Neuron-level *Pruning* Framework for *Bayesian Neural Networks*

x

Background

Pruning

- Pruning helps reduce model size and speeds up inference by finding redundant network parameters that can be removed post-training.
- Traditional neural network pruning usually targets individual weights using criteria like magnitude or loss sensitivity.
- **Neuron-level (structured) pruning** removes entire neurons, shrinking weight matrices.
- Structured pruning yields practical speedups on standard hardware, unlike unstructured (sparse) pruning.
- The *Lottery Ticket Hypothesis* supports pruning: small subnetworks ("winning tickets") can match the performance of full networks.

Bayesian Neural Networks (BNNs)

- BNNs assign **probability distributions to weights**, capturing model uncertainty.
- This enables BNNs to output **predictive distributions**, not just point estimates—crucial for safety-critical applications like healthcare.

$$p(y^*|x^*, \mathcal{D}) = \int p(y^*|x^*, w)\, p(w|\mathcal{D})\, dw. \qquad (1)$$

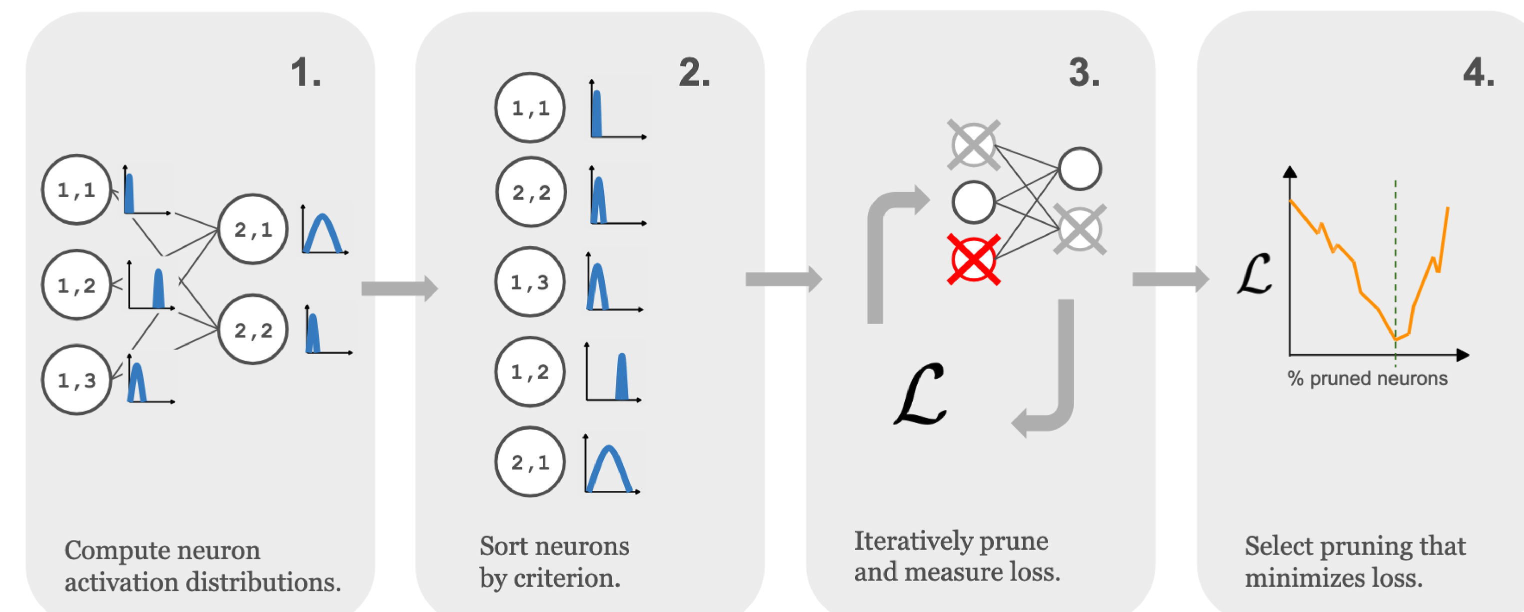
- **BNNs are computationally expensive** due to repeated forward passes—pruning helps make them practical.
- This work introduces a **neuron-level pruning framework for BNNs** that:
  - Preserves predictive uncertainty,
  - Requires no hyperparameter tuning,
  - Achieves over 80% neuron pruning on benchmark datasets.

