THE SPACE BETWEEN: ON FOLDING, SYMMETRIES AND SAMPLING

Michal Lewandowski[†], **Bernhard Heinzl**[†], **Raphael Pisoni**[†], **Bernhard A.Moser**[†*]
[†]Software Competence Center Hagenberg (SCCH)
[*]Johannes Kepler University of Linz (JKU)
{name.surname}@scch.at

ABSTRACT

Recent findings suggest that consecutive layers of neural networks with the ReLU activation function *fold* the input space during the learning process. While many works hint at this phenomenon, an approach to quantify the folding was only recently proposed by means of a space folding measure based on Hamming distance in the ReLU activation space. We generalize this measure to a wider class of activation functions through introduction of equivalence classes of input data, analyse its mathematical and computational properties and come up with an efficient sampling strategy for its implementation. Moreover, it has been observed that space folding values increase with network depth when the generalization error is low, but decrease when the error increases. This underpins that learned symmetries in the data manifold (e.g., invariance under reflection) become visible in terms of space folds, contributing to the network's generalization capacity. Inspired by these findings, we outline a novel regularization scheme that encourages the network to seek solutions characterized by higher folding values.

1 INTRODUCTION

Recent works in machine learning indicate that neural networks *fold* the input space during training (Montúfar et al., 2014; Keup & Helias, 2022). This phenomenon draws inspiration from the way certain natural structures—such as proteins and amino acids—fold to encode information efficiently (Dill et al., 2008; Jumper et al., 2021). Building on these ideas, Lewandowski et al. (2025) proposed a range-based measure in the discrete activation space of ReLU networks to quantify how *much* a network folds its input space as it learns. Their analysis focuses on deviations from convexity when mapping a straight-line path in the input space to the Hamming activation space: While Euclidean distances increase monotonically along the path, the corresponding Hamming distances may decrease, signaling a folding effect (cf. Fig. 1 right and Fig. 2). Originally developed for ReLU networks, this approach leverages the fact that the ReLU activation function partitions the input space into disjoint linear regions (Makhoul et al., 1989; Montúfar et al., 2014).

In our paper, we firstly show that these regions correspond to equivalence classes defined by the pre-images of either $\{0\}$ or the strictly positive interval $(0, \infty)$. Extending $\{0\}$ to $(-\infty, 0]$ provides a straightforward generalization to a broader class of activation functions that accommodate negative values, including Swish (Ramachandran et al., 2018), GELU (Hendrycks & Gimpel, 2016), and SwiGLU (Shazeer, 2020). Secondly, we focus on characterizing properties of the space folding measure χ, which do also hold in the general case. Thirdly, since computing χ relies on sampling from different activation regions, we introduce a non-parametric sampling algorithm that exploits the structure of the aforementioned equivalence classes, thereby reducing redundant computations. Lastly, we leverage the fact that space folding values have been observed to *increase* with network depth when the generalization error is low, but *decrease* when the error increases (Lewandowski et al., 2025). We thus hypothesize the increased folding contribute to the network's generalization capacity, and hint at a novel regularization strategy that applies the folding measure at regular intervals (e.g., every n training epochs) to induce stronger folding in the early stages and diminish its influence later in training. Our contributions are as follows.

- We generalize the space folding measure beyond the ReLU activation function. Our approach relies on the fact that the pre-image of the partition $\{(-\infty, 0], (0, \infty)\}$ divides the domain into two connected sets, $f^{(-1)}((-\infty, 0])$ and $f^{(-1)}((0, \infty))$.
- We state and prove general properties of the folding measure, such as (*i*) its stability under traversing different activation regions (Proposition 4.1), (*ii*) the sufficient and necessary, i.e., characterizing, condition for flatness (Proposition 4.2), (*iii*) its sensitivity to the direction of the path (Remark 4.3), (*iv*) invariance of flatness to direction of path (Corollary 4.4).
- We propose a parameter-free sampling strategy in the Hamming activation space that limits steps within the same equivalence class to reduce redundancy, and we analyze its computational complexity.
- We introduce a new regularization procedure for training neural networks by penalizing low space folding values.

The remainder of the paper is organized as follows. Sec. 2 details related work; Sec. 3 introduces necessary concepts and fixes notation for the rest of the paper; Sec. 4 recalls the definition of the space folding measure and then provides its detailed analysis paired with the introduction of the global folding measure; Sec. 5 introduces a sampling technique from activation paths along a 1D path which relies on the Hamming distances between samples; Sec. 6 proposes a novel regularization scheme; Sec. 7 summarizes our paper and outlines future research directions.

