

OUI Need to Talk About Weight Decay: A New Perspective on Overfitting Detection

Alberto Fernández-Hernández*
Universitat Politècnica de València
Valencia, Spain
a.fernandez@upv.es

Jose I. Mestre
Universitat Jaume I
Castelló de la Plana, Spain
jmiravet@uji.es

Manuel F. Dolz
Universitat Jaume I
Castelló de la Plana, Spain
dolzm@uji.es

Jose Duato
Qsimov Quantum Computing S.L. Universitat Politècnica de València
Madrid, Spain
jduato@qsimov.com

Enrique S. Quintana-Orti
Universitat Politècnica de València
Valencia, Spain
quintana@disca.upv.es

Abstract—We introduce the **Overfitting-Underfitting Indicator (OUI)**, a novel tool for monitoring the training dynamics of Deep Neural Networks (DNNs) and identifying optimal regularization hyperparameters. Specifically, we validate that OUI can effectively guide the selection of the Weight Decay (WD) hyperparameter by indicating whether a model is overfitting or underfitting during training without requiring validation data. Through experiments on DenseNet-BC-100 with CIFAR-100, EfficientNet-B0 with TinyImageNet and ResNet-34 with ImageNet-1K, we show that maintaining OUI within a prescribed interval correlates strongly with improved generalization and validation scores. Notably, OUI converges significantly faster than traditional metrics such as loss or accuracy, enabling practitioners to identify optimal WD (hyperparameter) values within the early stages of training. By leveraging OUI as a reliable indicator, we can determine early in training whether the chosen WD value leads the model to underfit the training data, overfit, or strike a well-balanced trade-off that maximizes validation scores. This enables more precise WD tuning for optimal performance on the tested datasets and DNNs. All code for reproducing these experiments is available at <https://github.com/AlbertoFdezHdez/OUI>.

Index Terms—Overfitting-Underfitting Indicator, regularization, generalization, training dynamics.

I. INTRODUCTION

The challenge of overfitting in training DNNs has become increasingly pronounced, fueled by the overparameterization characteristic of many state-of-the-art architectures. Although DNNs with strong *expressive power* [1]–[3]—*i.e.*, the ability to approximate arbitrarily complex functions with increasing precision—hold the promise of exceptional performance in terms of validation scores, they often exploit this by memorizing specific details of the training set that are not related

This research was funded by the projects PID2023-146569NB-C21 and PID2023-146569NB-C22 supported by MICIU/AEI/10.13039/501100011033 and ERDF/UE. Alberto Fernández-Hernández was supported by the predoctoral grant PREP2023-001826 supported by MICIU/AEI/10.13039/501100011033 and ESF+. Jose I. Mestre was supported by the predoctoral grant ACIF/2021/281 of the *Generalitat Valenciana*. Manuel F. Dolz was supported by the Plan Gen-T grant CIDEXG/2022/013 of the *Generalitat Valenciana*.

*Corresponding author

to the classification task. This misdirection undermines the DNN’s ability to generalize, resulting in a significant gap between training and validation scores. To address this problem, regularization techniques have emerged as essential tools in modern Deep Learning (DL) [4], [5]. Indeed, understanding and enhancing generalization has become a central focus of contemporary research, as highlighted by works such as [6] and [7].

In this paper, we delve into the role of one specific regularization technique: L_2 regularization [8], [9]. This approach modifies the standard loss function \mathcal{L} by adding an extra term that penalizes large weight values. Specifically, the modified loss function is given by

$$\mathcal{L}' = \mathcal{L} + \lambda \cdot \|\omega\|_2^2,$$

where $\|\omega\|_2$ represents the L_2 norm of the DNN weights, and λ serves as a hyperparameter that dictates the strength of the regularization effect. By penalizing overly large weight values, L_2 regularization inherently guides the DNN toward solutions that are more robust and less prone to overfitting.

The importance of this technique is widely recognized by the scientific community [10], and its inclusion has not only become a standard practice but has also driven innovations such as the AdamW optimizer [11]. However, a persistent challenge lies in determining the optimal WD (hyperparameter) value. Striking the right balance is critical, as a very small (or nonexistent) value for the WD fails to counteract the inherent tendency of DNNs to overfit, whereas an excessively large value imposes overly strict constraints on the L_2 norm of the parameters, severely limiting the network’s learning capability and inevitably leading to an underfitting scenario. The process often requires exhaustive and costly hyperparameter search techniques, including grid search, random search, or more advanced strategies like Bayesian optimization and tools such as Optuna [12]. Commonly, values for λ are explored within the range of 10^{-5} to 10^{-2} , with final decisions based on validation scores. Despite these efforts, identifying the ideal

WD remains an open question, underscoring the need for more efficient and principled approaches.

To pave the way in this direction, we introduce a novel indicator, the Overfitting-Underfitting Indicator (OUI), designed to provide on-the-fly insights into the training dynamics of DNNs. OUI offers a quantitative measure, ranging between 0 and 1, that captures the DNN’s tendency toward underfitting (values close to 0) or overfitting (values close to 1). One of the characteristics of this indicator that sets it apart from the state-of-the-art techniques to tune the WD is its reliance solely on the DNN’s *activation patterns* [13], [14]—*i.e.*, the specific configuration of neurons that activate or remain inactive for each training sample—extracted during the forward pass, without requiring access to labels or validation data. By efficiently leveraging this intrinsic information, OUI enables a precise and computationally lightweight assessment of the DNN’s behavior throughout the training process. This approach not only deepens our understanding of the interplay between λ and the under- and overfitting problems, but also establishes a foundation for optimizing training strategies for complex DNN architectures.

OUI carries profound theoretical significance, as it encapsulates the DNN’s ability to leverage its expressive power. Low OUI values correspond to models with limited variability in activation patterns, indicating behavior closer to that of a linear function and signaling underfitting. Conversely, high OUI values indicate an excessive exploitation of the DNN’s expressive power, leading to erratic and overly specific activation patterns that strongly correlate with overfitting. In this sense, OUI provides a quantitative lens through which to evaluate the balance of a DNN’s learning dynamics between these two extremes.

In this work, we make the following key contributions:

- We introduce OUI, a novel indicator that quantifies the balance between overfitting and underfitting in DL models, providing a robust theoretical and empirical framework for understanding activation pattern variability across samples.
- We demonstrate that OUI can serve as a practical tool for selecting an appropriate value of the WD by monitoring its evolution during the very early stages of training. Specifically, models whose OUI values fall within an intermediate range early on training tend to achieve higher validation scores, indicating that such intermediate OUI values can reliably guide WD selection.
- We provide experimental validation showcasing OUI’s predictive power in identifying optimal regularization strategies on DenseNet-BC-100 with CIFAR-100, EfficientNet-B0 with TinyImageNet and ResNet-34 with ImageNet-1K.
- We propose a guideline for the ideal OUI range during training, offering a novel perspective on regularization that reduces computational overhead due to WD search.