Problem Formulation

- When pruning a Bayesian Neural Network (BNN), we aim to balance two competing objectives:
  1. **Prune as many neurons as possible**, reducing model size and inference cost.
  2. **Preserve the shape of the posterior predictive distribution** $p(y^* \mid x^*, \mathcal{D})$, which captures model uncertainty.
- To formalize this trade-off, we define a cost function over a pruning mask $\boldsymbol{\alpha} \in \{0,1\}^{|w|}$, where $\alpha_i = 0$ means neuron i is pruned.

$$\alpha^* = \arg \min_{\alpha \in \{0,1\}^{|w|}} \underbrace{\frac{\mathbb{W}_2(p_{\bar{1}_D}, p_{\alpha_D})}{\mathbb{W}_2(p_{\bar{1}_D}, p_{\bar{0}_D})}}_{\text{Preservation of predictive distribution}} + \lambda \cdot \underbrace{\frac{1}{|w|}\sum_{i=1}^{|w|} \alpha_i}_{\text{Sparsity penalty}}$$

- $p_{\bar{1}_D}$ and p_{α_D} are the predictive distributions of the original and pruned networks over dataset $\mathcal{D}$. $p_{\bar{0}_D}$ is posterior of a fully pruned network.
- $\mathbb{W}_2(\cdot,\cdot)$ is the Wasserstein-2 metric, measuring how pruning shifts the predictive output.
- λ controls the trade-off: larger λ favors compression, smaller λ favors distribution fidelity.



Methodology

Challenge: The original pruning objective $\mathbb{W}_2(p_{\bar{1}_D}, p_{\alpha_D})$ is intractable to optimize directly due to the intractability of relating changes in neurons to changes in output distributions. **Solution: Neuron-Level Approximation:**

- Develop tractable criterion that measures how much removing a neuron i in layer k affects the predictive distribution.
- Key insight: Output distribution changes directly depend on individual neuron activation distributions
- Prioritize pruning neurons whose removal would least impact the network's predictive capabilities.
- Pruned neurons output δ_0 - we can measure how much it would change if we were to prune it by **criterion** $\mathbb{W}_2(\tilde{p}^{\alpha}_{i,k}, \delta_0) = \mathbb{V}_{\tilde{z}\sim\tilde{p}^{\alpha}_{i,k}}[\tilde{z}] + \mathbb{E}_{\tilde{z}\sim\tilde{p}^{\alpha}_{i,k}}[\tilde{z}]^2$
- Aggregating activations across diverse input samples provides a representative measure of the neuron's contribution
- We then use this criterion in a 4 step iterative pruning procedure.

