

Evaluating Training in Binarized Neural Networks Through the Lens of Algorithmic Information Theory

Eduardo Y. Sakabe¹
eyujis@gmail.com

Felipe S. Abrahão^{2,3,4,5}
felipesabrahao@gmail.com

Alexandre Simões⁶
alexandre.simoes@unesp.br

Esther Colombini⁷
esther@ic.unicamp.br

Paula Costa¹
paulad@unicamp.br

Ricardo Gudwin¹
gudwin@unicamp.br

Hector Zenil^{3,4,8,9}
hector.zenil@kcl.ac.uk

¹School of Electrical and Computer Engineering, University of Campinas (UNICAMP), Brazil

²Centre for Logic, Epistemology and the History of Science, University of Campinas (UNICAMP), Brazil

³Oxford Immune Algorithmics, Oxford University Innovation & London Institute for Healthcare Engineering, U.K.

⁴Algorithmic Dynamics Lab, Karolinska Institute & King's College London, U.K.

⁵DEXL, National Laboratory for Scientific Computing (LNCC), Brazil

⁶Department of Control and Automation Engineering, São Paulo State University (UNESP), Brazil

⁷Institute of Computing, University of Campinas (UNICAMP), Brazil

⁸Research Departments of Biomedical Computing and Digital Twins, School of Biomedical Engineering and Imaging Sciences

⁹King's Institute for Artificial Intelligence, King's College London, U.K.

Abstract

Understanding and controlling the informational complexity of neural networks is a central challenge in machine learning, with implications for generalization, optimization, and model capacity. While most approaches rely on entropy-based loss functions and statistical metrics, these measures often fail to capture deeper, causally relevant algorithmic regularities embedded in network structure. We propose a shift toward algorithmic information theory, using Binarized Neural Networks (BNNs) as a first proxy. Grounded in algorithmic probability (AP) and the universal distribution it defines, our approach characterizes learning dynamics through a formal, causally grounded lens. We apply the Block Decomposition Method (BDM)—a scalable approximation of algorithmic complexity based on AP—and demonstrate that it more closely tracks structural changes during training than entropy, consistently exhibiting stronger correlations with training loss across varying model sizes and randomized training runs. These results support the view of training as a process of algorithmic compression, where learning corresponds to the progressive internalization of structured regularities. In doing so, our work offers a principled estimate of learning progression and suggests a framework for complexity-aware learning and regularization, grounded in first principles from information theory, complexity, and computability.

1 Introduction

Understanding the distributional structure of neural network weights from an information-theoretic perspective has driven a range of advances in training efficiency, model compression, and architecture optimization. For example, [31] proposed an entropy-based criterion to dynamically adjust the number of hidden neurons, relating increases in weight entropy to growing representational demands. Similarly, [32] used the stabilization of weight entropy as a stopping criterion during training. Other approaches incorporate entropy directly into the loss function to regularize complexity and encourage more compact representations [13, 22, 23, 37]. Complementarily, post-training compression techniques leverage entropy coding and quantization to reduce model size with minimal impact on accuracy [5, 9, 24, 38, 39].

From the universal (algorithmic) coding theorem (see Section 2) within the context of algorithmic information theory (AIT) [3, 4, 7, 19], these approaches are often grounded in algorithmic probability [6, 10, 42, 45] and universal (Solomonoff) induction [15, 19], such as the minimum description length (MDL) principle [19, 26], which posits that the best model is the one that minimizes the total length of two descriptions: the model itself (its parameters or structure) and the data given the model (how well it fits the data). Shannon entropy is widely used in this context because it quantifies the expected bit-length required to encode outcomes from a probabilistic source [28], directly aligning with MDL. In the case of neural networks, weight entropy serves as a proxy for model complexity, estimating the information needed to represent the network’s parameters. Yet, entropy—while widely adopted—captures only statistical variability, overlooking algorithmic and causal structure critical to understanding how neural networks internalize and compress information. AIT offers an encompassing perspective focused on formal-theoretic measures of complexity (particularly, *algorithmic (program-size) complexity* and *algorithmic probability*) rather than statistical ones, emphasizing the need to capture not just the statistical properties of the data, but also the generative structure (or process) underlying data. Universal induction has been regarded as a theoretical solution to Artificial General Intelligence (AGI), positing that the most intelligent systems are those capable of compressing and generalizing via the shortest explanatory programs [10, 21].

We build on this foundation using the Block Decomposition Method (BDM) [42, 44–46], a computable and resource effective approximation to the (semicomputable) algorithmic complexity values, to estimate its value in application to the (global) complexity of neural network weights. BDM offers a practical and computable approximation that captures both statistical (at global scales) and algorithmic regularities (at local scales), providing a complexity measure that is more granular and more robust to changes in computation models, programming languages, and feature selection characterization of (irreducible/incompressible) information content than entropy is [18, 40, 42]. Such an approach aligns more closely with the theoretical underpinnings of intelligence as algorithmic compression and model synthesis [10, 16, 45]. Because the most powerful implementation of BDM operates on binary representations (even when it can also deal with non-binary objects) and can deal with 2D objects such as weight matrices, we binarize network weights and constrain our experiments to Binarized Neural Networks (BNNs) [14], which enable its direct application.