2 Related Work

Folding. The idea of folding the (input) space has been investigated, among others, in computational geometry (Demaine et al., 2000). In context of neural networks, Montúfar et al. (2014) in Section 2.4 argued that each hidden layer in a ReLU neural network acts as a folding operator, recursively collapsing input-space regions. In Phuong & Lampert (2020), in the Appendix A.2 the authors defined the folds by ReLU networks, but left the exploration quite early on. Lewandowski et al. (2025) proposed the first measure to quantify the folding by ReLU neural networks. Our approach builds on the proposition therein, and is further motivated by the observation that folding gives rise to symmetries as discussed below.

Symmetries. The modern study of symmetries (in physics) was initiated by Noether (1916), who linked them to *conservation* laws: energy to time translation, momentum to space translation, and angular momentum to rotational symmetry. In the context of machine learning, researchers working with object recognition emphasised the importance of learning representations that are *invariant* to transformations, e.g., (Krizhevsky et al., 2012). Somewhat implicitly, symmetries have been at the core of some of the most successful deep neural network architectures, e.g., CNNs (Fukushima, 1980; LeCun et al., 1989) are equivariant to translation invariance characteristic of image classification tasks, while GNNs (Battaglia et al., 2018) are equivariant to the full group of permutations (see Higgins et al. (2022) for a detailed overview). Our work analyzes symmetries (reflection groups) that arise by space folding and their impact on the generalization capacity of the model.

Linear Regions Sampling. Analyzing neural network linear regions is challenging. Early work bounded their number as a measure of expressivity in ReLU MLPs (Montúfar et al., 2014; Raghu et al., 2017; Serra et al., 2018; Montúfar et al., 2021), later extending to CNNs (Xiong et al., 2020) and GNNs (Chen et al., 2022). Empirical studies indicate that linear regions are denser near training data (Zhang & Wu, 2020), yet standard sampling methods (e.g., Monte Carlo or Sobol sequences (Sobol, 1967)) often miss small regions. Goujon et al. (2024) showed that along one-dimensional paths, nonlinearity points scale linearly with depth, width, and activation complexity, while Gamba et al. (2022) proposed a direction-based method that requires costly minimal step calculations. In contrast, we introduce a sampling strategy in the Hamming activation space to efficiently identify linear regions along $\mathbf{d} = \mathbf{x}_2 - \mathbf{x}_1$.

3 Preliminaries

We define a *ReLU neural network* $\mathcal{N} : \mathcal{X} \to \mathcal{Y}$ with the total number of N neurons as an alternating composition of the ReLU function $\sigma(x) := \max(x, 0)$ applied element-wise on the input x, and

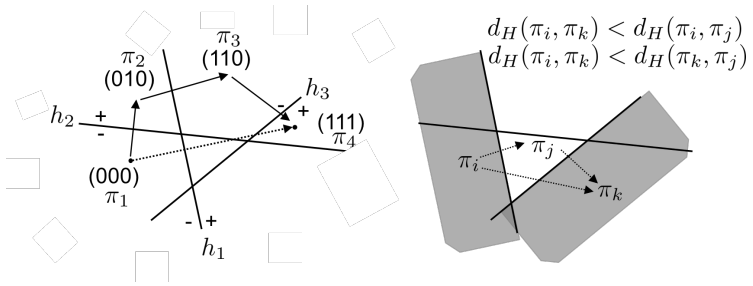

Figure 1: **Left:** Illustration of a walk on a straight path in the Euclidean input space and the Hamming activation space. The dotted line represent the shortest path in the Euclidean space. The arrows represent *a* shortest path in the Hamming distance between activation patterns π_1 and π_4 (in the Hamming space the shortest path is not unique). **Right:** Symmetry in the activation space: gray regions are closer to each other in the Hamming distance than to the region π_j that lies between them.

affine functions with weights W_k and biases b_k at layer k. An input $x \in \mathcal{X}$ propagated through $\mathcal{N}$ generates non-negative activation values on each neuron. A *binarization* is a mapping $\pi : \mathbb{R}^N \to \{0,1\}^N$ applied to a vector $v = (v_1, \ldots, v_N) \in \mathbb{R}^N$, resulting in a binary vector by clipping strictly positive entries of v to 1, and non-positive entries to 0, that is $\pi(v_i) = 1$ if $v_i > 0$, and $\pi(v_i) = 0$ otherwise. In our case, the vector v is the concatenation of all neurons of all hidden layers, called an *activation pattern*, and it represents an element in a binary hypercube $\mathcal{H}^N := \{0,1\}^N$ where the dimensionality is equal to the number N of (hidden) neurons in network $\mathcal{N}$. A *linear region* is an element of a partition covering the input domain where the network behaves as an affine function (Fig. 1, left). The Hamming distance, $d_H(u, v) := |\{u_i \neq v_i \text{ for } i = 1, \ldots, N\}|$, measures the difference between $u, v \in \mathcal{H}^N$, and for binary vectors is equivalent to the L_1 norm between those vectors. Lastly, as we will deal with paths of activation patterns, we denote the operation of joining those paths with the operator $\oplus : \mathcal{H}^{k \cdot N} \times \mathcal{H}^{(n-k+1) \cdot N} \to \mathcal{H}^{n \cdot N}$ such that $\{\pi_1, \ldots, \pi_k\} \oplus \{\pi_k, \ldots, \pi_n\} = \{\pi_1, \ldots, \pi_k, \ldots, \pi_n\}$. The operation $\oplus$ is defined for connected paths, where the last activation pattern of one path matches the first activation pattern of the other.