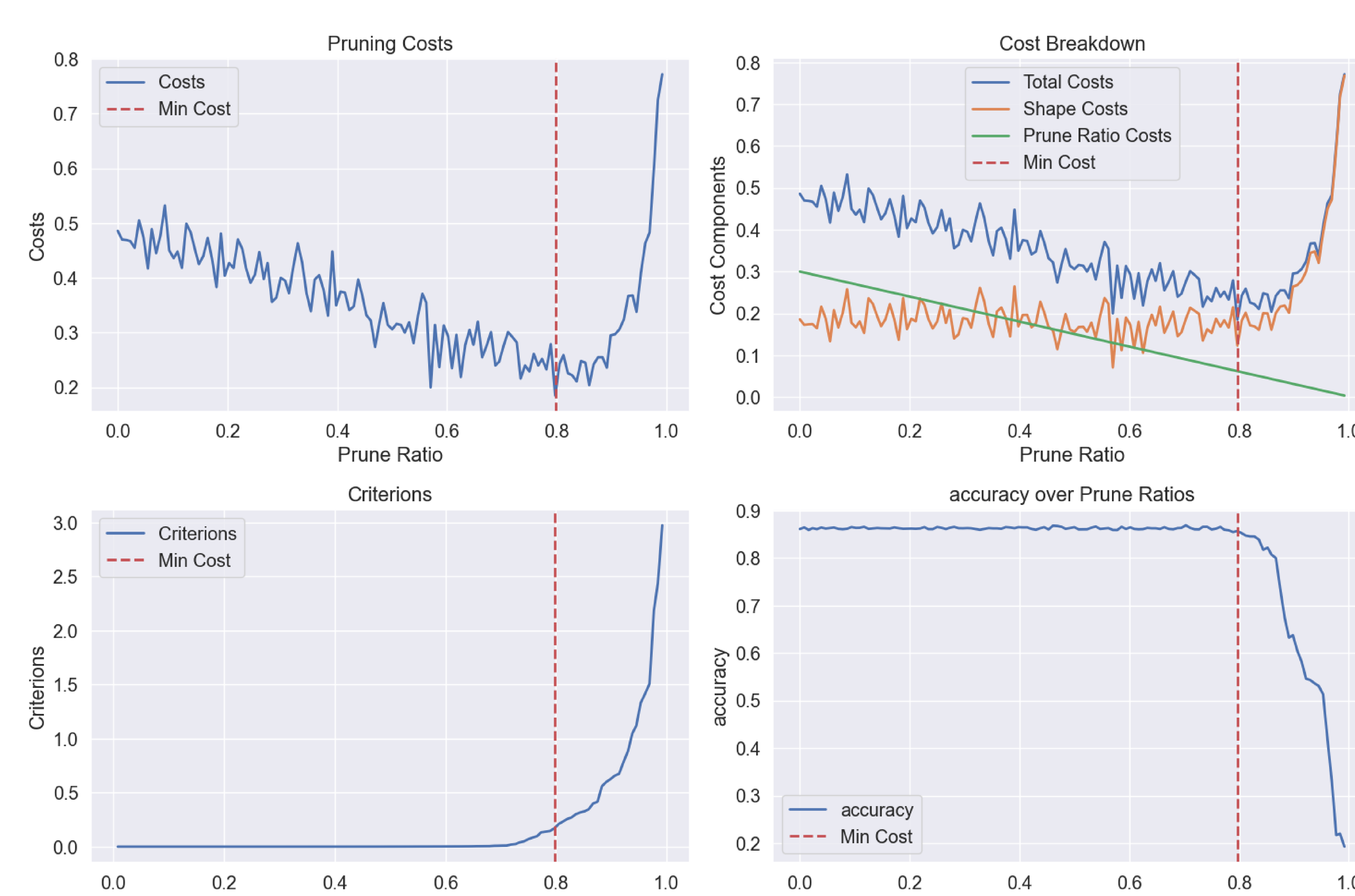


Figure 1. Fashion MNIST neural network pruning experiments. **Top left**: Pruning costs versus prune ratio, showing total cost increasing sharply after 80% optimal pruning. **Top right**: Decomposition into shape costs (distribution change) and prune ratio costs (sparsity), illustrating the trade-off between model compression and performance. **Bottom left**: Heuristic criterion value for the last pruned neuron, demonstrating increased neuron importance at higher compression. **Bottom right**: Test accuracy remains stable until critical pruning threshold at 80%, followed by rapid performance degradation.