In our main experiment, we trained binarized Multilayer Perceptrons (MLPs) [14] with varying hidden layer sizes on MNIST [17], evaluating the correlation between model complexity—measured via BDM and entropy—and training loss across 200 training runs per architecture. Our findings reveal that the Pearson and Spearman correlations between BDM and training loss were consistently higher than those between entropy and loss, suggesting that BDM may serve as a more effective indicator of model complexity and its relationship with training dynamics (see also further discussion in Section 5). This empirical result supports the broader theoretical claim that training in neural networks can be understood as a process of algorithmic compression [33]—where structure is extracted and encoded in the weights—mirroring the regression/prediction principles proposed e.g. by Solomonoff [10]. These insights highlight the limitations of entropy-based approaches and point to the need for complexity-aware learning principles rooted in algorithmic probability [40, 41], such as those introduced in [12], which applied BDM to guide learning on non-differentiable spaces.

While related work has primarily examined entropy in terms of layer outputs and mutual information, such as those investigating the bottleneck principle [2, 30, 35], our approach focuses instead on the complexity of the model itself—specifically the distribution and structure of its weights—rather than the dynamics of input-output mappings in activation values across layers. This distinction matters: weight-based complexity reflects the internal representational capacity and structural organization of

the model (which is the case we study in this work); whereas layer output-based measures pertain to how data is processed during inference. Our results suggest, and we argue, that understanding and quantifying such an intrinsic complexity of a model is essential not only for interpretability and regularization, but also for advancing theoretical and practical progress toward AGI [10, 21], where learning must reflect causal inference and universal compression rather than statistical fitting alone.

Code and scripts to reproduce our experiments will be made available upon acceptance.

2 Background

2.1 Basics concepts and results in algorithmic information theory

The (unconditional prefix) *algorithmic* (Solomonoff-Kolmogorov-Chaitin) *complexity* of a finite string x , denoted by $\mathbf{K}(x)$, is the length of the shortest program $x^* \in \mathbf{L}_{\mathbf{U}}$ such that $\mathbf{K}(x) = |x^*| = \min \{|z| \mid \mathbf{U}(z) = x\}$ and $\mathbf{U}(x^*) = x$, where $\mathbf{L}_{\mathbf{U}}$ denote any (prefix-free or self-delimiting) universal programming language running on a machine \mathbf{U} .

Let $\mathbf{P}_{\mathbf{U}}[x] = \sum_{\mathbf{U}(p)=x} 2^{-|p|}$ denote the *universal a priori probability* of an arbitrary event x which can be understood as the probability of randomly generating (by an i.i.d. stochastic process) a prefix-free (or self-delimiting) program that outputs x —in other words, the probability that event x occurs resulting from the outcome of at least one of all possible computable generative models, formal theories, computer programs, Turing machines, etc.

A computably enumerable (c.e.) semimeasure $\mathbf{m}(\cdot)$ is said to be *maximal* if, for any other computably enumerable semimeasure $\mu(\cdot)$ with domain defined for all possible encoded objects, where $\sum_{x \in \{0,1\}^*} \mu(x) \leq 1$, there is a constant $C > 0$ (which does not depend on x) such that, for every encoded object x , $\mathbf{m}(x) \geq C \mu(x)$.

From the *algorithmic coding theorem* [3, 4, 7, 19] (or universal coding theorem) we have that

$$\mathbf{K}(x) = -\log(\mathbf{P}_{\mathbf{U}}[x]) \pm \mathbf{O}(1) = -\log(\mathbf{m}(x)) \pm \mathbf{O}(1) \quad (1)$$

holds, where p denotes a program running on machine \mathbf{U} such that it returns $\mathbf{U}(p) = x$ as output, where $|p|$ is the length of the program p .

We call $2^{-\mathbf{K}(x)}$ the *algorithmic probability* (AP) of x .

2.2 Block Decomposition Method

The Block Decomposition Method (BDM) [42, 44] presents an estimator of algorithmic information redundancies defined by a decomposition of the object into parts for which one already has an algorithmic complexity estimation, obtained by means of, for example, the Coding Theorem Method (CTM) [6] based on Algorithmic Probability (AP) (see Section 2.1) and the related universal distribution [15] which takes into consideration all the statistical and algorithmic regularities and redundancies in data. By finding the smallest generating programs (or models), BDM extends the power of CTM by joining these programs together (in a coarse-graining manner) in order to offer a generative computational model of the object, so that one can always achieve tighter lower bounds on AP (or upper bounds on \mathbf{K}) by running CTM further. As mentioned in Section 2.1, AP gives an agnostic and invariant probability measure for a randomly generated explanation (e.g. a randomly generated computer program) to explain an object [15] or a (-n encodable) set of phenomena so that it is independent (up to a ‘small’ constant that has been proven to present a stable rate [18, 40]) for the chosen computation model, most prominently for low-complexity (or equivalently high algorithmically probable) objects. In addition, it demonstrated to be invariant for an arbitrarily chosen programming language, prior probability distribution, and formal theory in the asymptotic limit as the object size increases.

In general case for any encodable multidimensional object [25], the BDM of an object x is defined by

$$\text{BDM}(x, i, m) = \sum_{(r_j, n_j) \in P_i(x)} (\log(n_j) + K_m(r_j)), \quad (2)$$

where:

- the partition (to which one assigns the corresponding index i) is one of the ways to decompose the object x ;
- $P_i(x)$ is the set of pairs (r_j, n_j) obtained when decomposing the object x according to a partition i of contiguous parts r_j , each of which appears n_j times in such a partition (i.e., n_j is the multiplicity of r_j), that is, the number of exact repetitions;
- $K_m(r_j)$ is an approximation to $\mathbf{K}(r_j)$ and m is the index of the approximation method employed to calculate $K_m(r_j)$.