Beyond its theoretical importance, OUI proves to be a practical tool for optimizing training strategies in DNNs. It offers a powerful mechanism for determining the ideal

value for the WD during training, as demonstrated in our experiments. Specifically, the optimal WD value should enable OUI to evolve throughout training in a way that ultimately stabilizes within a prescribed interval, as determined in this article. This capability not only refines regularization strategies but also highlights OUI’s potential as a guiding indicator for robust and efficient model training.

This article is structured as follows: Section II reviews related literature on WD behavior, overfitting detection, and activation pattern interpretation in DNNs. Section III formally defines OUI, providing its mathematical formulation and theoretical justification as a reliable indicator of a model’s expressive power. Section IV presents empirical validation, demonstrating OUI’s effectiveness in tracking underfitting, overfitting, and optimizing regularization parameters. Finally, Section V summarizes the findings and discusses their broader implications for DL regularization.

II. RELATED WORK

Optimizing the WD hyperparameter has been a focal point of research in DNN training, with numerous strategies proposed to enhance its effectiveness. Dynamic adjustment techniques have garnered significant attention, as explored in [11], [15], [16] and [17], where adaptive mechanisms aim to balance regularization strength throughout training. Another line of inquiry considers layer-specific adjustments, with [18] proposing strategies to tailor the WD parameter based on the depth of the layer, reflecting the unique role of each layer in the learning process.

Beyond WD, considerable effort has been directed toward designing metrics to detect overfitting without relying on validation data. For example, [19] introduced a novel approach that identifies overfitting by analyzing only the weights of a Convolutional Neural Network (CNN), bypassing the need for additional data. Similarly, [20] focused on the vulnerabilities inherent to overfitted models, developing a method to quantify these issues solely from training data. These contributions have paved the way for more robust training diagnostics that do not depend on external validation sets, a property that OUI also exhibits.

While these advancements provide valuable insights into regularization and overfitting detection, recent studies have increasingly highlighted the critical role of neural activation patterns in understanding and enhancing the expressive power of DNNs. Activation patterns offer a complementary perspective by focusing on the behavior of neurons during training, shedding light on the DNNs’ expressive power and their ability to generalize. [21] pioneered a visualization technique that reveals the input stimuli activating specific feature maps in DNNs, providing a deeper understanding of feature evolution during training. Building on this, [22] presented a comprehensive review of methods for interpreting DNNs, emphasizing the importance of activation patterns in improving model transparency and interpretability.

In parallel, other works have explored how activation patterns evolve during training. [14] introduced an efficient

method to record activation statistics—such as entropy and pattern similarity—during the training of Rectified Linear Unit (ReLU) DNNs, offering insights into the dynamic changes in these patterns. Additionally, [23] proposed a theoretical framework to analyze activation pattern evolution as stochastic processes, linking these changes to control overfitting. Together, these contributions illustrate the power of activation pattern analysis as a key component for diagnosing and mitigating training challenges in DL. Moreover, a novel perspective is provided by [24], [25], who examine the distinct convergence properties of structural knowledge—an abstraction of activation patterns—compared to the overall network training process. Their findings suggest that these structural patterns stabilize significantly earlier than the network’s learned parameters, which allows leveraging this property to freeze what the authors define as structural knowledge and separate it from quantitative knowledge. This highlights the potential of leveraging activation dynamics for more efficient training strategies, and reinforces the idea that activation patterns encode essential information about the learning trajectory of a model, which can be exploited to optimize training efficiency and reduce computational overhead.

Building on these foundations, our work bridges these lines of research by proposing a novel indicator that integrates the insights gained from activation patterns into the framework of overfitting and underfitting analysis. By quantifying how activation patterns evolve during training and correlating them with regularization strategies, our approach offers a unified perspective that enhances model training and optimization. Specifically, OUI provides a practical and efficient tool to monitor training dynamics, enabling the identification of the optimal WD parameter in the early stages of the training process and ensuring better validation scores.

III. DEFINITION OF OUI

A. Activation Functions and Expressive Power in DNNs

To formally define the Overfitting-Underfitting Indicator (OUI), we begin by exploring key concepts fundamental to its formulation. OUI is computed based on the activation patterns of a DNN, whose expressive power stems from its activation functions. These functions introduce the non-linearities that allow DNNs to model complex data relationships, distinguishing them from simple linear models [1]. ReLU is among the most commonly used activation functions. Appearing in architectures such as ResNet [26], VGG [27], and DenseNet [28], this activation function has remained widely used since its success in ImageNet-1K [29], [30]. More recent alternatives, such as Gaussian Error Linear Unit (GELU), commonly used in Transformer-based architectures like ViT [31], [32], and Sigmoid Linear Unit (SiLU), which was popularized through its use in EfficientNet [33], provide smoother transitions for negative inputs. Among these, a relevant subset of activation functions follows a characteristic pattern: for positive inputs, they return values close to the input itself, while for negative inputs, they produce small residual values, much lower in magnitude. This class, which we refer to as ReLU-like activations,

encompasses functions sharing this behavior, including but not limited to ReLU, GELU, and SiLU. Throughout this article, we focus exclusively on this type of activation functions, as their structure is key to the formulation of OUI. Concretely, the *activation state* of a neuron with a ReLU-like activation for a given sample is defined as *active* if its output is greater than zero and *inactive* otherwise. This distinction thus varies depending on the activation function: ReLU strictly outputs zero for inactive neurons, whereas activations like GELU and SiLU return small residual negative values, reflecting a gradual rather than abrupt transition between activation states.

Since activation functions play a fundamental role in the expressive power of DNNs, their evolution during training must be carefully monitored. Allowing the weights of a DNN to grow excessively leads to large-magnitude neuron outputs, yielding highly volatile activation states across different samples. Conversely, imposing strong constraints through a high WD value encourages weight magnitudes to remain too small, drastically reducing the variability in activation states across the network. Striking a balance between these two extremes is essential to ensure that the DNN can effectively leverage its expressive power and achieve optimal validation performance.

B. Understanding Activation Patterns through OUI

In this context, the concept of an activation pattern becomes central to understanding and analyzing DNN behavior. When a specific sample is propagated through the DNN, it elicits an activation state from each neuron. This response can be represented mathematically as an *activation pattern*: a binary vector that represents the activation state of neurons in response to a specific input. Consider a training set with m samples denoted by $X = \{x_1, \dots, x_m\}$ and a DNN with L hidden layers, and neurons n_1, n_2, \dots, n_k on a fixed layer $l \in \{1, \dots, L\}$.¹ For each training sample x_i and layer l , the activation pattern $P_l(x_i)$ is a vector of length k defined as follows: the j -th element of $P_l(x_i)$, denoted $P_l(x_i)_j$, is equal to 1 if the neuron n_j is activated by the input x_i , and 0 otherwise, where $i \in \{1, \dots, m\}$ and $j \in \{1, \dots, k\}$. This encoding provides a compact and precise way to capture the activation states of all neurons in a layer l for any input x_i , offering valuable insights into how the DNN processes information. For a visual interpretation of the activation pattern concept, see Figure 1.