4 Space Folding

4.1 Construction

Consider a straight line connecting two input points $\mathbf{x}_1, \mathbf{x}_2$ in the Euclidean input space. The intermediate points are realized by varying the parameter t in a convex combination $(1-t)\mathbf{x}_1 + t\mathbf{x}_2$. Due to practicality, Lewandowski et al. (2025) spaced the parameter t equidistantly on $[0, 1]$, creating n segments. Equal spacing, though easy and fast to implement, frequently results in sub-optimal choice of the intermediate points (we address this issue in Sec. 5). To obtain a walk through activation patterns, we map the straight line $[\mathbf{x}_1, \mathbf{x}_2]$ through a neural network $\mathcal{N}$ to a *path* $\Gamma := \{\pi_1, \ldots, \pi_n\} \in \mathcal{H}^{n \cdot N}$ in the Hamming activation space, where the intermediate activation patterns belong to a binary hypercube, $\pi_i \in \mathcal{H}^N$ for all $i \in \{1, \ldots, n\}$ (see Fig. 2). We consider a change in the Hamming distance with respect to the initial activation pattern π_1 at each step i, $\Delta_i := d_H(\pi_{i+1}, \pi_1) - d_H(\pi_i, \pi_1)$, and then look at the maximum of the cumulative change $\max_k \sum_{i=1}^{k} \Delta_i$ along the path Γ,

$$r_1(\Gamma) = \max_i \sum_{j=1}^{i} \Delta_j = \max_i d_H(\pi_i, \pi_1). \tag{1}$$

We further keep track of the total distance traveled on the hypercube when following the path,

$$r_2(\Gamma) = \sum_{i=1}^{n-1} d_H(\pi_i, \pi_{i+1}). \tag{2}$$

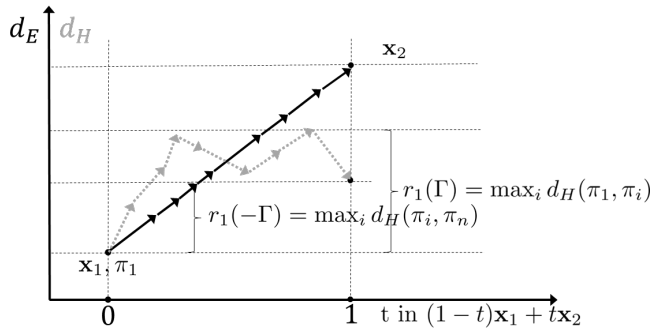

Figure 2: 1D straight walk from $\mathbf{x}_1$ to $\mathbf{x}_2$ in the Euclidean space (black full arrows) and the Hamming activation space (gray dotted arrows). Observe that in the Hamming activation space it might happen that $d_H(\pi_1, \pi_n) < \max_i d_H(\pi_1, \pi_i)$, which indicates space folding. The steps are optimized to visit each equivalence class exactly once (not equidistant).

For a measure of *space flatness*, we consider the ratio $r_1(\Gamma)/r_2(\Gamma)$. Equivalently, the *space folding* measure equals

$$\chi(\Gamma) := 1 - \max_i d_H(\pi_i, \pi_1) \Big/ \sum_{i=1}^{n-1} d_H(\pi_i, \pi_{i+1}). \tag{3}$$

The folding measure is lower and upper bounded, $\chi \in [0, 1]$; it equals 0 if there are no folds in the activation space, and it converges towards 1 for a path $\Gamma = \{\pi_1, \pi_2, \pi_1, \pi_2, \ldots\}$ looped between two activation regions such that $r_1(\Gamma) = d_H(\pi_1, \pi_2) = c \in \mathbb{R}^+$ and $r_2(\Gamma) \to \infty$. Although theoretically possible, this edge case example might be not realizable in practice.

4.2 Properties

In this section, we prove several properties of the folding measure, starting with emphasizing the importance of an appropriate sampling strategy. We show that taking multiple steps in the same activation region multiple times does not change the measure χ.

