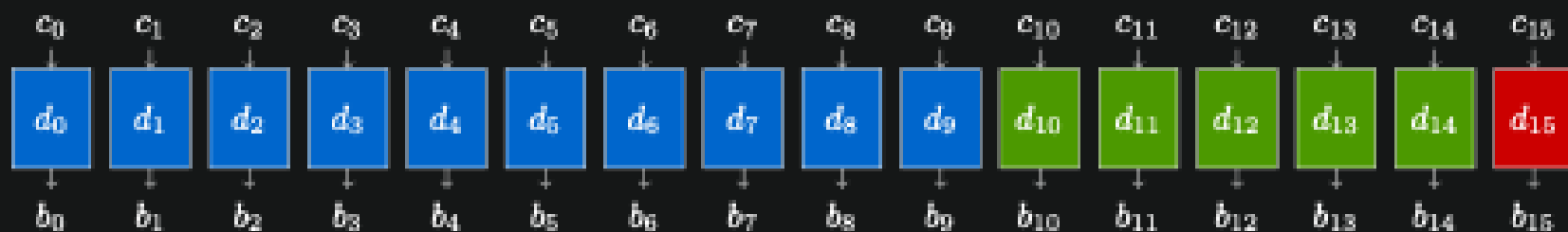


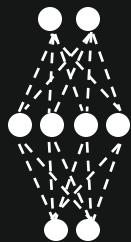
$$C = \{(b_i, p_i) \mid p_i \in [0, 1], b_i \in \{b_0, \dots, b_{15}\}, \text{ where} \quad (1)$$

$$p_i = \begin{cases} \frac{o_i - 4}{2^{(2^e)-1}} & \text{if } i \in \{10, \dots, 14\}, \text{ and} \\ 0.5 & \text{otherwise} \end{cases} \quad (2)$$

$$o_i = 2^{2^{i-1}} + \sum_{k=1}^{c-1} 2^{2^{i-1}+k} \cdot \mathbf{1}_{\{i>1\}} + \sum_{k=1}^{z-1} 2^{2^{i-1}+2^i \cdot k} + \sum_{k=1}^{z-1} \sum_{g=1}^{c-1} 2^{2^{i-1}+2^i \cdot k+g} \cdot \mathbf{1}_{\{z>1\}}, \text{ and} \quad (3)$$

$$z = 2^{-i+c}, c = 2^{i-1}, e = 5. \quad (4)$$



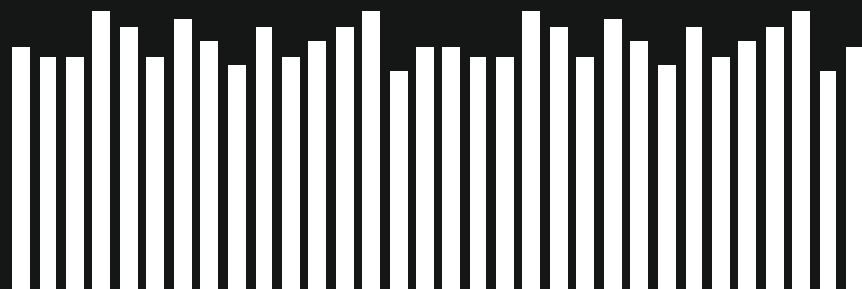
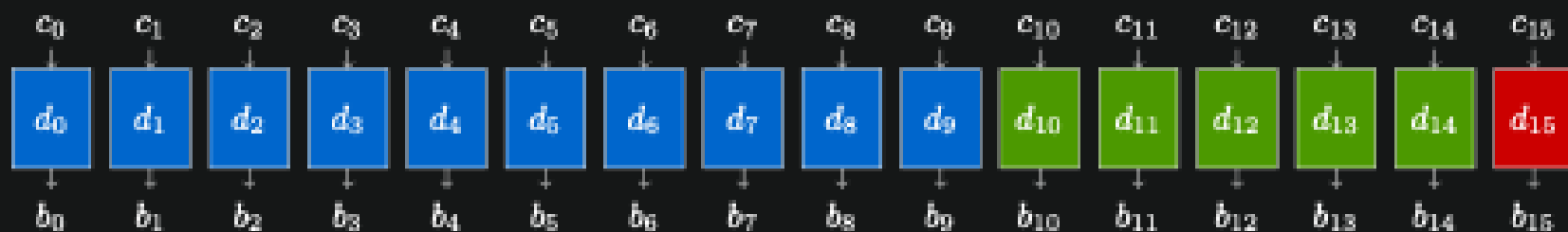