The indicator proposed in this work, OUI, builds upon these patterns to assess how effectively the DNN distinguishes between samples. By quantifying the variability and distinctiveness of activation patterns across inputs, the indicator reveals the degree to which the DNN is leveraging its expressive power to represent and learn meaningful features.

The underlying concept is intuitive: a DNN that underutilizes its expressive power fails to fully exploit its non-linearities, resulting in neurons that remain consistently active

¹Naturally, the number of neurons k in layer l depends on l , although this dependency is not explicitly expressed to maintain clarity in notation. Since we consider l as a fixed but arbitrarily chosen element from $\{1, \dots, L\}$, we rely on context to convey this dependence. We trust that the reader will accept this slight abuse of notation for the sake of readability.

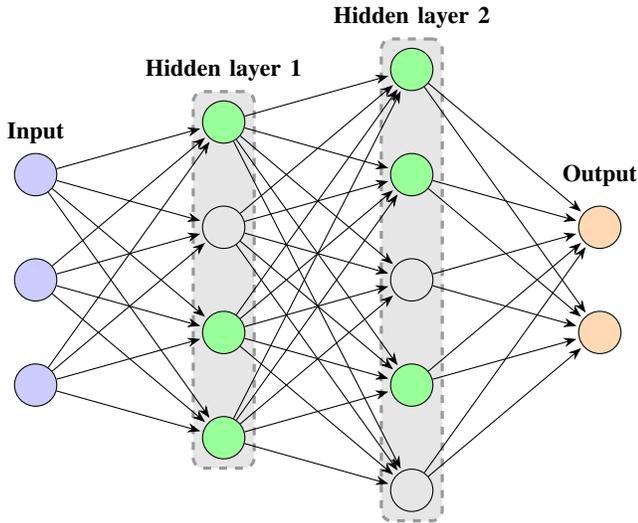


Fig. 1. Simple DNN to illustrate the concept of an activation pattern, capturing which neurons are active (green) or inactive (grey) for a specific input sample along the hidden layers of the DNN. For the illustrated input x , the activation pattern for the two hidden layers are $P_1(x) = [1, 0, 1, 1]$ and $P_2(x) = [1, 1, 0, 1, 0]$.

or inactive for all samples. In such cases, activation patterns become overly similar for different samples, indicating that the DNN is underfitting the training data. Conversely, when a DNN uses its expressive power to the fullest, neurons exhibit highly distinct activation states, which is a signal for overfitting. Motivated by these observations, we aim to formalize this idea into a measurable indicator, namely OUI.

In order to quantify the proximity between different activation patterns, we compute the normalized Hamming distance $d_H(P_l(x_i), P_l(x_j))$, which is a value that ranges from 0 to 1 and indicates the proportion of neurons that have distinct behavior for the samples x_i and x_j .² For instance, consider the first hidden layer of the DNN of Figure 1, containing four neurons. Suppose we have two input samples, x_1 and x_2 , whose activation patterns in this layer are given by $P_1(x_1) = [0, 1, 0, 0]$ and $P_1(x_2) = [1, 1, 0, 0]$. The normalized Hamming distance between these activation patterns is $d_H(P_1(x_1), P_1(x_2)) = 0.25$, indicating that 25% of the neurons in the layer exhibit different activations, while the remaining neurons share the same activation state.

Hence, the extreme value $d_H(P_l(x_i), P_l(x_j)) = 0$ only occurs when the activation patterns of x_i and x_j are identical. Small values of $d_H(P_l(x_i), P_l(x_j))$ among the layers l of the DNN and the possible couples of training samples (x_i, x_j) suggest a strong positive correlation between activation patterns, implying that the DNN may be underusing its expressive power by assigning nearly the same activation pattern to distinct samples, hence manifesting underfitting.

Determining the value of d_H that indicates overfitting is more nuanced. Consider a DNN that assigns activation patterns

²Notice that d_H does not measure a distance between the overall activation patterns. Instead, it is computed separately for each layer l , comparing the activations $P_l(x_i)$ and $P_l(x_j)$.

in a highly erratic way as a consequence of an overuse of its expressive power. In this extreme overfitting scenario, where activation patterns are assumed to behave as random variables, and neurons are assumed to activate with a 50% chance, the normalized Hamming distance would be 0.5 on average. Thus, a distance of 0.5 is relevant for our purposes, as it reflects an extreme situation suggestive of strong overfitting.

Hence, the normalized Hamming distance $d_H(P_l(x_i), P_l(x_j))$ can be interpreted as an indicator of correlation among the activation patterns $P_l(x_i)$ and $P_l(x_j)$. Values of d_H increasing from 0 to 0.5 correspond to decreasing positive correlation, ranging from a correlation of 1 (when $d_H = 0$) to 0 (when $d_H = 0.5$). When d_H is close to 0, the activation patterns are nearly identical, meaning the DNN assigns almost the same structure to both samples. On the other hand, a value of $d_H = 0.5$ is consistent with patterns that appear random, reflecting a complete lack of correlation. Beyond this threshold, d_H values from 0.5 to 1 indicate growing negative correlation, culminating in a fully anti-correlated state at $d_H = 1$, where the activation of one sample consistently coincides with the deactivation of the other.

Because only positive correlations between the activation patterns $P_l(x_i)$ and $P_l(x_j)$ are relevant for detecting underfitting or overfitting, the value of OUI for layer l , denoted OUI_l , is computed by truncating distances at 0.5 and then averaging over all sample pairs in the set

$$Y = \{(i, j) : i, j \in \{1, \dots, m\}, i \neq j\}.$$

Finally, the computation of OUI for the entire DNN on the given dataset X follows by averaging OUI_l across all hidden layers l . Formally:

Definition 1. On a dataset $X = \{x_1, \dots, x_m\}$, the value of OUI for layer l is computed as

$$OUI_l = \frac{2}{|Y|} \sum_{(i,j) \in Y} \min(d_H(P_l(x_i), P_l(x_j)), 0.5),$$

and the value of OUI for the entire DNN corresponds to

$$OUI = \frac{2}{L \cdot |Y|} \sum_{l=1}^L \sum_{(i,j) \in Y} \min(d_H(P_l(x_i), P_l(x_j)), 0.5).$$

Note that the factor of 2 in the formulae ensures that the final value of the indicator is normalized to lie within the range $[0, 1]$. Both the minimum value of 0 and the maximum value of 1 represent asymptotic behaviors that the DNN can approach during training.

C. How OUI Correlates with DNN Generalization

OUI quantifies how far the activation patterns assigned by the DNN are from being either overly similar or excessively noisy. In other words, it measures the extent to which the DNN's behavior transitions between acting as a linear model and utilizing its expressive power erratically. This interpretation can be formalized and proven rigorously, as follows.

