

Efficient Training of CNN Ensembles via Feature-Prioritized Boosting

Biyi Fang¹, Truong Vo¹,
Jean Utke², Diego Klabjan¹
¹Northwestern University, ²Allstate



Northwestern
University



Introduction

Deep Learning Dilemma:

- Deep CNNs achieve state-of-the-art results but are computationally expensive, often requiring millions of parameters.
- Finding the optimal architecture is difficult; automated methods like Neural Architecture Search (NAS) can take **thousands of GPU hours**.

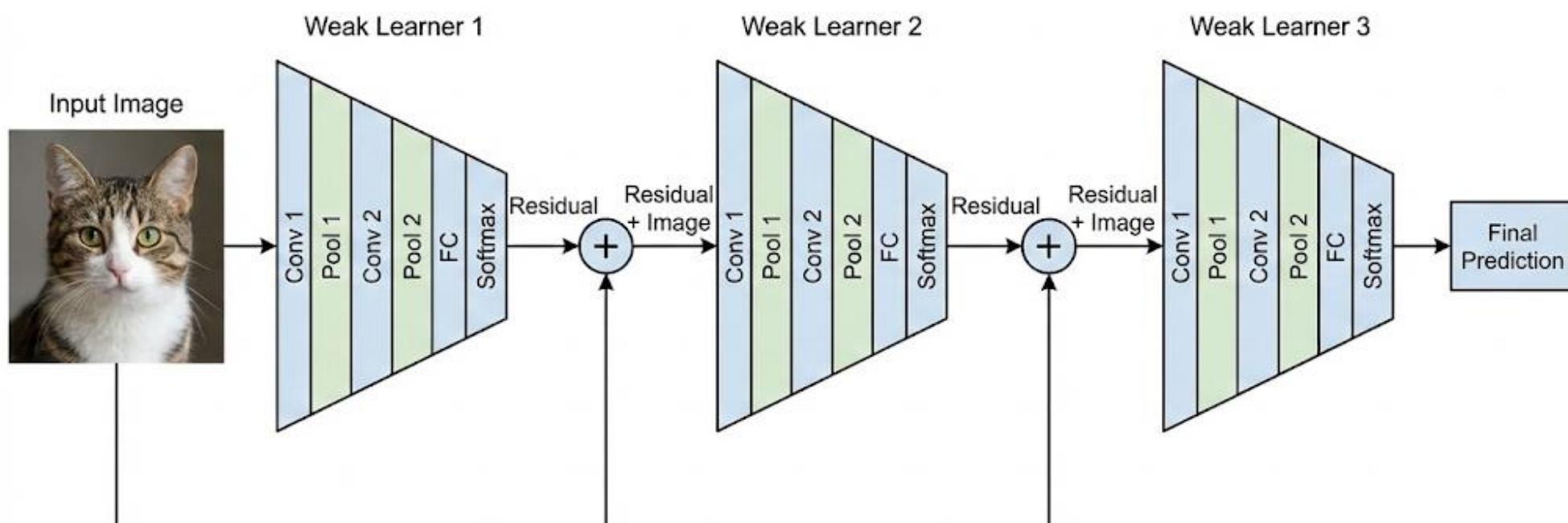
Boosting Bottleneck:

- Ensemble methods (Boosting) theoretically improve accuracy and simplify architecture design.
- However, standard **BoostCNN** is slow and memory-intensive because every weak learner must process the **full-size image**, leading to redundant computation.

Our Solution: Subgrid BoostCNN

- We introduce a novel framework that integrates **Dynamic Feature Selection** (Subgrids) with Boosting.
- **Key Innovation:** Instead of training on the whole image, each weak learner focuses only on "informative" pixels (high gradients/residuals), significantly reducing training complexity while maintaining high accuracy.

Background



Core Idea: Sequentially train "Weak Learners" (CNNs) to correct the errors of the previous ensemble.

Mechanism:

- Input image goes into Weak Learner 1.
- Calculate Residuals (Errors).
- Pass residuals/weights to Weak Learner 2.
- **Final Output:** Weighted sum of all learners: $f(x) = \sum \alpha_t g_t(x)$.