Experiments

We Can Prune More as the Network Gets Larger/Overparameterized

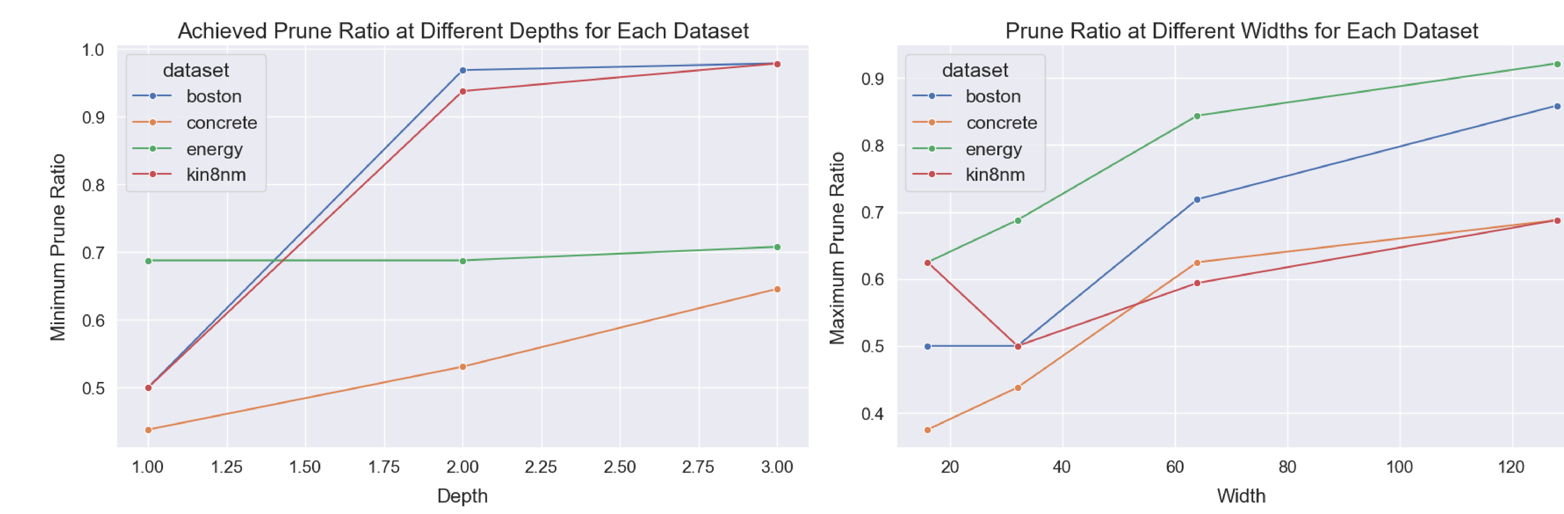


Figure 2. Maximum prune ratio achieved for increasing depth/width. For the depth graph, a fixed width of 32 neurons was used. For the width graph, one layer was used.

Pruning Is Steerable with λ Parameter

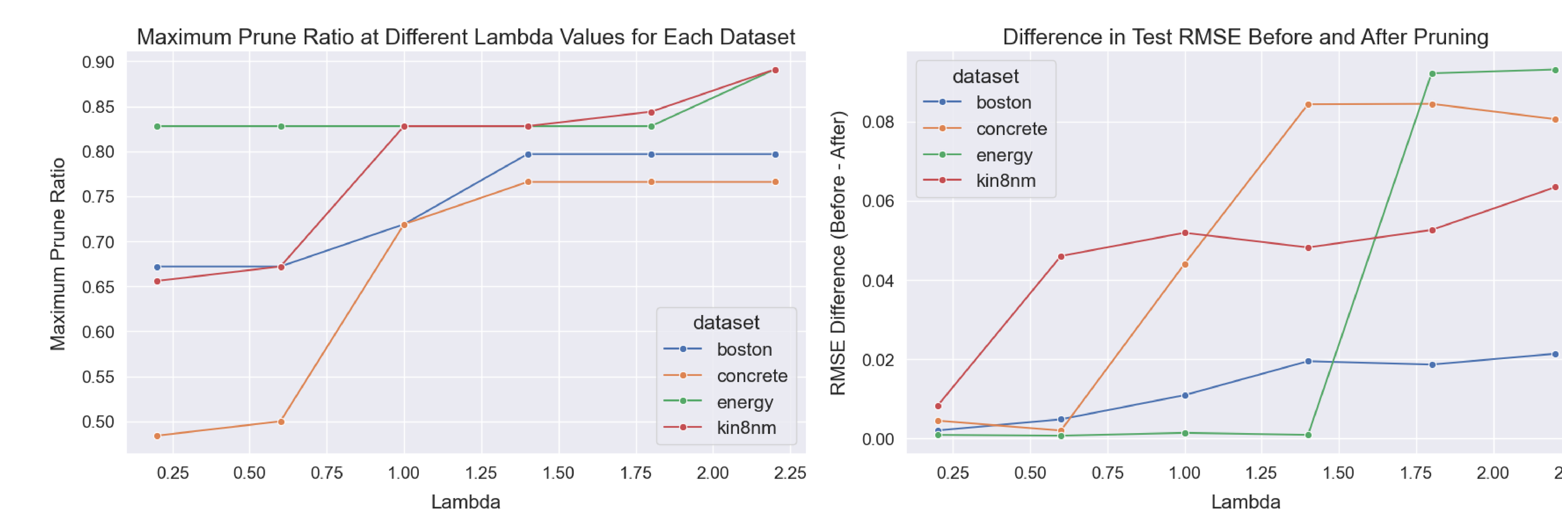


Figure 3. On the left, prune ratio achieved with different λ values. On the right, performance drops in RMSE before and after pruning for different λ values