We define a DNN as suffering *chaotic activation dynamics* when the activation patterns of distinct samples differ in more than half of their states. Formally:

Definition 2. A DNN suffers *chaotic activation dynamics* on a dataset X if

$$d_H(P_l(x_i), P_l(x_j)) \geq 0.5$$

for every pair of distinct samples $x_i, x_j \in X$ and every hidden layer l of the DNN.

It is important to emphasize that this definition represents an extreme and theoretical case, that will be shown to correspond with $\text{OUI} = 1$ in the following proposition. In practical scenarios, reaching such a state through natural training is virtually impossible. Instead, this condition defines a conceptual upper bound that allows us to formalize the idea of maximal activation pattern disorder. OUI, in turn, can be interpreted as measuring how close a given DNN is to such an extreme scenario. In the upcoming proposition, we will formally establish how this notion relates to OUI and its limits.

Similarly, the lower bound $\text{OUI} = 0$ is shown to correspond to a scenario in which the DNN behaves exactly like a linear model, meaning that its activation patterns remain fully structured and predictable, without taking advantage of the DNN’s non-linear expressive power. Just as achieving fully chaotic activation dynamics is infeasible in practice, obtaining a perfectly linear response in a deep DNN trained on complex data is equally unrealistic. Thus, OUI provides a means to quantify how closely a DNN approximates either of these two theoretical extremes.

With this framework in place, we rigorously establish the described extreme cases for OUI:

Proposition 1. *A DNN with ReLU activations satisfies:*

- $\text{OUI} = 0$ if and only if the DNN behaves exactly as a linear model on the training set.
- $\text{OUI} = 1$ if and only if the DNN suffers chaotic activation dynamics on the training set.

Remark 1. The previous proposition extends naturally to any piecewise linear activation function whose set of nonlinearities is exactly $\{0\}$. This includes not only ReLU but also its generalizations such as Parametric Rectified Linear Unit (PReLU) and Leaky ReLU.

The proof of this proposition is provided in the Appendix B of this document, as well as a remark regarding the cases in which the activation is not linear outside the origin.

Consequently, OUI serves as a measure of the DNN’s complexity with respect to its activation patterns. By construction, it directly correlates with the presence of overfitting or underfitting during the training process, offering valuable insights into the DNN’s behavior and generalization ability.

Naturally, this relationship is influenced by the choice of DNN architecture and dataset. For example, a highly sophisticated DNN may utilize only a fraction of its expressive power and still achieve near-perfect validation accuracy on a simple

dataset like MNIST. Conversely, a less complex DNN trained on a challenging dataset like ImageNet-1K might fully exhaust its expressive power yet fail to generalize, instead learning spurious noise patterns that lead to overfitting. Thus, when the DNN architecture is well-matched to the dataset’s complexity, OUI provides a sharper correlation between activation pattern differences and the balance between underfitting and overfitting.

One additional strength of OUI lies in its computational efficiency. By leveraging statistical evidence, the indicator can be approximated by averaging over a subset of sample pairs from a batch with a relative error of less than 5%. This design allows OUI to be computed dynamically during training, with batch-level values aggregated to determine the indicator for each epoch. Such an approach drastically reduces the computational cost and memory requirements, ensuring that OUI remains practical for use even in large-scale training scenarios. For more details on the computational aspects and implementation of OUI, we refer the reader to the Appendix A.

Given its ability to quantify the nuanced interplay between overfitting and underfitting, OUI offers more than just theoretical insight—it becomes a tool to address practical challenges in training. In particular, its relationship with regularization techniques, such as L_2 regularization, raises a compelling question: How does OUI evolve during training for different values of the WD?

This pivotal question lays the foundation for the next section, where we present a series of experiments designed to explore this relationship in depth. Beyond understanding the interaction between OUI and WD, these experiments demonstrate that OUI can be harnessed to determine the optimal WD value early in training, offering a practical pathway to more efficient and effective DNN training.

IV. RELATIONSHIP BETWEEN OUI AND L_2 REGULARIZATION

In this section, we investigate the role of OUI as a tool for selecting a WD value in DNNs, with a particular emphasis on CNN architectures due to their structured activation patterns and widespread use in image recognition tasks. Although the principles underlying OUI are general and applicable to a wide variety of DNNs, our experimental validation specifically targets CNN-based models, given their clear interpretability and relevance in practical scenarios.

Exploring the behavior of OUI in other architectures, such as Transformers, is left for future work, as tuning OUI values across the fully connected components of these models demands a more in-depth investigation—one that goes beyond the scope of this initial study and cannot be treated as just another experiment to include.

We explore how the evolution of OUI throughout training reflects a CNN’s capacity to generalize, aiming to determine whether specific OUI dynamics can be associated with optimal regularization. To this end, we analyze the behavior of OUI under different WD values, examining its stability and progression over the course of training. By doing so,

we seek to establish a systematic approach to leverage OUI for regularization tuning, minimizing the need for exhaustive hyperparameter searches.

Additionally, we study how the early evolution of OUI correlates with the final validation accuracy of the DNN models. In particular, we assess whether a rapid transition of OUI from its initial state toward a stable regime is indicative of effective training dynamics. Understanding these relationships will provide valuable insights into the practical utility of OUI as a guiding metric for training and optimizing DNN architectures.

A. Experimental Setup

To evaluate the hypothesis that OUI can guide the selection of the optimal WD during training, we conducted experiments on three distinct DNNs and datasets:

1) *DenseNet-BC-100* ($k = 12$) with *CIFAR-100*, representing a classification task with 100 classes in images of moderate scale. Training was performed over 200 epochs using Stochastic Gradient Descent (SGD) with momentum of 0.9 and batch size of 64. The learning rate followed a cosine annealing schedule, gradually reducing from an initial value of 0.01. WD values ranged from $3.16 \cdot 10^{-5}$ to $3.16 \cdot 10^{-2}$, sampled logarithmically to ensure coverage of underfitting, overfitting, and intermediate scenarios. The *CIFAR-100* dataset was processed with data augmentation strategies that included cropping, flipping, augmenting, and normalization.

2) *EfficientNet-B0* with *TinyImageNet*, representing a mid-scale classification task with 200 classes. Training was conducted over 150 epochs with a batch size of 64 also using SGD with a momentum of 0.9. The learning rate increased linearly from 10^{-3} to 10^{-2} during the first 15 epochs (warm-up phase), followed by a cosine annealing schedule for the remainder of the training. WD values were sampled logarithmically from 10^{-5} to 10^{-2} . Data preprocessing included cropping, flipping, color jittering, and normalization.

3) *ResNet-34* with *ImageNet-1K*, a more complex classification task with 1,000 classes in a large-scale dataset. Training was conducted over 90 epochs with a batch size of 256, and SGD with momentum 0.9 as usual. A two-phase learning rate schedule was employed: an initial warm-up phase over the first 5 epochs from 0.01 to 0.1, followed by cosine annealing from 0.1 to 0 for the remaining epochs. WD values were selected from a range of $3.16 \cdot 10^{-6}$ to $3.16 \cdot 10^{-3}$, similarly spaced to explore diverse regularization regimes. Input preprocessing involved cropping, flipping, and color adjustments.