Equation (2) can be expressed for unidimensional objects but also to encodable multidimensional objects in general, such as non-binary strings and n -dimensional objects such as graphs, matrices, images, vectors and tensors [40, 42–44]. For example, for a bit string x , Equation (2) holds for a partition defined by the sequence of contiguous linear blocks (of length ≥ 1) whose concatenation reconstructs x .

From classical information theory, we have that

$$\mathbf{H}_i(X^{(i)}) = - \sum_{(r_j, n_j) \in P_i(x)} p(r_j) \log(p(r_j)) \quad (3)$$

is the *block* (Shannon) entropy \mathbf{H}_i of an i.i.d. source $X^{(i)}$ such that $p(r_j) \rightarrow \frac{n_j}{N_i}$ as $|x| \rightarrow \infty$, the random variable $X^{(i)}$ can assume values in the set $\{r_1, \dots, r_j, \dots, r_{|P_i(x)|}\}$, and $N_i = \left(\sum_{(r_j, n_j) \in P_i(x)} n_j \right)$.

Thus, \mathbf{H} is a basis for (statistical) compression methods that are subsumed into BDM while for sufficiently large objects both converge to each other. This is because BDM characterizes the information content of the entire object by adding the estimated (local) complexity given by \mathbf{K} and the (global) Entropy (\mathbf{H}) values as described in Equation (2) [25, 42].

Our results in this paper corroborate these mathematical properties of BDM and entropy, the former expected to perform better for smaller objects while being more sensitive to patterns other than statistical ones. In Section 4.6, our control experiment evinces the case in which both are indeed expected to converge.

3 Measuring the Complexity of Binarized Neural Networks

Our primary objective is to investigate whether algorithmic complexity, estimated via the Block Decomposition Method (BDM), serves as a more informative indicator of neural network learning dynamics than entropy. We hypothesize that training a neural network reduces the algorithmic complexity of its weights by aligning them with the structural regularities of the data. In this framing, learning is understood as a form of algorithmic compression: transforming initially random, high-complexity parameters into structured configurations that encode the input-output mappings required by the task.

Accordingly, we expect BDM—which captures local algorithmic regularities beyond statistical variability—to correlate more strongly with training loss than entropy does. While entropy quantifies the expected bit-length under a probabilistic model, it does not account for causal or generative structure within the weight matrix. In contrast, BDM, grounded in algorithmic information theory, approximates algorithmic complexity by detecting repeatable, low-complexity patterns, even in systems that may appear statistically random.

This hypothesis builds on the assumption that effective learning involves the internalization of data structure into the model’s parameters in a compact, structured form. We take the training loss as a proxy for this process, assuming that as the loss decreases, the network is increasingly aligned with the regularities in the training data. However, because low loss can also result from memorization rather than generalization, we constrain our analysis to the training regime before the onset of overfitting, as indicated by a plateau in validation loss—where the model is likely compressing the data’s functional structure rather than encoding idiosyncratic details of the training data.

Our approach is consistent with the Minimum Description Length (MDL) principle [26], and related perspectives in deep learning that frame training as a compression process [27, 35]. By directly comparing BDM and entropy under identical training conditions, we aim to evaluate whether BDM better captures meaningful structural transformations in the model’s parameters during learning.

This view of training as a form of algorithmic compression is supported by recent commentary by Sutskever [33], who suggests that gradient-based optimization—particularly Stochastic Gradient Descent (SGD)—can be seen as an implicit algorithmic search, uncovering compressed programs within the neural network’s weights.

3.1 Computing BDM and Entropy in Binarized Neural Networks

To assess the complexity of a fully connected binarized neural network during training, we compute two measures over its binarized weight matrices: algorithmic complexity using the Block Decomposition Method (BDM), and statistical complexity using entropy. Both measures are derived from a common decomposition of the weights into fixed-size binary submatrices.

Weight Extraction and Binarization: We extract all weight matrices from the model, excluding batch normalization layers. Each matrix is binarized by applying the sign function, mapping values > 0 to 1 and ≤ 0 to 0. This produces a set of 2D binary matrices, suitable for pattern-based analysis.

Shared Block Decomposition: Each binarized matrix is partitioned into non-overlapping 4×4 blocks. This yields a multiset of binary patterns used as atomic units for both BDM and entropy computation. For each matrix, we count the occurrences of each unique 4×4 block. These counts define an empirical distribution over the observed patterns.

Shannon Entropy: The entropy of a matrix is computed using the empirical distribution of 4×4 patterns as defined in Equation 2.2. Here, $p(r_j)$ corresponds to the relative frequency of pattern r_j across all blocks. This entropy captures the statistical variability of local structures in the network’s weights.

Block Decomposition Method (BDM): To estimate algorithmic complexity, we apply BDM as defined in Equation 2.2. Each unique 4×4 block r_j is assigned a complexity value based on the Coding Theorem Method (CTM), and repeated occurrences are penalized logarithmically. This yields a composite complexity score reflecting both diversity and compressibility of local patterns.

All computations, including block partitioning, empirical distribution estimation, entropy, and BDM values, were implemented using the pybdm library [34].

4 Experiments

We evaluated the relationship between model complexity and learning dynamics in binarized neural networks trained on MNIST. To assess the robustness of our findings, we also included a control experiment with random binary data and labels, described in Section 4.6.