Limitation:

Standard BoostCNN trains full models each round leading to high cost

Algorithms

Subgrid Selection Strategy:

- **Objective:** Reduce computational burden by training on a subset of informative pixels rather than the full image⁹.
- **Importance Index ($I_{j,k}$):** We calculate the importance of each pixel based on the gradient of the loss function. High gradient = high importance (needs correction)

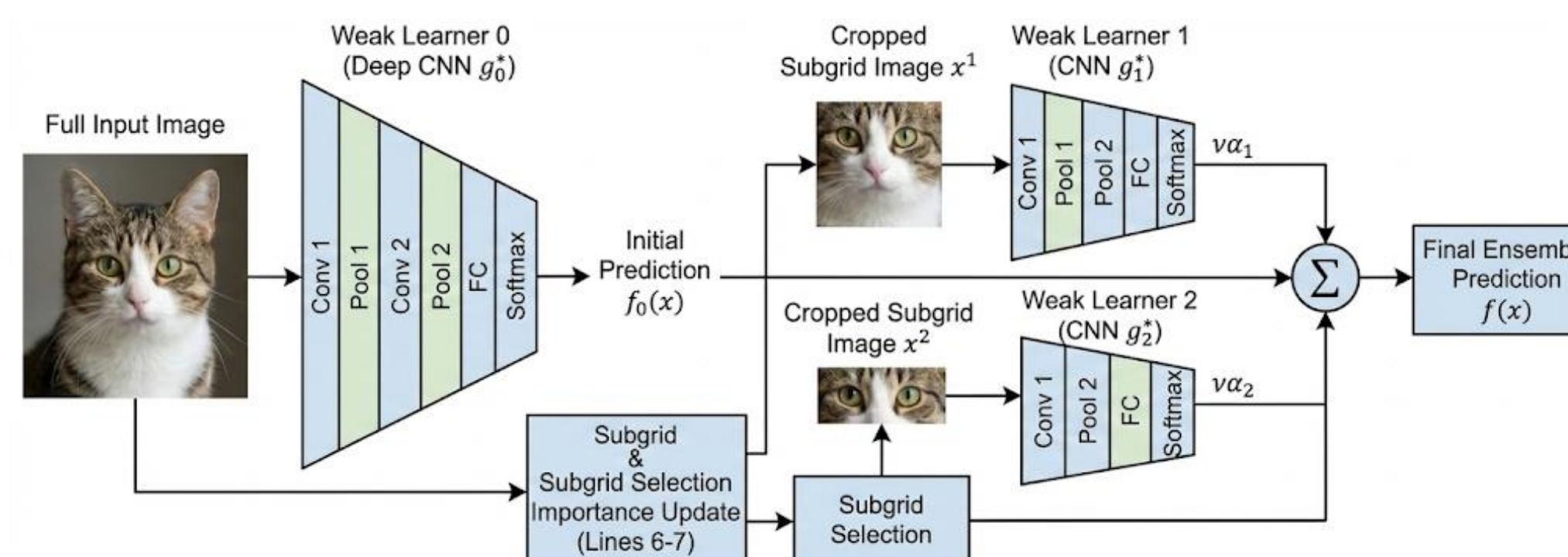
$$I_{j,k} = \frac{1}{|\mathcal{D}|} \sum_{(x_i, z_i) \in \mathcal{D}} \sum_{c \in C} \left| \frac{\partial \mathcal{L}}{\partial x_i^{j,k,c}} \right|$$

- **Dynamic Selection:** At each iteration t , we drop the least important rows/columns (approx 10%) and train the next learner only on the selected **Subgrid**.

Architecture Reuse & Optimization

- **Efficient Training:** To avoid re-optimizing the full CNN every time:
 - **Reuse:** We reuse the convolutional layers (feature extractor) from the previous learner.
 - **Fixed Head:** We pair them with a fixed classifier head to compute pixel importance quickly.
- **Loss Function:** We treat the boosting step as a regression problem. The weak learner $g(x)$ minimizes the Least Squares error against the boosting weights $w(x)$:

$$\mathcal{L}(w, g) = \sum \|g(x_i) - w(x_i)\|^2$$



Subgrid BoostCNN Overview:

Initialize: $f(x) = 0$.

Loop ($t=1$ to N_b):

- Compute Boosting Weights $w(x)$ based on current error.
- **Subgrid Step:** Calculate Importance Index $I_{j,k}$ and select top pixels.
- **Train:** Train weak CNN g_t on the **Subgrid** to match weights.
- **Update:** Add to ensemble. $f(x) \leftarrow f(x) + \nu \alpha_t g_t$.