Validates Lottery Ticket Hypothesis in Bayesian Setting

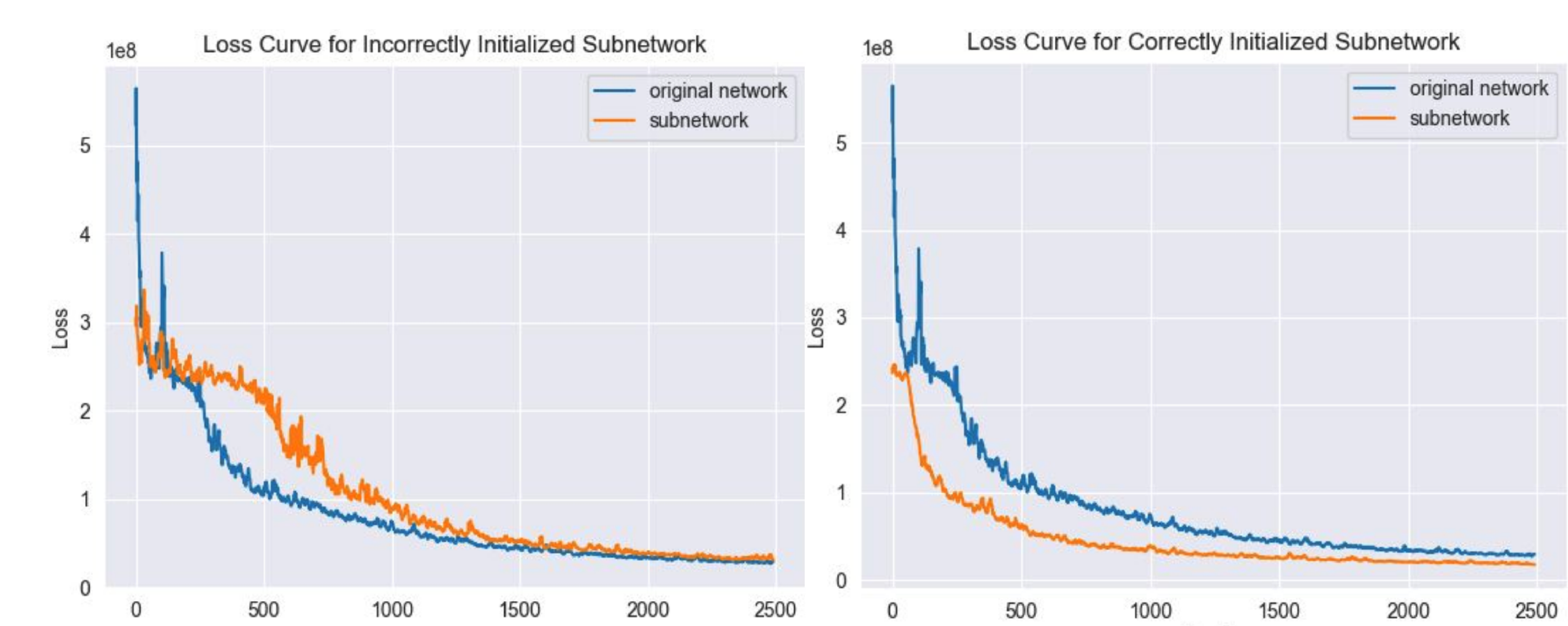


Figure 4. Comparison of a subnetwork trained from scratch (**Left**) with weight initialization from the original network vs different random initialization (**Right**).

Conclusion & Future Work

- **General Framework:** Formalized Bayesian pruning objective balancing predictive distribution preservation with network complexity reduction
- **Hardware Advantages:** Neuron-level pruning enables speedups and memory reduction on current hardware, unlike sparse methods
- **BNN Scalability:** Successfully pruned majority of neurons across classification/regression while maintaining performance
- **Future Directions:** Why the choice of optimizer influences performance? Could our pruning criterion speed up training, not only inference?

Author: **Vaclav Kubon** | Supervisors: **Steven Adams, Dr. Luca Laurenti, Dr. Avishek Anand**

TUDelft