4.1 Dataset

We used the MNIST dataset of handwritten digits [17], a standard image classification benchmark consisting of 28×28 grayscale images of digits from 0 to 9, where the task is to assign each image to its corresponding digit class. We resized each image to 10×10 pixels to reduce input dimensionality and avoid the domain where BDM approximates entropy. Pixel values were normalized using the dataset mean and standard deviation.

To evaluate generalization and monitor overfitting, we constructed a validation set of 10,000 examples, stratified by class, from the original 60,000-image training set. The remaining 50,000 examples formed a pool from which we generated 200 independent training subsets. Each subset consisted of a stratified random sample of 25,000 examples, drawn without replacement, preserving class proportions and ensuring disjointness from the validation set.

The standard MNIST test set (10,000 examples) was held out and used only for reporting final accuracy of the selected models.

4.2 Model Architecture

We used a binarized Multilayer Perceptron (MLP) [14] with two hidden layers, where both weights and activations were binarized using the Straight-Through Estimator (STE) [1], allowing backpropagation through discrete functions. The model processed resized 10×10 pixel MNIST images, with the number of neurons in the two hidden layers denoted as N_1 and N_2 . We applied batch normalization to each hidden layer to stabilize training.

We experimented with different hidden layer configurations, testing (N_1, N_2) pairs of (8, 4), (16, 8), (32, 16), (64, 32), and (128, 64).

4.3 Training Procedure

For each model configuration (N_1, N_2) , we trained 200 independent model instances, each on a different stratified subset of the training data. Training employed early stopping with a patience of 5 epochs based on validation loss: if the validation loss did not improve for 5 consecutive epochs, training was halted, and the model with the lowest validation loss was selected.

We optimized the cross-entropy loss using the Adam optimizer with a learning rate of 1×10^{-3} and a mini-batch size of 128.

To enable post hoc analysis of model complexity and entropy, we saved the model weights after every training step—that is, after each backpropagation update. BDM and entropy were computed at this resolution and later averaged per epoch to align with the training and validation loss: the former was averaged over batches, and the latter was computed once per epoch.

All experiments were run on a MacBook Pro with an M4 Max chip using PyTorch with Metal acceleration. Training 200 models for the largest architecture took about 5 hours, and BDM/entropy computations required an additional 2.5 hours.

4.4 Evaluation

We assessed the relationship between model complexity and learning dynamics by analyzing the correlations between mean training loss and mean complexity metrics (BDM and entropy) computed per epoch throughout training. Our evaluation followed three main steps:

Metric Normalization: To ensure comparability and reduce noise, we applied a normalization pipeline to the per-epoch series of training loss, BDM, and entropy values. We excluded the final five epochs prior to early stopping to avoid the overfitting regime. We applied log transformation, Gaussian smoothing, and MinMax scaling to the remaining values.

Correlation Analysis: Using the normalized values, we computed Pearson and Spearman correlation coefficients between training loss and each complexity metric. These correlations quantified both linear (Pearson) and monotonic (Spearman) relationships, providing insight into how closely each metric tracked the progression of learning.

Bootstrap Confidence Intervals: To assess statistical reliability, we estimated 95% confidence intervals via bootstrap resampling over the 200 independently trained models for each architecture. This provided robust estimates of variability for all reported correlations.

4.5 Results

The results of our experiments are summarized in Table 1, which presents 95% confidence intervals for Pearson and Spearman correlations between model complexity—measured using BDM and entropy—and the mean training loss across different model configurations. We also reported the final test accuracies of the selected models based on the lowest validation loss. The configurations of hidden layers were denoted by N_1, N_2 , where N_1 is the number of neurons in the first hidden layer and N_2 in the second.

Correlations were consistently stronger for BDM than entropy, particularly in smaller models, where the influence of algorithmic structure was more pronounced.

To ensure the reliability of the correlation estimates, we excluded runs that terminated before 10 epochs, allowing for a consistent 5-epoch window preceding early stopping. Consequently, the

Table 1: Correlation [95% CI] between training loss and complexity metrics across model sizes, with final test accuracy reported as mean \pm standard deviation. r and ρ denote Pearson and Spearman correlations, respectively. Bold values indicate the higher correlation in each pair. BDM consistently outperforms entropy across all models.

N_1, N_2	BDM r	Entropy r	BDM ρ	Entropy ρ	Accuracy (%)
8, 4	[0.72, 0.81]	[0.45, 0.53]	[0.59, 0.70]	[0.50, 0.62]	51.5 \pm 5.1
16, 8	[0.85, 0.89]	[0.55, 0.63]	[0.74, 0.81]	[0.55, 0.66]	67.8 \pm 2.3
32, 16	[0.90, 0.92]	[0.69, 0.74]	[0.73, 0.80]	[0.61, 0.70]	79.8 \pm 1.1
64, 32	[0.91, 0.92]	[0.82, 0.83]	[0.84, 0.88]	[0.79, 0.84]	85.9 \pm 0.5
128, 64	[0.91, 0.91]	[0.85, 0.87]	[0.90, 0.93]	[0.88, 0.91]	89.0 \pm 0.3

number of runs included in the bootstrap analysis was 175 for the (8, 4) model and 195 for the (16, 8) model, while all other configurations retained the full set of 200 runs. These sample sizes remained sufficient for robust statistical inference.

To complement the aggregate correlation results presented in Table 1, we visualized the evolution of BDM and entropy values alongside training and validation loss across training epochs for each model architecture in Figure 1a-e.