Proposition 4.1 (**Stability**). *Multiple steps in the same activation region do not influence the space folding measure* χ.

Proof. Consider a path $\{\pi_1, \ldots, \pi_i, \ldots, \pi_{i+l} \ldots, \pi_n\}$, where $\pi_i = \pi_{i+1} = \ldots = \pi_{i+l}$ and all other activation patterns are distinct. Observe that

$$\max_{j \in \{1,\ldots,i\}} d_H(\pi_1, \pi_j) = \max_{j \in \{1,\ldots,i+l\}} d_H(\pi_1, \pi_j)$$

and

$$\sum_{j=1}^{i-1} d_H(\pi_j, \pi_{j+1}) = \sum_{j=1}^{i+l-1} d_H(\pi_j, \pi_{j+1}),$$

thus traversing the same activation pattern more that once does not change the folding measure. □

Proposition 4.1 further highlights the importance of the sampling strategy that visits every activation pattern between samples. In Sec. 5, we propose a sampling algorithm that relies on the Hamming distances between samples in the Hamming activation space. We now show a necessary condition for space flatness, relevant for the upcoming analysis of the direction of folding.

Proposition 4.2 (**Flatness**). $\chi(\Gamma) = 0$ *if and only if* $d_H(\pi_1, \pi_i)$ *is non-decreasing for* $i = 1, \ldots, n$ *along the path* Γ.

Proof. We show that the space flatness implies that $d_H(\pi_1, \pi_i)$ is non-decreasing along the path Γ. Note that $\chi(\Gamma) = 0 \Rightarrow \max_j d_H(\pi_1, \pi_j) = \sum_{i=1}^{j-1} d_H(\pi_i, \pi_{i+1})$ for every index $j \in \{1, \ldots, n-1\}$. Let us argue through contradiction: Suppose that $d_H(\pi_1, \pi_i)$ decreases along the path Γ for some

index i^*, i.e., $d_H(\pi_1, \pi_{i^*}) < d_H(\pi_1, \pi_{i^*-1})$. This contradicts flatness, since $\sum_{i=1}^{i^*} d_H(\pi_1, \pi_i) > \sum_{i=1}^{i^*-1} d_H(\pi_1, \pi_i)$ while $\max_{j \leq i^*-1} d_H(\pi_1, \pi_j) = \max_{j \leq i^*} d_H(\pi_1, \pi_j)$, indicating folding of the space and thus finishing the proof. □

Proposition 4.2 implies that the folding occurs if r_1 (Eq. (1)) decreases at least once along the path. In the next step, we show that the folding measure is neither sub- nor super-additive, i.e., it neither holds that $\chi(\Gamma_1) \oplus \chi(\Gamma_2) \leq \chi(\Gamma_1 \oplus \Gamma_2)$ nor $\chi(\Gamma_1) \oplus \chi(\Gamma_2) \geq \chi(\Gamma_1 \oplus \Gamma_2)$ for connected paths Γ_1, Γ_2. Indeed, consider $\Gamma_1 = \{\pi_1, \pi_2\}$ and $\Gamma_2 = \{\pi_2, \pi_3, \pi_4\}$. A counter example for the sub-additivity is a path traversing the activation regions defined as

$$\pi_1 = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}, \pi_2 = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, \pi_3 = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}, \pi_4 = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}, \tag{4}$$

where $\chi(\Gamma_1 \oplus \Gamma_2) = \frac{1}{4}$ and $\chi(\Gamma_1) + \chi(\Gamma_2) = 0 + \frac{1}{3} = \frac{1}{3}$, thus $\chi(\Gamma_1) + \chi(\Gamma_2) \geq \chi(\Gamma_1 \oplus \Gamma_2)$ (for connected paths Γ_1 and Γ_2). To see that we can also construct a counter example for super-additivity, consider paths as previously with the activation patterns defined as

$$\pi_1 = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}, \pi_2 = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}, \pi_3 = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, \pi_4 = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \tag{5}$$

Then, $\chi(\Gamma_1 \oplus \Gamma_2) = \frac{4}{7}$ while $\chi(\Gamma_1) + \chi(\Gamma_2) = 0 + \frac{1}{2} = \frac{1}{2}$, thus $\chi(\Gamma_1) + \chi(\Gamma_2) \leq \chi(\Gamma_1 \oplus \Gamma_2)$.

