In-Model Merging for Enhancing the Robustness of Medical Imaging Classification Models

Hu Wang[1], Ibrahim Almakky[1], Congbo Ma[2], Numan Saeed[1], Mohammad Yaqub[1]
[1]*Mohamed Bin Zayed University of Artificial Intelligence, Abu Dhabi, UAE*
[2]*New York University Abu Dhabi, Abu Dhabi, UAE*

BMVC 2025
The 36th British Machine Vision Conference 24th - 27th November 2025, Sheffield, UK

Abstract

Motivation: Model merging effectively integrates multiple models to improve performance, while being more efficient than ensemble learning as it adds *no inference overhead*. Yet, it remains unexplored whether merging can occur **within a single model** to improve **robustness**, a crucial aspect in **medical imaging**.

Our contribution: we are the first to propose **in-model merging (InMerge)**, a novel approach that enhances the model's robustness by selectively merging similar convolutional kernels in the deep layers of a single CNN during the training process for classification. We analytically reveal key characteristics that influence how in-model merging should be performed, providing **guidance for future research**. We validate InMerge on **four major medical image datasets** and across different CNN architectures. The **InMerge-trained models significantly outperform** traditionally trained models in both accuracy and robustness.

Code availability: `https://github.com/billhhh/In-Model-Merging`

Methodology

Kernel Similarity Computation

The similarity between two convolutional kernels is computed via cosine similarity, which measures the angle between their vectorized forms. For kernels $\mathbf{K}_i$ and $\mathbf{K}_j$ in the same layer, we first flatten them into 1D vectors:

$$\mathbf{k}_i = \text{vec}(\mathbf{K}_i), \quad \mathbf{k}_j = \text{vec}(\mathbf{K}_j), \tag{1}$$

The cosine similarity is then computed as:

$$\text{sim}(\mathbf{k}_i, \mathbf{k}_j) = \frac{\mathbf{k}_i^T \mathbf{k}_j}{\|\mathbf{k}_i\| \|\mathbf{k}_j\|}, \tag{2}$$

In-Model Kernel Merging Strategy

To preserve shallow features, merging is applied only to deeper layers (beyond the first L_s). In these layers, similar kernels are merged during training. Let $\mathbf{W} = \mathbf{K}_1, \mathbf{K}_2, \ldots, \mathbf{K}_n$ denote the kernels in a layer. For each $\mathbf{K}_i$, a different kernel $\mathbf{K}_j$ is randomly chosen. If their similarity exceeds a threshold τ, they are merged with probability p:

$$\mathbf{K}_i \leftarrow \alpha \mathbf{K}_i + (1-\alpha)\mathbf{K}_j, \quad s.t.\ \text{sim}(\mathbf{k}_i, \mathbf{k}_j) > \tau, \tag{3}$$

where $\tau \in [-1, 1]$ is the similarity threshold. This reduces redundancy while maintaining network expressiveness.

Layer Selection and Probabilistic Merging

To maintain the diversity of low-level feature representations, the first L_s shallow layers are excluded from kernel merging. Additionally, merging is performed in a probabilistic manner to avoid excessive regularization and unintended degradation of feature maps. Specifically, each kernel undergoes merging with probability p, ensuring stochasticity in the process:

$$P(\mathbf{K}_i \leftarrow \alpha \mathbf{K}_i + (1-\alpha)\mathbf{K}_j) = p. \tag{4}$$

This probabilistic merging mechanism prevents the network from over-collapsing to a limited set of feature extractors, giving the effect of regularization, thereby retaining sufficient expressiveness while enhancing generalization.

Scan Me to Get the Code

Please scan me to access our publicly available code!