4.6 Control Experiment with Random Data

To assess whether the observed correlations between complexity and training loss reflected meaningful structure in the data, we conducted a control experiment using a synthetic dataset consisting of 10×10 random float inputs (sampled uniformly from $[0, 1)$) and randomly assigned class labels. This design ensured that any learning reflected memorization rather than compression of structured information.

We used the same (32, 16) architecture and training procedure as in the corresponding MNIST experiment, with 200 class-balanced subsets to control for label imbalance effects. We chose this configuration because it represented a mid-sized model where BDM consistently outperformed entropy. Unlike the main experiments, models were trained for a fixed duration of 15 epochs without early stopping, as the absence of shared structure between training and validation sets rendered validation loss uninformative for model selection or generalization; accordingly, it was not computed during training.

We computed correlations between training loss and both complexity measures across all runs to evaluate their behavior in the absence of learnable structure. Despite the lack of meaningful patterns, nonzero correlations may still emerge due to memorization. In this random-data setting, the 95% confidence intervals were: BDM — Pearson [0.72, 0.77], Spearman [0.43, 0.54]; Entropy — Pearson [0.76, 0.80], Spearman [0.46, 0.55]. The corresponding training dynamics are illustrated in Figure 1f. We discuss these results in the following section.

5 Discussion

We first verified that all models performed significantly above chance on MNIST. While a random classifier achieves roughly 10% accuracy, even the smallest architecture exceeded 50%, indicating that the networks have learned meaningful input-output mappings from the data.

The correlation results reveal strong positive relationships between model complexity and training loss for both BDM and entropy. These findings indicate that as models reduce error over time, their structural and statistical complexity also decrease. Across all configurations, BDM exhibits higher Pearson and Spearman correlation coefficients than entropy, suggesting that BDM is more sensitive to changes in the model throughout training, particularly in smaller architectures where algorithmic regularities are more distinct.

The higher Pearson correlations imply that BDM tracks the magnitude of changes in training loss more closely. Concurrently, the stronger Spearman correlations indicate that BDM better preserves the relative ordering of complexity over training epochs. Together, these results suggest that BDM provides a richer signal of learning progression than entropy, likely due to its foundation in algorithmic information theory, which captures more than just statistical regularities (see Section 2).