While nor super- nor sub-additivity hold for every path Γ, in our experiments we have only observed sub-additivity of the folding measure. The counterexample for super-additivity (Eq. (5)), seems to be a rare occurrence in trained networks, though can be observed in specially constructed examples (see CantorNet by Lewandowski et al. (2024)). The general lack of super- or sub-additivity, but empirical sub-additivity motivates us to introduce the deviation from additivity for two paths Γ_1 and Γ_2 as $\Delta : \mathcal{H}^{n_1 \cdot N} \times \mathcal{H}^{n_2 \cdot N} \to [0, 1]$, where

$$\mathcal{I}(\Gamma_1, \Gamma_2) := |\chi(\Gamma_1 \oplus \Gamma_2) - \chi(\Gamma_1) - \chi(\Gamma_2)|. \tag{6}$$

4.3 On the Directnedness of Folding

So far, we have elaborated on the properties of the folding measure χ given by Eq. (3). We note that a path $\Gamma = \{\pi_1, \ldots, \pi_n\}$ along which the measure is computed is *directed*, i.e., the measure χ computed along the reversed path, $-\Gamma := \{\pi_n, \ldots, \pi_1\}$, may reach different folding values than $\chi(\Gamma)$, which we phrase as Remark 4.3.

Remark 4.3 (**Asymmetry**). Consider $\Gamma = \{\pi_1, \pi_2, \pi_3\}$, where $\pi_1 = (000), \pi_2 = (111), \pi_3 = (001)$, and its reverse $-\Gamma = \{\pi_3, \pi_2, \pi_1\}$. Then, $r_2(\Gamma) = r_2(-\Gamma)$ but $r_1(\Gamma) = 3$ and $r_1(-\Gamma) = 2$, thus $\chi(\Gamma) \neq \chi(-\Gamma)$.

It can be shown that, while folding is direction-sensitive, flatness is direction-invariant, expressed as Corollary 4.4.

Corollary 4.4 (**Flatness Invariance**). $\chi(\Gamma) = 0$ *if and only if* $\chi(-\Gamma) = 0$ *for a path* $\Gamma = \{\pi_1, \ldots, \pi_n\}$.

Proof. Observe that it is sufficient to prove Corollary 4.4 only in way direction as we can re-index the path Γ to obtain its reverse. We use Proposition 4.2: if $\chi(\Gamma) = 0$, then $d_H(\pi_1, \pi_i)$ is non-decreasing, what also implies that along the reversed path Γ the Hamming distance $d_H(\pi_n, \pi_i)$ is non-decreasing, indicating that $\chi(-\Gamma) = 0$. □

4.4 Global Space Folding Measure

We now adapt a new, global measure of folding. Consider a classification problem with classes $C = \{1, \ldots, L\}$. Suppose that we have computed the folding measure for every pair of samples between classes C_i and C_j by pairwise computation and taking the median of non-zero values

$\chi_+(C_i, C_j)$. We propose the average of inter-class[1] folding values, i.e.,

$$\Phi_{\mathcal{N}} := \frac{1}{(L-1)L} \sum_{C_i \neq C_j} \chi_+(C_i, C_j) \in [0, 1] \tag{7}$$

as the global folding measure. We posit that the global folding as characterized by Eq. (7) is the same if computed between every pair of samples regardless of their class assignment. Remark that, for a dataset with as little as 10^4 data points (e.g., MNIST test set), computing folding values pairwise would require $10^4!$ computations, which is computationally prohibitive even for small networks. As it has been observed in (Lewandowski et al., 2025) that the folding values in smaller networks (totaling 60 hidden neurons) and in larger architectures (totaling 600 neurons) remain approximately the same for a fixed number of layers (if the networks are trained to a low generalization error), we posit that, for classification problems, the global folding is a *feature* of the neural architecture $\mathcal{N}$ for which it has been computed, i.e.,

$$\Phi_{\mathcal{N}} \xrightarrow[\text{no. neurons}\to\infty]{\text{no. samples}\to\infty} \text{const}(\mathcal{N}). \tag{8}$$

The constant values of folding with increasing size of the network has yet another consequence. It means that, although there is an increasing number of linear regions as indicated by the works which provide bounds on this number, e.g., (Montúfar et al., 2014; Raghu et al., 2017; Serra et al., 2018; Hanin & Rolnick, 2019), the networks fold the space in a very similar manner if the generalization error is low, which we exploit in Section 6.

4.5 Beyond ReLU

Thus far we have proven several properties of the folding measure χ and provided additional interpretations. In this section, we interpret a walk through activation regions in ReLU-based MLP as a walk traversing distinct equivalence classes, and then show how this extends to *any* activation function. This makes our study directly applicable to vast range of activation functions, such as Swish (Ramachandran et al., 2018), GELU (Hendrycks & Gimpel, 2016) or SwiGLU (Shazeer, 2020). We start by defining the input equivalence relationship for ReLU neural networks.