For each case, the training process explored a range of WD values to cover scenarios of underfitting, overfitting, and an optimal balance between regularization and expressive power. By capturing OUI alongside training and validation loss and accuracy, we examined their relationship and tested whether OUI dynamics can predict the optimal WD early in training.

OUI values were tracked at each epoch, providing insight into their evolution alongside training and validation loss. This setup enabled a detailed comparison of the effect of different

WD values and the predictive capability of OUI in identifying optimal training configurations.

The training process for the experiments was designed to isolate the effect of WD on OUI and its relationship with the final validation accuracy. To achieve this, all models were trained from scratch, without using preloaded weights, in a controlled Python environment with `PyTorch`. The trainings were conducted on a single NVIDIA A100 GPU with 80 GB of memory, leveraging `PyTorch`'s optimized GPU acceleration to efficiently process large-scale datasets while maintaining consistency across experiments.

The full code required to reproduce the experiments, including training scripts and configurations, is available at the link provided at the end of the abstract.

B. Results and Analysis

The results of the experiments are summarized in Figure 2. Each column corresponds to a pair of DNN and dataset. Four plots are displayed for each setting, enabling an analysis of OUI dynamics and their relationship with key training and validation metrics.

For each column, the plots in the top row show the evolution of OUI for the selected WD values equispaced logarithmically, illustrating how different regularization strengths affect OUI behavior during training. The remaining plots provide a detailed view of training and validation loss dynamics, as well as OUI evolution, for three representative WD values:

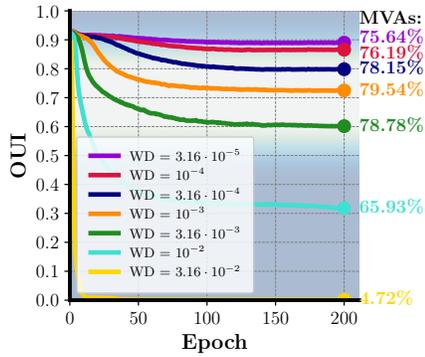
- Low WD (second row): Reflects weak regularization and high OUI values, characteristic of overfitting.
- Intermediate WD (third row): Demonstrates stable OUI within intermediate values, optimal generalization, and the highest validation accuracy.
- High WD (fourth row): Exhibits rapid decreases in OUI, underfitting, and poor generalization.

The title on top of each plot indicates the corresponding WD, as well as the Maximum Validation Accuracy (MVA) achieved during each training process.

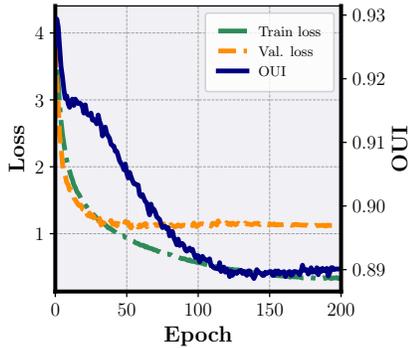
DenseNet-BC-100 and CIFAR-100: The results summarized in the leftmost column of Figure 2 highlight OUI's ability to distinguish between underfitting, overfitting, and optimal regularization. Low WD values ($3.16 \cdot 10^{-5}$, 10^{-4} , $3.16 \cdot 10^{-4}$) maintain OUI near 0.9 throughout training, reflecting insufficient regularization and overfitting. In contrast, high WD values (10^{-2} , $3.16 \cdot 10^{-2}$) cause OUI to drop below 0.6 early during training, indicative of excessive regularization and underfitting. Intermediate values (10^{-3} , $3.16 \cdot 10^{-3}$) stabilize OUI by epoch 30 around 0.7, achieving the highest validation accuracies (78.8% to 79.5%). These results emphasize OUI's predictive power: models with OUI reaching reasonable intermediate values within the first 15% of epochs show optimal generalization, whereas models with extreme OUI behavior fail to generalize effectively.

EfficientNet-B0 and TinyImageNet: The experiments shown in the middle column of Figure 2 align with the patterns observed in the other two experiments while also illustrating dataset-specific nuances. Low WD values (10^{-5} , $3.16 \cdot 10^{-5}$,

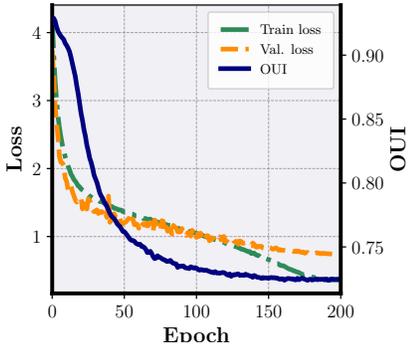
DenseNet-BC-100 and CIFAR-100



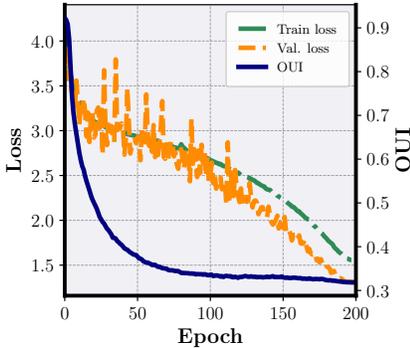
WD = $3.16 \cdot 10^{-5}$, MVA = 75.64 %