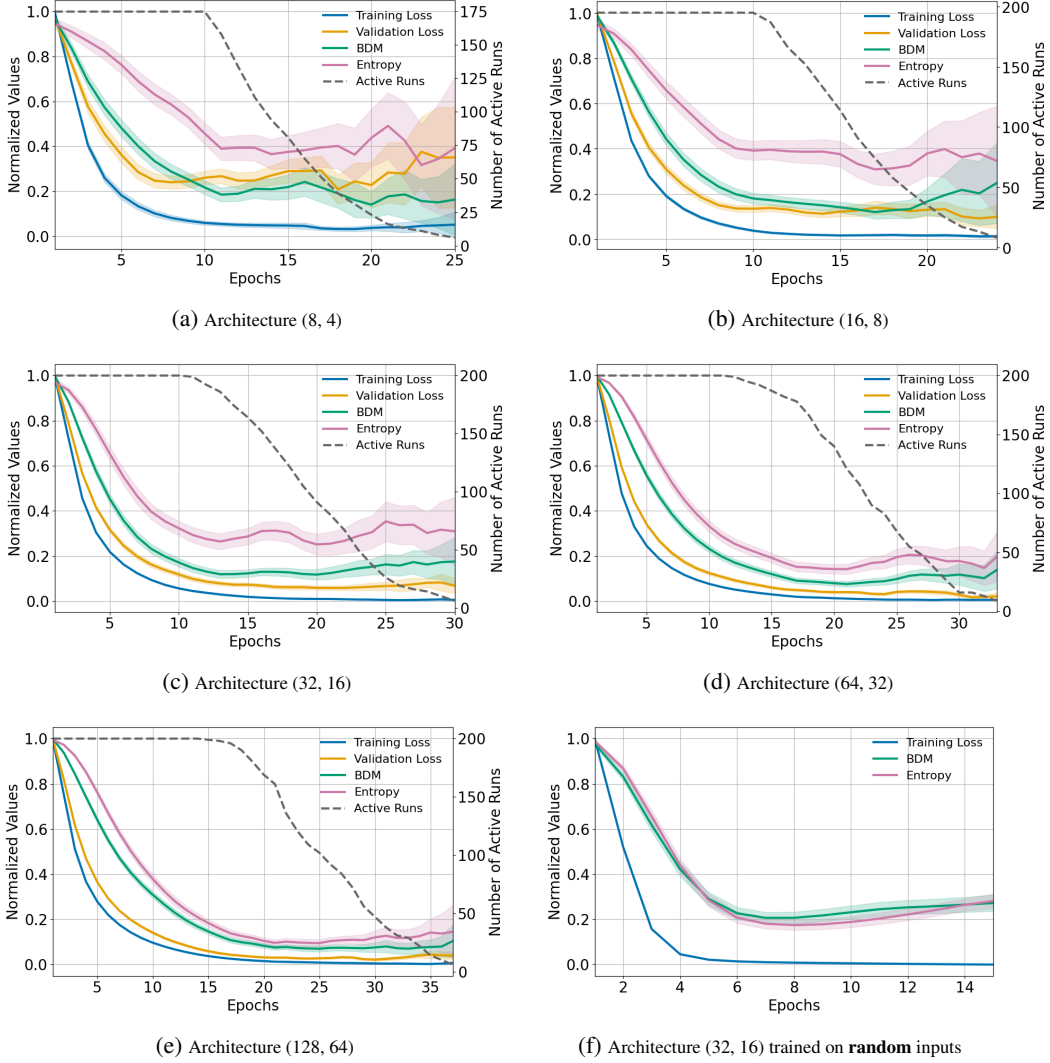


Figure 1: Evolution of training loss, validation loss, BDM, and entropy across epochs for each model architecture. Subplots (a)–(e) correspond to increasing model sizes trained on MNIST, while (f) shows the control condition using random inputs with the (32, 16) architecture. All metrics are normalized to enable direct comparison across runs. Shaded regions represent 95% confidence intervals, computed only for epochs with at least five active runs. The dashed gray line indicates the number of active runs per epoch, reflecting early stopping behavior. Across all architectures, BDM more closely follows the trajectory of training loss than entropy, particularly during early and mid-training. This alignment suggests that BDM is more sensitive to the structural changes induced by learning. BDM also exhibits lower variance across runs, providing more stable complexity estimates throughout training. In the random-data condition (f) from our control experiment (see Sections 4.6 and 5), where no meaningful structure is present, the distinction between BDM and entropy largely vanishes—reinforcing the interpretation that BDM’s advantage depends on its sensitivity to underlying algorithmic regularities.

However, this advantage diminishes as model size increases, since BDM relies on evaluating fixed-size 4×4 binary blocks via the CTM. As the size of the network grows, the decomposition process leads to increasing redundancy and a heavier influence of the multiplicity term $\log_2 n_i$ in Equation 2.2. This results in a loss of granularity and a convergence of BDM toward entropy-like behavior, reducing its ability to discriminate structural complexity. Thus, in larger models, BDM transitions from an algorithmic to a more statistical measure (see Section 2 and a discussion on limitations in Section 5.1).

The training dynamics shown in Figure 1a-e further support these findings. Across all architectures, BDM exhibits a trajectory that more closely follows the evolution of training loss compared to entropy. This temporal alignment reinforces the view that BDM is more responsive to the structural transformations that occur as the model learns. Moreover, although confidence intervals naturally widen toward later epochs due to early stopping and reduced sample sizes, entropy displays greater variability across runs at each epoch. This difference in variance suggests that BDM not only correlates more strongly with training loss but also produces more stable complexity estimates during training.

In the control experiment with random data, the model successfully minimized training loss over 15 epochs, demonstrating its capacity to memorize arbitrary inputs in the absence of learnable structure. Compared to the MNIST setting, BDM correlations with training loss were reduced, while entropy correlations remained relatively stable and slightly higher. This contrast reinforces the interpretation that BDM is specifically sensitive to compressible structure, whereas entropy continues to reflect general memorization.

In this unstructured condition, entropy exhibited slightly higher correlations than BDM across both Pearson and Spearman metrics. This reversal relative to the MNIST results suggests that, in the absence of algorithmic regularities, BDM converges toward statistical behavior and loses its advantage. As shown in Figure 1f, the trajectories of BDM and entropy are closely aligned, reflecting the lack of structural signals and supporting the view that BDM’s value emerges only in the presence of causal or generative patterns in the data.

5.1 Limitations

While algorithmic information theory provides a more principled and causally grounded foundation for characterizing neural network complexity, our approach has several limitations that constrain its broader applicability.

First, algorithmic complexity is inherently non-continuous and non-differentiable. This prevents its direct integration into gradient-based optimization algorithms such as backpropagation, making it unsuitable as a training-time regularizer in its current form. Second, the current implementation of the Block Decomposition Method (BDM) requires binary inputs in two-dimensional structures. Consequently, our analysis is restricted to Binarized Neural Networks (BNNs), which, while useful as a simplified model class, are less commonly used than full-precision architectures in practical applications. Third, for larger and more complex objects, BDM tends to converge toward entropy, as discussed earlier. This convergence reduces its sensitivity to deeper algorithmic structure in high-dimensional settings and limits its advantage over entropy-based measures in such cases.

Taken together, these limitations currently preclude the direct application of our method to large-scale, non-binarized models. Overcoming these constraints—either by developing differentiable approximations of algorithmic complexity or by extending BDM to richer data representations—remains an important direction for future work.

6 Conclusion

This work presents a principled investigation of neural network training through the lens of algorithmic information theory. By applying the Block Decomposition Method (BDM) to Binarized Neural Networks (BNNs), we demonstrated that algorithmic complexity offers a more sensitive and stable indicator of training dynamics than traditional entropy.

Empirical results across multiple architectures show that BDM correlates more strongly with training loss than entropy, particularly in smaller models, where algorithmic regularities are more pronounced. These findings offer direct empirical support for the view that training operates as a process of algorithmic compression, transforming random initializations into structured, compressible configurations that reflect the underlying data-generating process. Control experiments with random-input data further reinforce this interpretation: in the absence of meaningful structure, the advantage of BDM disappears, and its behavior converges toward that of entropy. These results confirm that BDM captures structural features intrinsic to learning, beyond distributional statistics.