Definition 4.5. We define the equivalence relation between two inputs $\mathbf{x}_1, \mathbf{x}_2$ with respect to a neural network $\mathcal{N}$ as

$$\mathbf{x}_1 \sim_{\mathcal{N}} \mathbf{x}_2 \iff d_H\left(\pi(\mathbf{x}_1), \pi(\mathbf{x}_2)\right) = 0$$

For ReLU neural networks the equivalence class $[\mathbf{x}_1]_{\mathcal{N}} := \{\mathbf{z} \in \mathbb{R}^m \mid \mathbf{z} \sim_{\mathcal{N}} \mathbf{x}_1\}$ corresponds to a linear region which contains point $\mathbf{x}_1$. We now show that the relation in Def. 4.5 is that of equivalence. Indeed, *reflexivity* holds as $\mathbf{x} \sim \mathbf{x} \Rightarrow \pi(\mathbf{x}) = \pi(\mathbf{x}) \Rightarrow d_H(\pi(\mathbf{x}), \pi(\mathbf{x})) = 0$, and vice-versa, $d_H(\pi(\mathbf{z}), \pi(\mathbf{x})) = 0$ holds for all $\mathbf{z}$ such that $\mathbf{z} \in [\mathbf{x}]_{\mathcal{N}}$, which also contains $\mathbf{x}$. *Symmetry* is straightforward to check, and *transitivity* holds as $\mathbf{x} \sim \mathbf{y}$ and $\mathbf{y} \sim \mathbf{z}$ implies that $d_H(\pi(\mathbf{x}), \pi(\mathbf{y})) = 0$ and $d_H(\pi(\mathbf{y}), \pi(\mathbf{z})) = 0$ thus also $d_H(\pi(\mathbf{x}), \pi(\mathbf{z})) = 0$, and inversely, 0 Hamming distances between $\pi(\mathbf{x})$ and $\pi(\mathbf{y})$ as well as $\pi(\mathbf{y})$ and $\pi(\mathbf{z})$ imply that $\mathbf{z} \in [\mathbf{x}]_{\mathcal{N}}$. In the following, we will extend the above results to a richer class of activation functions. While it is possible, we lose the geometrical interpretation of equivalence classes as "linear regions". Henceforth for the computation of the folding measure $\Phi_{\mathcal{N}}$, we consider a walk through input equivalence classes, not linear regions, thus extending the applicability of the space folding measure to much wider class of neural architectures. In order to obtain binary activation vectors, we threshold the values on intermediate layers (after applying the activation function) in a similar way as with the ReLU function, i.e., for a vector of activation values $\mathbf{a} \in \mathbb{R}^n$ we create an *activation pattern* by only considering strictly positive vs. non-positive activation values, and denoting them with 1 and 0, respectively. For monotonic activation functions, we obtain a disjoint partition of the input space, thus the equivalence relationship as defined in Def. 4.5 holds.

[1] We used the Mann-Whitney test (Mann & Whitney, 1947) to compare intra- and inter-class median folding values in networks with low generalization error. A statistically significant difference (per thresholds in Cohen (1992)) showed that inter-class folding values are higher, suggesting that the network folds space within each digit class for more efficient representation, thus justifying their separate analysis.

5 SAMPLING STRATEGY

In Proposition 4.1, we have shown the importance of an appropriate sampling for numerically computing the folding measure efficiently (see also Fig. 3). In this section, we introduce a 1D sampling strategy in the Hamming activation space, presented in Algorithm 1. Our algorithm is based on the Hamming distance between activation patterns along the path. Our method is parameter-free, straightforward to implement and intuitive to understand. It is based on the following intuition. Starting from $\mathbf{x}_1$, we move incrementally towards $\mathbf{x}_2$, generating the next point $\mathbf{x}_{\text{next}}$. Initially, we take a step of length Δ_{init}, compute the activation pattern under the network $\mathcal{N}$, and measure the Hamming distance $d_H(\pi_1, \pi_{\text{next}})$. If $d_H(\pi_1, \pi_{\text{next}}) = 0$, we proceed without storing π_{next}; if $d_H(\pi_1, \pi_{\text{next}}) = 1$, we store π_{next}. Lastly, if $d_H(\pi_1, \pi_{\text{next}}) > 1$, we iteratively reduce Δ until either $d_H(\pi_1, \pi_{\text{next}}) = 1$ or Δ reaches Δ_{min}. In the latter case, we accept π_{next} and continue moving toward $\mathbf{x}_2$. We note that adjacent activation regions may have the Hamming distance exceeding 1 – the algorithm will work unaffected, but this issue highlights the importance of the choice of the minimal step size Δ_{min}.