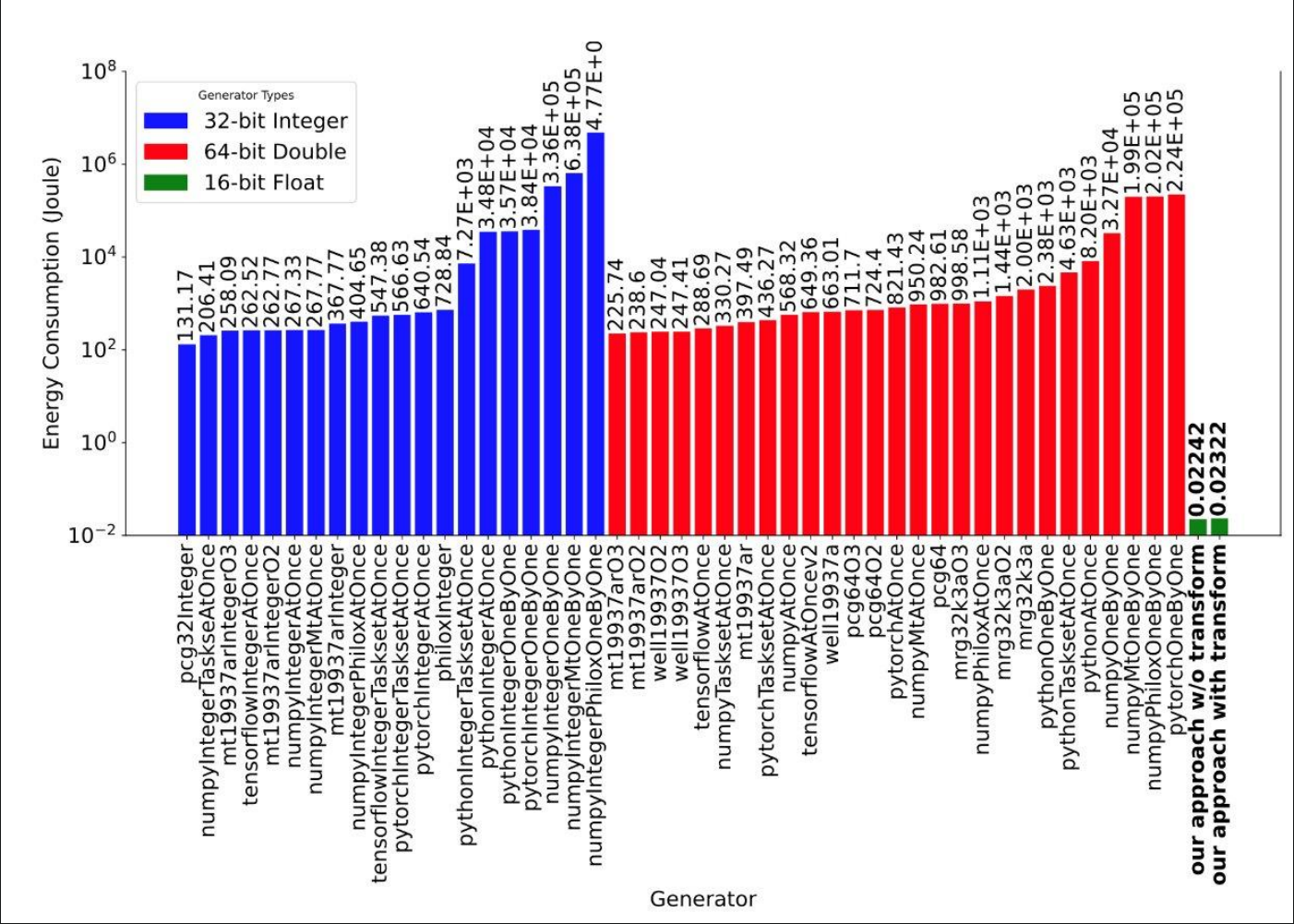
$$C = \{(b_i, p_i) \mid p_i \in [0, 1], b_i \in \{b_0, \dots, b_{15}\}, \text{ where} \quad (1)$$

$$p_i = \begin{cases} \frac{o_i - 0.5}{2^{(2^e)-1}} & \text{if } i \in \{10, \dots, 14\}, \text{ and} \\ 0.5 & \text{otherwise} \end{cases} \quad (2)$$

$$o_i = 2^{2^{i-1}} + \sum_{k=1}^{c-1} 2^{2^{i-1}+k} \cdot \mathbf{1}_{\{i>1\}} + \sum_{k=1}^{z-1} 2^{2^{i-1}+2^i \cdot k} + \sum_{k=1}^{z-1} \sum_{g=1}^{c-1} 2^{2^{i-1}+2^i \cdot k+g} \cdot \mathbf{1}_{\{z>1\}}, \text{ and} \quad (3)$$

$$z = 2^{-i+c}, c = 2^{i-1}, e = 5. \quad (4)$$





Alder, N., Kajale, S. N., Tunsiricharoengul, M., Sarkar, D., & Herbrich, R. Energy-Efficient Random Number Generation Using Stochastic Magnetic Tunnel Junctions. In *NeurIPS 2024 Workshop Machine Learning with new Compute Paradigms*.

Thank you.