Taken together, our results highlight the potential of algorithmic complexity measures to enrich our understanding of neural network behavior. They open new directions for future work, including the development of complexity-aware training regimes, regularization strategies based on algorithmic information theory, and the design of learning systems that exploit causal-compressibility as a guiding principle. This perspective is especially relevant in the context of emerging architectures characterized by localized computation—such as sparse neural networks [8], transformers [36], Mixture-of-Experts (MoE) models [29], and Kolmogorov–Arnold Networks (KANs) [20]. In these systems, BDM may have an even greater advantage, as it is particularly well-suited for characterizing modular structures [11].

Our work, alongside that of [12], begins to address a longstanding challenge: integrating algorithmic complexity and algorithmic probability—long proposed as a theoretical solution to AI through universal induction—into practical machine learning. Despite their foundational role in the theoretical foundations of artificial intelligence, these concepts have remained largely disconnected from neural network training. By operationalizing algorithmic complexity via BDM in binarized architectures, we take a step toward bridging this gap—replacing statistical proxies like entropy with causally grounded, algorithmic measures. In doing so, we contribute to realizing algorithmic theories of learning in practice, bringing foundational principles of AI closer to their application in modern machine learning.

7 Acknowledgements

This project was supported by the Ministry of Science, Technology, and Innovation of Brazil, with resources granted by the Federal Law 8.248 of October 23, 1991, under the PPI-Softex. The project was coordinated by Softex and published as Intelligent agents for mobile platforms based on Cognitive Architecture technology [01245.003479/2024-10]. This study was partially funded by the Coordenação de Aperfeiçoamento de Pessoal de Nivel Superior – Brasil (CAPES) – Finance Code 001. Felipe S. Abrahão acknowledges support from the São Paulo Research Foundation (FAPESP), grants 2021/14501-8.

References

- [1] Yoshua Bengio. Estimating or Propagating Gradients Through Stochastic Neurons, May 2013. URL <http://arxiv.org/abs/1305.2982>. arXiv:1305.2982 [cs].
- [2] Ivan Butakov, Alexander Tolmachev, Sofia Malanchuk, Anna Neopryatnaya, Alexey Frolov, and Kirill Andreev. Information Bottleneck Analysis of Deep Neural Networks via Lossy Compression, May 2024. URL <http://arxiv.org/abs/2305.08013>. arXiv:2305.08013 [cs].
- [3] Cristian S. Calude. *Information and Randomness: An algorithmic perspective*. Springer-Verlag, 2 edition, 2002. ISBN 3540434666.
- [4] Gregory Chaitin. *Algorithmic Information Theory*. Cambridge University Press, 3 edition, 2004. ISBN 0521616042.
- [5] Yoojin Choi, Mostafa El-Khamy, and Jungwon Lee. Towards the Limit of Network Quantization, November 2017. URL <http://arxiv.org/abs/1612.01543>. arXiv:1612.01543 [cs].
- [6] Jean Paul Delahaye and Hector Zenil. Numerical evaluation of algorithmic complexity for short strings: A glance into the innermost structure of randomness. *Applied Mathematics and Computation*, 219(1):63–77, sep 2012. ISSN 00963003. doi: 10.1016/j.amc.2011.10.006.
- [7] Rodney G. Downey and Denis R. Hirschfeldt. *Algorithmic Randomness and Complexity*. Theory and Applications of Computability. Springer New York, New York, NY, 2010. ISBN 978-0-387-95567-4. doi: 10.1007/978-0-387-68441-3. URL <http://link.springer.com/10.1007/978-0-387-68441-3>.
- [8] Jonathan Frankle and Michael Carbin. The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks, March 2019. URL <http://arxiv.org/abs/1803.03635>. arXiv:1803.03635 [cs].
- [9] Song Han, Huizi Mao, and William J. Dally. Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding, February 2016. URL <http://arxiv.org/abs/1510.00149>. arXiv:1510.00149 [cs].