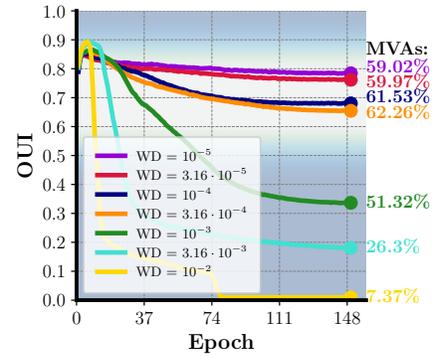
WD = 10^{-3} , MVA = 79.54 %



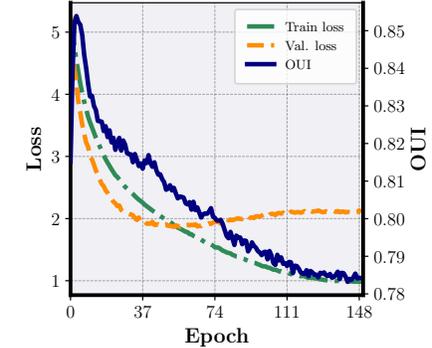
WD = 10^{-2} , MVA = 65.93 %



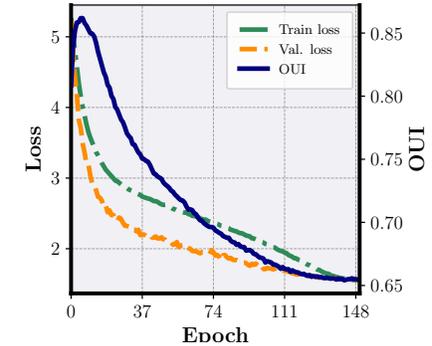
EfficientNet-B0 and TinyImageNet



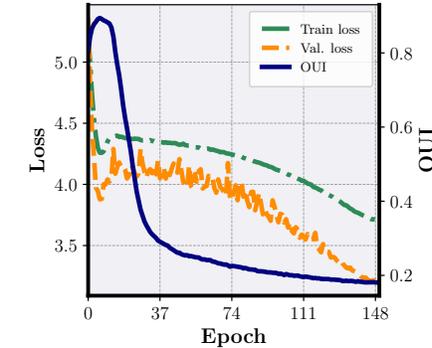
WD = 10^{-5} , MVA = 59.02 %



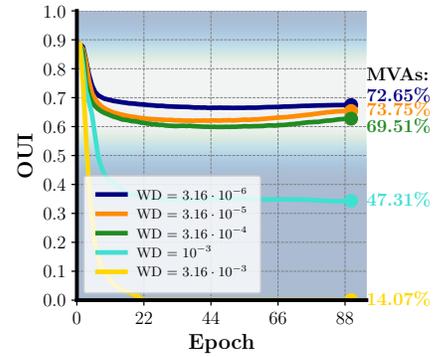
WD = $3.16 \cdot 10^{-4}$, MVA = 62.26 %



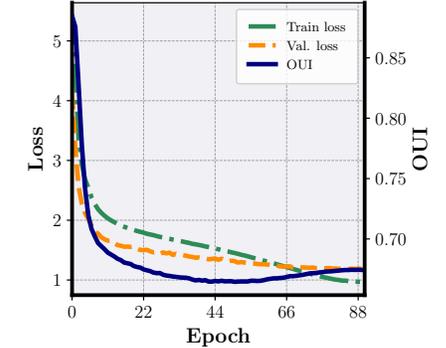
WD = $3.16 \cdot 10^{-3}$, MVA = 26.3 %



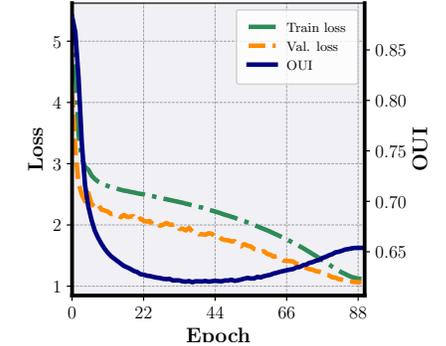
ResNet-34 and ImageNet-1K



WD = $3.16 \cdot 10^{-6}$, MVA = 72.65 %



WD = $3.16 \cdot 10^{-5}$, MVA = 73.75 %



WD = 10^{-3} , MVA = 47.31 %

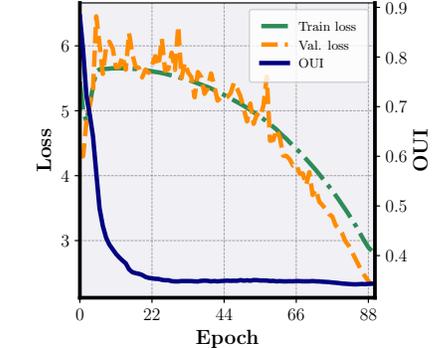


Fig. 2. Comparison of OUI evolution and training dynamics across different architectures and datasets. Each column corresponds to an experiment: DenseNet-BC-100 on CIFAR-100 (left), EfficientNet-B0 on TinyImageNet (center), and ResNet-34 on ImageNet-1K (right). Each row represents a specific analysis: the first row shows OUI trajectories for multiple WD values, while the second, third, and fourth rows correspond to training and validation loss dynamics for low, intermediate, and high WD values, respectively.

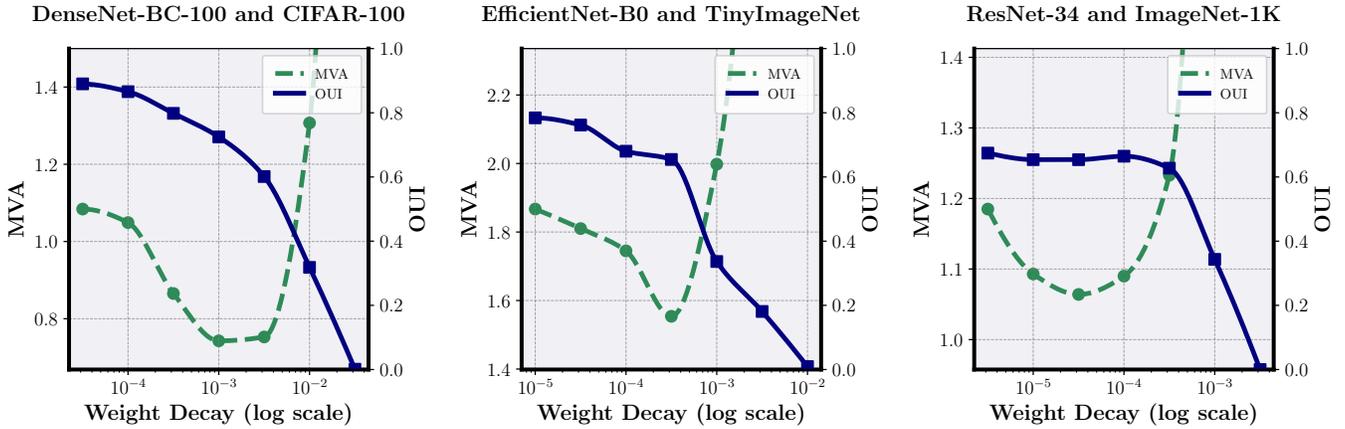


Fig. 3. MVA and final OUI as functions of WD for DenseNet-BC-100, EfficientNet-B0 and ResNet-34.

10^{-4}) keep OUI consistently high—near 0.8—throughout training, indicating mild overfitting and suboptimal generalization. The highest validation accuracy (MVA: 62.26%) is achieved when WD is set to $3.16 \cdot 10^{-4}$, with OUI stabilizing around 0.65. For larger WD values (10^{-3} and above), OUI quickly drops below 0.6—in some cases as early as epoch 20, as observed with $WD = 3.16 \cdot 10^{-3}$ —signaling excessive regularization and underfitting. These results reinforce the utility of OUI as an early indicator of generalization performance.