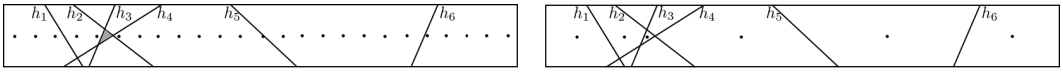

Figure 3: 2D slice of the ReLU tessellation defined by hyperplanes $h_1, \ldots, h_6$ highlights the need for optimal sampling. **Left:** Equally spaced points may revisit regions and miss small ones (gray). **Right:** The optimized path visits each region exactly once.

input points $\mathbf{x}_1, \mathbf{x}_2$; network $\mathcal{N}$; initial step size Δ_{init}; minimal step size Δ_{min}
output $\mathcal{P}$: activation patterns between $\mathbf{x}_1$ and $\mathbf{x}_2$
$\Delta t \leftarrow \Delta_{\text{init}}$
$\pi_{\text{prev}} \leftarrow \text{GetActivationPattern}(\mathbf{x}_1)$
$\mathcal{P} \leftarrow \{\pi_{\text{prev}}\}$
while $t < 1$ **do**
  $t_{\text{next}} \leftarrow \min(t + \Delta t,\ 1)$;
  $x_{\text{next}} \leftarrow \mathbf{x}_1 + t_{\text{next}}(\mathbf{x}_2 - \mathbf{x}_1)$;
  $\pi_{\text{next}} \leftarrow \text{GetActivationPattern}(\mathbf{x}_{\text{next}})$;
  if $d_H(\pi_{\text{prev}}, \pi_{\text{next}}) = 1$ **then**
    $t, \pi_{\text{next}}, \mathcal{P} \leftarrow \text{UpdateParams}(t, \pi_{\text{prev}}, \mathcal{P})$
  else if $d_H(\pi_{\text{prev}}, \pi_{\text{next}}) > 1$ **then**
    if $\Delta t \leq \Delta_{\text{min}}$ **then**
      $t, \pi_{\text{next}}, \mathcal{P} \leftarrow \text{UpdateParams}(t, \pi_{\text{prev}}, \mathcal{P})$
    else
      $\Delta t \leftarrow \Delta t/2$;
    end if
  else
    $t \leftarrow t_{\text{next}}$
  end if
end while

Algorithm 1: Sampling Strategy

Complexity Analysis. Let M be the total number of steps actually taken (including refined steps), $O(\mathtt{C})$ be the cost of running the network in the inference mode. Hence, the total computational cost is $O(M \cdot \mathtt{C})$. In the worst case, if many boundaries are crossed in very small intervals, the step size keeps halving, leading to a potentially large M. Each halving leads to a geometric progression, resulting in $\log(\Delta_{\text{init}}/\Delta_{\text{min}})$ refinement steps in some regions. In a typical scenario, M might be on the order of a few hundred. In a pathological scenario, it can grow larger but is still upper-bounded by repeated halving.

6 FOLDING AS A REGULARIZATION STRATEGY

Thus far, Lewandowski et al. (2025) observed that increased space folding values correlate positively with the generalization capabilities of ReLU-based MLP, motivating its use as a regularization dur-

ing the training process. One approach is to modify the loss function as follows:

$$Loss \leftarrow Loss + \lambda \frac{1}{(\Phi_{\mathcal{N}} + 1)^2}.$$

This formulation takes advantage of the fact that $\Phi_{\mathcal{N}} \in [0, 1]$: During the early stages of the training when $\Phi_{\mathcal{N}}$ is low, the regularization effect is strong; as the network learns and $\Phi_{\mathcal{N}}$ increases, the influence diminishes. To further encourage early folding, we use $(\Phi_{\mathcal{N}} + 1)^2$. To incorporate $\Phi_{\mathcal{N}}$ into a gradient-based learning algorithm, we replace the non-differentiable maximum function in r_1 with a smooth approximation using the log-sum-exp function. For a temperature parameter $\beta > 0$, define

$$\widetilde{r_1}(\Gamma) = \frac{1}{\beta} \log \left(\sum_{i=1}^{n} \exp \left(\beta d_H \left(\pi_1, \pi_i \right) \right) \right)$$

As β increases, $\widetilde{r_1}(\Gamma)$ approaches the true maximum; for finite β, the function is smooth and differentiable. The resulting folding value can be incorporated into the loss function and optimized using backpropagation. The regularization procedure generalizes to any activation function through equivalence classes (not limited to ReLU-based MLP, see Sec. 4.5).

7 Final Remarks

Future Work. We outline several directions to pursue in relation to the folding measure: (*i*) The proposed regularization scheme in Section 6 remains untested – in its current form, it requires multiple stops during training (every n epochs), suggesting opportunities for further optimization; (*ii*) While we have extended the space folding measure beyond the ReLU activation function, its behavior in other neural architectures necessitates further investigation; (*iii*) Lastly, although we have defined the interaction effect (Eq. (6)), we have not used it in empirical evaluations. We intend to do so in context the of adversarial attacks.