How In-Model Merging Works

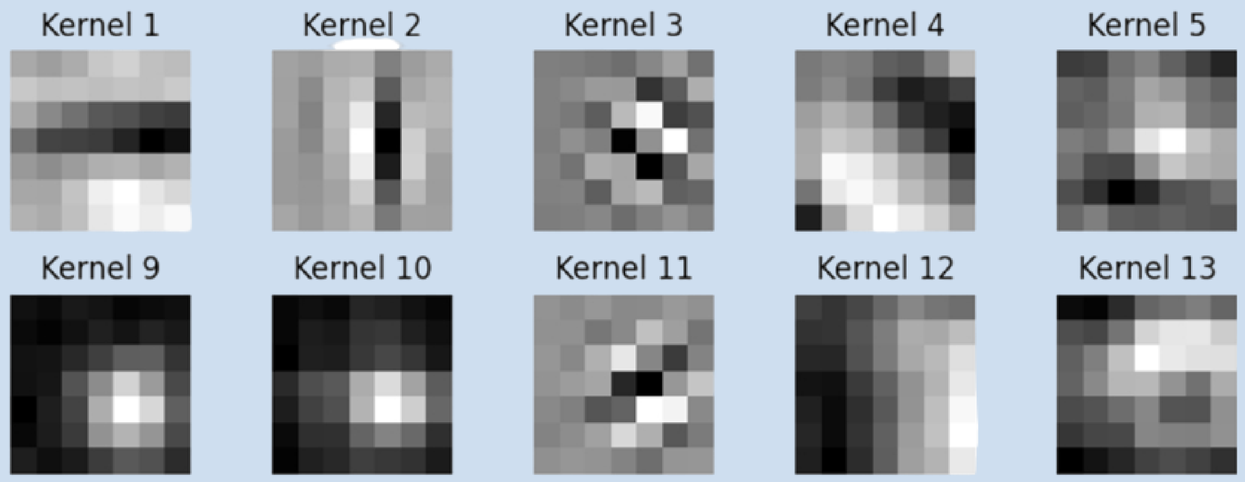

Figure 1: An example showing the similarity differences between patterns stored in kernels in a well-trained ResNet34 layer. The more similar pair of kernels would incur larger similarity. Kernel 9 and Kernel 10 are similar, the similarity $\text{sim}(\mathbf{k}_9, \mathbf{k}_{10}) = 0.9372$; Kernel 3 and Kernel 11 nearly have orthogonal textures, so $\text{sim}(\mathbf{k}_3, \mathbf{k}_{11}) = -0.0234$; Kernel 4 and Kernel 12 have very different textures and colors, $\text{sim}(\mathbf{k}_4, \mathbf{k}_{12}) = -0.5049$; Kernel 5 and Kernel 13 are somewhat similar, $\text{sim}(\mathbf{k}_5, \mathbf{k}_{13}) = 0.4562$.

Model Performance on ChestXRay Data

Models	Atel	Card	Effu	Infi	Mass	Nodu	Pneu1	Pneu2	Cons	Edem	Emph	Fibr	P_T	Hern	Ave
U-DCNN [23]	0.700	0.810	0.759	0.661	0.693	0.669	0.658	0.799	0.703	0.805	0.833	0.786	0.684	0.872	0.745
LSTM [26]	0.733	0.856	0.806	0.673	0.718	0.777	0.684	0.805	0.711	0.806	0.842	0.743	0.724	0.775	0.761
AGCL [20]	0.756	0.887	0.819	0.689	0.814	0.755	0.729	0.850	0.728	0.848	0.906	0.818	0.765	0.875	0.803
CheXNet [17]	0.769	0.885	0.825	0.694	0.824	0.759	0.715	0.852	0.745	0.842	0.906	0.821	0.766	0.901	0.807
DNet [11]	0.767	0.883	0.828	0.709	0.821	0.758	0.731	0.846	0.745	0.835	0.895	0.818	0.761	0.896	0.807
CRAL [10]	0.781	0.880	0.829	0.702	0.834	0.773	0.729	0.857	0.754	0.850	0.908	0.830	0.778	0.917	0.816
CAN [15]	0.777	0.894	0.829	0.696	0.838	0.771	0.722	0.862	0.750	0.846	0.908	0.827	0.779	**0.934**	0.817
DualCXN [3]	0.784	0.888	0.831	0.705	0.838	0.796	0.727	0.876	0.746	0.852	0.942	**0.837**	0.796	0.912	0.823
LLAGnet [4]	0.783	0.885	0.834	0.703	0.841	0.790	0.729	0.877	0.754	0.851	0.939	0.832	**0.798**	0.916	0.824
CheXGCN [5]	0.786	0.893	0.832	0.699	0.840	**0.800**	0.739	0.876	0.751	0.850	**0.944**	0.834	0.795	0.929	0.826
InMerge (ours)	**0.805**	**0.895**	**0.881**	**0.710**	**0.842**	0.750	**0.770**	**0.878**	**0.801**	**0.888**	0.915	0.818	0.773	0.903	**0.830**

Model Performance on MedMnist Data

Dataset	Cls#	Samp#	Train/Val/Test	Baseline	InMerge (ours)
PathMNIST	9	107,180	89,996/10,004/7,180	0.898±0.010	**0.924±0.002**
DermaMNIST	7	10,015	7,007/1,003/2,005	0.745±0.009	**0.760±0.005**
BloodMNIST	8	17,092	11,959/1,712/3,421	0.941±0.005	**0.948±0.004**

Table 2: Performance comparison of Baseline and InMerge across different MedMNIST datasets in accuracy.

Layer and Merging Weights Sensitivity

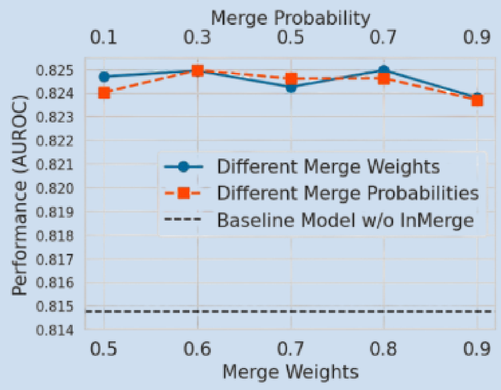

Figure 2: Merging with different merge weights and merge probabilities

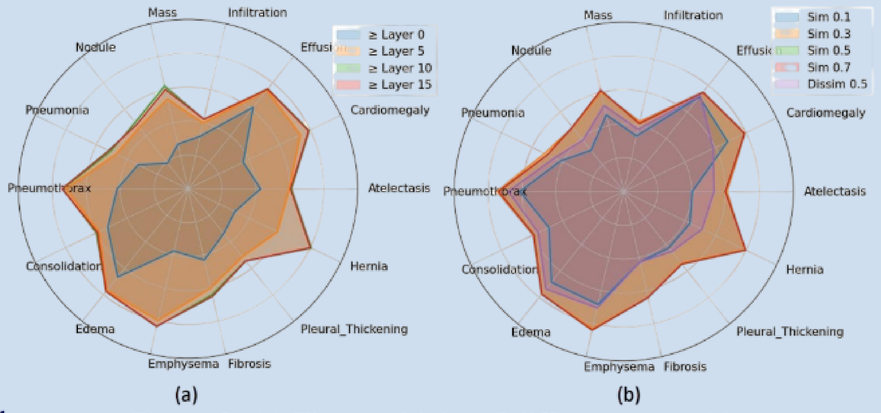

Figure 3: (a) The sensitivity of different merged layers; (b) Merging with different similarities.