ResNet-34 and ImageNet-1K: The results presented in the rightmost column of Figure 2 confirm the trends observed in the first two experiments, with notable distinctions due to the dataset’s complexity. Low WD values, such as $3.16 \cdot 10^{-6}$, stabilize OUI near 0.7, reflecting mild overfitting but yielding relatively high accuracies. Some of the studied low WD values had to be removed from the first plot in the rightmost column in order to prevent overlapping and to ensure that the final plot was clearly legible and free of redundancies. High WD values, such as 10^{-3} , rapidly reduce OUI below 0.6 by epoch 5, resulting in underfitting and poor generalization (MVA: 47.31%). The intermediate WD value $3.16 \cdot 10^{-5}$ stabilizes OUI by epoch 10 (less than 12% of total epochs) near 0.65, achieving the highest validation accuracy (MVA: 73.75%). Interestingly, the value $3.16 \cdot 10^{-4}$ shows slightly lower MVA than $3.16 \cdot 10^{-5}$, despite similar OUI behavior, which can be attributed to ImageNet-1K’s complexity. Unlike the previous datasets, ImageNet-1K makes underfitting more likely than overfitting, highlighting how OUI reflects the interplay between the model and dataset complexities. These results further establish OUI as a robust tool for assessing regularization dynamics in diverse settings.

Validation Loss and OUI Trends: Figure 3 presents the relationship between WD values and two key metrics: validation loss and final OUI value. The x -axis represents different WD values on a logarithmic scale, while the y -axes display the final OUI value and the validation loss reached at the end of training. The results reveal a strong inverse trend: while OUI generally decreases as WD increases, minor deviations appear due to experimental variability, yet the overall pattern

aligns with theoretical predictions. Meanwhile, validation loss exhibits a characteristic U-shaped curve, reaching its minimum within an optimal range of WD values. Notably, the OUI values associated with this optimal region for validation loss—corresponding to values of WD between $3.16 \cdot 10^{-4}$ to $3.16 \cdot 10^{-3}$ for the experiment on DenseNet-BC-100, 10^{-4} to $3.16 \cdot 10^{-4}$ on EfficientNet-B0 with TinyImageNet, and 10^{-5} to 10^{-4} for ResNet-34 trainings—consistently fall within the interval $[0.6, 0.8]$, experimentally validating that this interval corresponds to WD values that yield the best generalization. This observation is highly practical for training deep networks: since OUI converges well before training completion, it serves as an early indicator of whether a given weight decay setting will lead to optimal validation accuracy.

V. CONCLUSION

This work introduced OUI, a novel indicator designed to evaluate the training dynamics of DNNs and to provide insights into the relationship between regularization, expressive power, and generalization. By analyzing OUI’s behavior during training, we demonstrated that it serves as a powerful tool for identifying the optimal WD value. Specifically, we showed that maintaining OUI within the range $[0.6, 0.8]$ corresponds to the best balance between underfitting and overfitting, leading to improved validation accuracy across different datasets and architectures.

Our experiments on DenseNet-BC-100 with CIFAR-100, ResNet-34 with ImageNet-1K and ViT-16 with TinyImageNet validated the effectiveness of OUI as an early indicator for selecting WD. In these cases, OUI was found to converge significantly faster than training and validation loss, enabling the identification of optimal WD values within the first 15% of the epochs of the training process. This rapid convergence highlights the practicality of OUI in reducing the computational cost of hyperparameter tuning, as fewer training epochs are required to assess a model’s potential. Thus, OUI offers a compelling perspective on the interplay between underfitting and overfitting, providing a reliable and computationally efficient means to optimize training strategies.

A promising direction for future work is the development of an automatic WD adjustment mechanism based on OUI. Such a callback would dynamically tune the WD hyperparameter during training by observing OUI values and ensuring they remain within a desired range. This approach has the potential to streamline the hyperparameter tuning process, reduce computational costs, and adapt to evolving training dynamics on-the-fly. Additionally, the integration of OUI-based callbacks into standard training frameworks could further enhance its adoption and usability in practical applications.

Another direction involves broadening the range of neural architectures to which the metric is applied, particularly to investigate its behavior during the training of natural language models such as transformers. The technical complexity of training such systems makes this exploration more suitable for future work. Furthermore, transformers differ substantially from CNNs in that activations only occur in the feed-forward modules between attention layers. The diverse behavior of these blocks as a function of network depth introduces additional challenges, making a comprehensive analysis of the OUI metric in this context too extensive to be treated as just another case within the scope of this article.

By addressing these challenges, we aim to extend the generalizability and utility of OUI, paving the way for more efficient and adaptive training strategies in DL.

REFERENCES

- [1] G. Cybenko, "Approximation by superpositions of a sigmoidal function," *Mathematics of Control, Signals and Systems*, vol. 2, no. 4, pp. 303–314, 1989.
- [2] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, no. 5, pp. 359–366, 1989.
- [3] Z. Lu, H. Pu, F. Wang, Z. Hu, and L. Wang, "The expressive power of neural networks: A view from the width," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017.
- [4] X. Ying, "An overview of overfitting and its solutions," *Journal of Physics: Conference Series*, vol. 1168, no. 2, p. 022022, feb 2019.
- [5] M. M. Bejani and M. Ghatee, "A systematic review on overfitting control in shallow and deep neural networks," *Artificial Intelligence Review*, vol. 54, no. 8, pp. 6391–6438, 2021.
- [6] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning (still) requires rethinking generalization," *Commun. ACM*, vol. 64, no. 3, p. 107–115, Feb. 2021.
- [7] T. Poggio, K. Kawaguchi, Q. Liao, B. Miranda, L. Rosasco, X. Boix, J. Hidary, and H. Mhaskar, "Theory of deep learning iii: explaining the non-overfitting puzzle," 2018. [Online]. Available: <https://arxiv.org/abs/1801.00173>
- [8] A. Krogh and J. Hertz, "A simple weight decay can improve generalization," in *Advances in Neural Information Processing Systems*, J. Moody, S. Hanson, and R. Lippmann, Eds., vol. 4. Morgan-Kaufmann, 1991.
- [9] S. Bos and E. Chug, "Using weight decay to optimize the generalization ability of a perceptron," in *Proceedings of International Conference on Neural Networks (ICNN'96)*, vol. 1, 1996, pp. 241–246 vol.1.
- [10] F. D'Angelo, M. Andriushchenko, A. Varre, and N. Flammarion, "Why do we need weight decay in modern deep learning?" 2024.
- [11] F. H. Ilya Loshchilov, "Decoupled weight decay regularization," in *7th International Conference on Learning Representations, ICLR 2019*, 2019.
- [12] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna: A next-generation hyperparameter optimization framework," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2019.
- [13] X. Huang, D. Kroening, W. Ruan, J. Sharp, Y. Sun, E. Thamo, M. Wu, and X. Yi, "A survey of safety and trustworthiness of deep neural networks: Verification, testing, adversarial attack and defence, and interpretability," *Computer Science Review*, vol. 37, p. 100270, 2020.
- [14] D. Hartmann, D. Franzen, and S. Brodehl, "Studying the evolution of neural activation patterns during training of feed-forward ReLU networks," *Frontiers in Artificial Intelligence*, vol. 4, 2021.
- [15] K. Nakamura and B. Hong, "Adaptive weight decay for deep neural networks," *IEEE Access*, vol. 7, pp. 118 857–118 865, 2019.
- [16] A. Lewkowycz and G. Gur-Ari, "On the training dynamics of deep networks with L_2 regularization," in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, Eds., vol. 33, 2020, pp. 4790–4799.
- [17] M. A. Ghiasi, A. Shafahi, and R. Ardekani, "Improving robustness with adaptive weight decay," in *Advances in Neural Information Processing Systems*, A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, Eds., vol. 36. Curran Associates, Inc., 2023, pp. 79 067–79 080.
- [18] M. Ishii and A. Sato, "Layer-wise weight decay for deep neural networks," in *Image and Video Technology*, M. Paul, C. Hitoshi, and Q. Huang, Eds. Cham: Springer International Publishing, 2018, pp. 276–289.
- [19] S. Watanabe and H. Yamana, "Overfitting measurement of convolutional neural networks using trained network weights," *International Journal of Data Science and Analytics*, vol. 14, no. 3, pp. 261–278, 2022.
- [20] H. Rezaei and M. Sabokrou, "Quantifying overfitting: Evaluating neural network performance through analysis of null space," 2023.
- [21] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Computer Vision – ECCV 2014*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds. Cham: Springer International Publishing, 2014, pp. 818–833.
- [22] G. Montavon, W. Samek, and K.-R. Müller, "Methods for interpreting and understanding deep neural networks," *Digital Signal Processing*, vol. 73, pp. 1–15, 2018.
- [23] S. J. Lehmler, M. S. ur Rehman, T. Glasmachers, and I. Iossifidis, "Understanding activation patterns in artificial neural networks by exploring stochastic processes: Discriminating generalization from memorization," *Neurocomputing*, vol. 610, p. 128473, 2024.
- [24] J. Duato, J. I. Mestre, M. F. Dolz, and E. S. Quintana-Ortí, "GreenLightningAI: An efficient ai system with decoupled structural and quantitative knowledge," 2023. [Online]. Available: <https://arxiv.org/abs/2312.09971>
- [25] J. Duato, J. I. Mestre, M. F. Dolz, E. S. Quintana-Ortí, and J. Cano, "Decoupling structural and quantitative knowledge in relu-based deep neural networks," in *Proceedings of the 5th Workshop on Machine Learning and Systems*, ser. EuroMLSys '25. New York, NY, USA: ACM, 2025, p. 39–45.
- [26] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.
- [27] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *3rd International Conference on Learning Representations (ICLR 2015)*. Computational and Biological Learning Society, 2015, pp. 1–14.
- [28] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 2261–2269.
- [29] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255.
- [30] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 6, p. 84–90, May 2017.
- [31] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017.
- [32] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," in *International Conference on Learning Representations*, 2021.