Summary. Our study deepens the mathematical understanding of the space folding measure and lays the groundwork for further experimental work. We have extended the applicability of the space folding measure to any activation function, highlighted key theoretical properties, and suggested its potential as a regularization technique.

References

Peter W Battaglia, Jessica B Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinicius Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, et al. Relational inductive biases, deep learning, and graph networks. *arXiv preprint arXiv:1806.01261*, 2018.

Hao Chen, Yu Guang Wang, and Huan Xiong. Lower and upper bounds for numbers of linear regions of graph convolutional networks. *arXiv preprint arXiv:2206.00228*, 2022.

Jacob Cohen. A power primer. *Psychological Bulletin*, 1992.

Erik D. Demaine, Martin L. Demaine, and Anna Lubiw. Folding and cutting paper. *Discrete and Computational Geometry*, 2000.

Ken A. Dill, S. Banu Ozkan, M. Scott Shell, and Thomas R. Weikl. The protein folding problem. *Annual Review of Biophysics*, 2008.

Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 1980.

Matteo Gamba, Adrian Chmielewski-Anders, Josephine Sullivan, Hossein Azizpour, and Marten Bjorkman. Are all linear regions created equal? *AISTATS*, 2022.

Alexis Goujon, Arian Etemadi, and Michael Unser. On the number of regions of piecewise linear neural networks. *Journal of Computational and Applied Mathematics*, 441:115667, 2024.

Boris Hanin and David Rolnick. Deep relu networks have surprisingly few activation patterns. In *NeurIPS*, 2019.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

Irina Higgins, Sébastien Racanière, and Danilo Rezende. Symmetry-based representations for artificial and biological general intelligence. *Frontiers in Computational Neuroscience*, 2022.

J Jumper et al. Highly accurate protein structure prediction with alphafold. *Nature*, 2021.

Christian Keup and Moritz Helias. Origami in n dimensions: How feed-forward networks manufacture linear separability. *arXiv preprint arXiv:2203.11355*, 2022.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *NeurIPS*, 2012.

Yann LeCun, Bernhard E Boser, John S Denker, et al. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1989.

Michal Lewandowski, Hamid Eghbal-zadeh, and Bernhard A.Moser. Cantornet: A sandbox for testing topological and geometrical measures. *NeurIPS W*, 2024.

Michal Lewandowski, Hamid Eghbalzadeh, Bernhard Heinzl, Raphael Pisoni, and Bernhard A.Moser. On space folds of relu neural networks. *TMLR*, 2025.

J. Makhoul, R. Schwartz, and A. El-Jaroudi. Classification capabilities of two-layer neural nets. *International Conference on Acoustics, Speech, and Signal Processing*, 1989.

H. B. Mann and D. R. Whitney. On a Test of Whether one of Two Random Variables is Stochastically Larger than the Other. *The Annals of Mathematical Statistics*, 18(1):50 – 60, 1947.

Guido Montúfar, Yue Ren, and Leon Zhang. Sharp bounds for the number of regions of maxout networks and vertices of minkowski sums. *arXiv preprint arXiv:2104.08135*, 2021.

Guido F Montúfar, Razvan Pascanu, Kyunghyun Cho, and Yoshua Bengio. On the number of linear regions of deep neural networks. *NeurIPS*, 2014.

E. Noether. Der endlichkeitssatz der invarianten endlicher gruppen. *Mathematische Annalen*, 1916.

Mary Phuong and Christoph H. Lampert. Functional vs. parametric equivalence of relu networks. *ICLR*, 2020.

Maithra Raghu, Ben Poole, Jon Kleinberg, Surya Ganguli, and Jascha Sohl-Dickstein. On the expressive power of deep neural networks. *ICML*, 2017.

Prajit Ramachandran, Barret Zoph, and Quoc V. Le. Searching for activation functions. *ICLR*, 2018.

Thiago Serra, Christian Tjandraatmadja, and Srikumar Ramalingam. Bounding and counting linear regions of deep neural networks. *ICML*, 2018.

Noam Shazeer. Glu variants improve transformer. *arXiv preprint arXiv:2002.05202*, 2020.

Ilya M. Sobol. On the distribution of points in a cube and the approximate evaluation of integrals. *Zhurnal Vychislitel'noi Matematiki i Matematicheskoi Fiziki*, 1967.

Huan Xiong, Lei Huang, Mengyang Yu, Li Liu, Fan Zhu, and Ling Shao. On the number of linear regions of convolutional neural networks. *ICML*, 2020.

Xiao Zhang and Dongrui Wu. Empirical studies on the properties of linear regions in deep neural networks. *ICLR*, 2020.