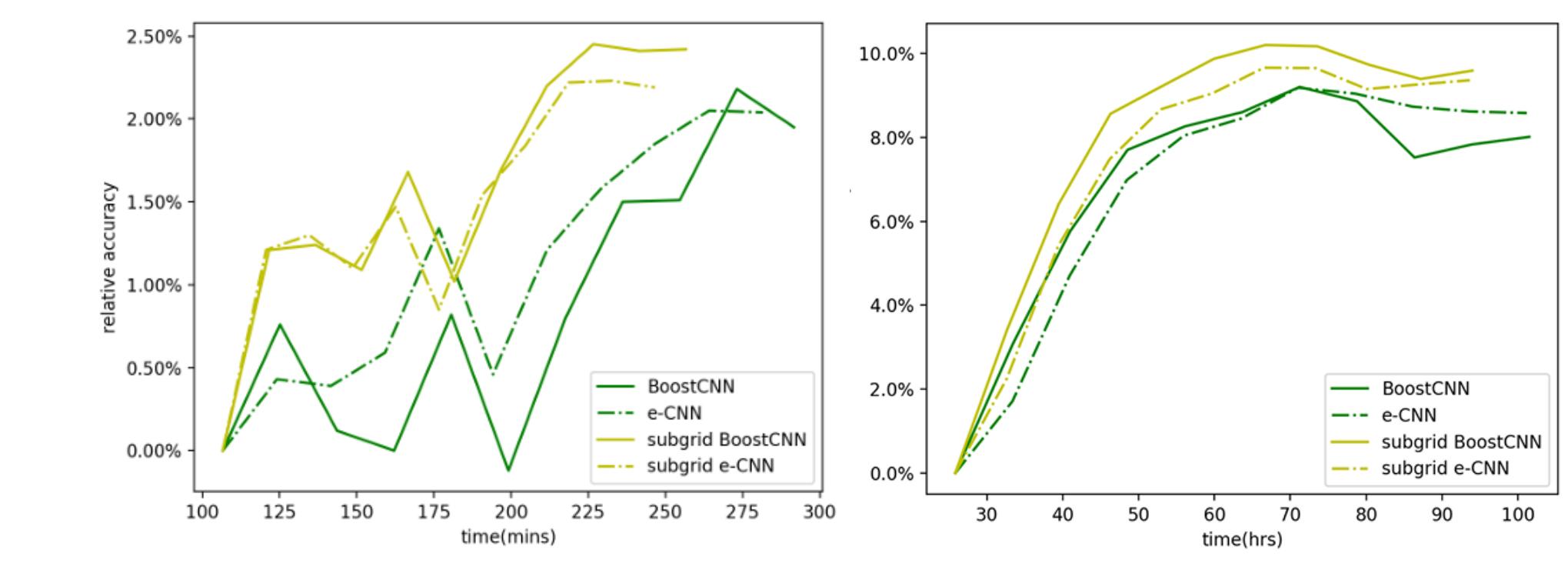
Experimental Study

Experimental Setup

- **Datasets:** CIFAR-10, SVHN, ImageNetSub.
- **Models:** ResNet-18, ResNet-50, ResNet-101 used as weak learners.
- **Baselines:** Compared against Standard BoostCNN and e-CNN (Ensemble of CNNs without boosting).

Main Results

- **Accuracy vs. Time:** Subgrid BoostCNN consistently achieves higher accuracy in less time compared to standard BoostCNN and e-CNN.
 - *Example:* On CIFAR-10, Subgrid BoostCNN outperforms all baselines for the same training duration.



- **Robustness:** Subgrid BoostCNN shows significantly lower standard deviation across different random seeds compared to e-CNN, indicating high stability.

	subgrid BoostCNN	subgrid e-CNN
CIFAR-10	0.478	2.519
SVHN	0.385	0.891
ImageNetSub	2.489	7.915

Standard deviation times 10^3 of the accuracy results by different seeds

Conclusion

- **Efficiency:** Reduces training complexity by focusing only on "important" image regions.
- **Performance:** Outperforms single deep CNNs (e.g., ResNet-101) using an ensemble of shallower learners (ResNet-50).
- **Impact:** A scalable, generalizable solution for high-accuracy image classification with reduced resource costs.