- [33] M. Tan and Q. Le, “EfficientNet: Rethinking model scaling for convolutional neural networks,” in *Proceedings of the 36th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, K. Chaudhuri and R. Salakhutdinov, Eds., vol. 97. PMLR, 09–15 Jun 2019, pp. 6105–6114.
- [34] G. W. Snedecor and W. G. Cochran, *Statistical Methods*, 8th ed. Ames, IA: Iowa State University Press, 1989.

APPENDIX A COMPUTATION OF OUI AND ITS IMPACT ON TRAINING TIME

Understanding how OUI is computed and its effect on training efficiency requires a closer look at how activation patterns evolve throughout training. Since OUI leverages activation patterns from intermediate layers, its computation is naturally integrated into the forward pass. Instead of considering the entire dataset, we restrict calculations to each batch, allowing OUI to capture DNN behavior on-the-fly without excessive computational overhead. For each batch, we compute a running average of OUI in the same manner as it is done for computing the training loss, and the final OUI value for the epoch is given by the final running average across all batches where OUI was computed.

However, since OUI values change gradually over time, recalculating it at every batch is unnecessary. A more efficient approach is to update the indicator periodically. In our experiments, we found that computing OUI every 10 batches provides an optimal balance between computational cost and tracking accuracy.

One challenge in computing OUI efficiently is the sheer number of sample pairs in each batch. For example, with a batch size of 64, the number of possible pairs reaches $\binom{64}{2} = 2016$. Since OUI is derived from averaging these pairwise comparisons, a key consideration is determining the number of pairs necessary for a statistically reliable estimate. Following the classical sample size determination formula for estimating a population mean ([34], Section 2.14), the required number of samples can be computed as

$$n = \left(\frac{Z_{\alpha/2} \cdot \sigma}{E} \right)^2,$$

where $Z_{\alpha/2}$ is the critical value from the standard normal distribution corresponding to a $1 - \alpha$ confidence level, σ represents the standard deviation of OUI measurements for different pairs of samples, and E defines the acceptable error margin.

For our experiments, we set an error margin of $E = 0.05$, ensuring a relative error of 5% while adopting a 95% confidence level ($\alpha = 0.05$). The observed standard deviations σ across DNN layers suggested that a range of 3 to 19 sample

pairs would be sufficient for a stable estimate of OUI. To ensure robust statistical significance while maintaining efficiency, we opted for $\binom{8}{2} = 28$ pairs per batch, effectively reducing computational overhead without compromising accuracy. This is equivalent to using the number of pairs in a batch of size 8, while randomly selecting pairs from the available samples within a larger batch.

Despite the additional computations required, the overall impact on training time remains minimal. Empirical measurements indicate that OUI calculations account for only 3.6% of total epoch time, making it a highly efficient tool for monitoring training behavior without introducing significant costs.

APPENDIX B PROOFS

Proof of Proposition 1. It easily follows by taking into account that, as $0 \leq d_H \leq 1$, then

$$0 \leq \min(d_H(P_l(x_i), P_l(x_j)), 0.5) \leq 0.5.$$

Since OUI is computed as a mean of these terms for $(i, j) \in X$ and $l \in \{1, \dots, d\}$, it then follows that $\text{OUI} = 0$ if and only if $d_H(P_l(x_i), P_l(x_j)) = 0$ for every (i, j) and every l . This can only happen if the activation patterns of every training sample are equal, which forces the model to behave linearly on the training set if the activation function is linear outside the origin.

Similarly, $\text{OUI} = 1$ if and only if $\min(d_H(P_l(x_i), P_l(x_j)), 0.5) = 0.5$ for every pair (i, j) and every l . This is, in turn, equivalent to have that $d_H(P_l(x_i), P_l(x_j)) \geq 0.5$, which is the definition given for a DNN to suffer chaotic activation dynamics on the training set. \square

Remark 2. For ReLU-like activation functions that are not piecewise linear, such as GELU and SiLU, the case $\text{OUI} = 1$ still holds without issue. However, the equivalence between $\text{OUI} = 0$ and strictly linear behavior no longer follows directly. These activation functions introduce a smooth approximation to ReLU, allowing the DNN to behave in a more gradual manner. Despite this, since these functions closely resemble ReLU, a scenario in which all training samples share the same activation pattern still corresponds to a DNN with a capacity very similar to that of a linear model. While the strict notion of linearity may not fully apply, the practical implication remains: an extremely low OUI value indicates a model whose expressive power has been neutralized, potentially leading to underfitting.