- [10] Alberto Hernández-Espinosa, Luan Ozelim, Felipe S. Abrahão, and Hector Zenil. SuperARC: An Agnostic Test for Narrow, General, and Super Intelligence Based On the Principles of Recursive Compression and Algorithmic Probability, April 2025. URL <http://arxiv.org/abs/2503.16743>. arXiv:2503.16743 [cs].
- [11] Santiago Hernández-Orozco, Narsis A. Kiani, and Hector Zenil. Algorithmically probable mutations reproduce aspects of evolution, such as convergence rate, genetic memory and modularity. *Royal Society Open Science*, 5(8):180399, August 2018. doi: 10.1098/rsos.180399. URL <https://royalsocietypublishing.org/doi/10.1098/rsos.180399>. Publisher: Royal Society.
- [12] Santiago Hernández-Orozco, Hector Zenil, Jürgen Riedel, Adam Uccello, Narsis A. Kiani, and Jesper Tegnér. Algorithmic Probability-Guided Machine Learning on Non-Differentiable Spaces. *Frontiers in Artificial Intelligence*, 3, January 2021. ISSN 2624-8212. doi: 10.3389/frai.2020.567356. URL <https://www.frontiersin.org/journals/artificial-intelligence/articles/10.3389/frai.2020.567356/full>. Publisher: Frontiers.
- [13] Geoffrey E. Hinton and Drew van Camp. Keeping the neural networks simple by minimizing the description length of the weights. In *Proceedings of the sixth annual conference on Computational learning theory*, COLT '93, pages 5–13, New York, NY, USA, August 1993. Association for Computing Machinery. ISBN 978-0-89791-611-0. doi: 10.1145/168304.168306. URL <https://dl.acm.org/doi/10.1145/168304.168306>.
- [14] Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Binarized Neural Networks. In *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016. URL https://proceedings.neurips.cc/paper_files/paper/2016/hash/d8330f857a17c53d217014ee776bfd50-Abstract.html.
- [15] Walter Kirchherr, Ming Li, and Paul Vitányi. The miraculous universal distribution. *The Mathematical Intelligencer*, 19(4):7–15, 1997. ISSN 0343-6993. doi: 10.1007/BF03024407.
- [16] Alexander Lavin, David Krakauer, Hector Zenil, Justin Gottschlich, Tim Mattson, Johann Brehmer, Anima Anandkumar, Sanjay Choudry, Kamil Rocki, Atılım Güneş Baydin, Carina Prunkl, Brooks Paige, Alexandr Isayev, Erik Peterson, Peter L. McMahon, Jakob Macke, Kyle Cranmer, Jiaxin Zhang, Haruko Wainwright, Adi Hanuka, Manuela Veloso, Samuel Assefa, Stephan Zheng, and Avi Pfeffer. Simulation Intelligence: Towards a New Generation of Scientific Methods, November 2022. URL <http://arxiv.org/abs/2112.03235>. arXiv:2112.03235 [cs].
- [17] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, November 1998. ISSN 1558-2256. doi: 10.1109/5.726791. URL <https://ieeexplore.ieee.org/abstract/document/726791>.
- [18] Zoe Leyva-Acosta, Eduardo Acuña Yeomans, and Francisco Hernandez-Quiroz. An additively optimal interpreter for approximating Kolmogorov prefix complexity. *arXiv Preprints*, arXiv:2407.21162, 2024.
- [19] Ming Li and Paul Vitányi. *An Introduction to Kolmogorov Complexity and Its Applications*. Texts in Computer Science. Springer, Cham, 4 edition, 2019. ISBN 978-3-030-11298-1. doi: 10.1007/978-3-030-11298-1.
- [20] Ziming Liu, Yixuan Wang, Sachin Vaidya, Fabian Ruehle, James Halverson, Marin Soljačić, Thomas Y. Hou, and Max Tegmark. KAN: Kolmogorov-Arnold Networks, February 2025. URL <http://arxiv.org/abs/2404.19756>. arXiv:2404.19756 [cs].
- [21] Marvin Minsky and World Science Festival. The limits of understanding, December 2014. URL <https://www.youtube.com/watch?v=DfY-DRsE86s&t=5392s>. Marvin Minsky discusses the importance of algorithmic probability and universal induction in this panel discussion.
- [22] Dmitry Molchanov, Arsenii Ashukha, and Dmitry Vetrov. Variational Dropout Sparsifies Deep Neural Networks. In *Proceedings of the 34th International Conference on Machine Learning*, pages 2498–2507. PMLR, July 2017. URL <https://proceedings.mlr.press/v70/molchanov17a.html>. ISSN: 2640-3498.
- [23] Steven J. Nowlan and Geoffrey E. Hinton. Simplifying Neural Networks by Soft Weight-Sharing. *Neural Computation*, 4(4):473–493, July 1992. ISSN 0899-7667. doi: 10.1162/neco.1992.4.4.473. URL <https://ieeexplore.ieee.org/document/6796174>.
- [24] Deniz Oktay, Johannes Ballé, Saurabh Singh, and Abhinav Shrivastava. Scalable model compression by entropy penalized reparameterization. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL <https://openreview.net/forum?id=HkgxW0EYDS>.

- [25] Luan Ozelim, Abicumaran Uthamacumaran, Felipe S. Abrahão, Santiago Hernández-Orozco, Narsis A. Kiani, Jesper Tegnér, and Hector Zenil. Assembly Theory Reduced to Shannon Entropy and Rendered Redundant by Naive Statistical Algorithms, March 2025. URL <http://arxiv.org/abs/2408.15108>. arXiv:2408.15108 [cs].
- [26] Jorma Rissanen. Stochastic Complexity and Modeling. *The Annals of Statistics*, 14(3):1080–1100, 1986. ISSN 0090-5364. URL <https://www.jstor.org/stable/3035559>. Publisher: Institute of Mathematical Statistics.
- [27] Jürgen Schmidhuber. Discovering Neural Nets with Low Kolmogorov Complexity and High Generalization Capability. *Neural Networks*, 10(5):857–873, July 1997. ISSN 0893-6080. doi: 10.1016/S0893-6080(96)00127-X. URL <https://www.sciencedirect.com/science/article/pii/S089360809600127X>.
- [28] C. E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27(3):379–423, July 1948. ISSN 0005-8580. doi: 10.1002/j.1538-7305.1948.tb01338.x. URL <https://ieeexplore.ieee.org/abstract/document/6773024>.
- [29] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer, January 2017. URL <http://arxiv.org/abs/1701.06538>. arXiv:1701.06538 [cs].
- [30] Ravid Shwartz-Ziv and Naftali Tishby. Opening the Black Box of Deep Neural Networks via Information, April 2017. URL <http://arxiv.org/abs/1703.00810>. arXiv:1703.00810 [cs].
- [31] Seba Susan and Mayank Dwivedi. Dynamic Growth of Hidden-Layer Neurons Using the Non-extensive Entropy. In *2014 Fourth International Conference on Communication Systems and Network Technologies*, April 2014. doi: 10.1109/CSNT.2014.104. URL <https://ieeexplore.ieee.org/document/6821445>.
- [32] Seba Susan, Rohit Ranjan, Udyant Taluja, Shivang Rai, and Pranav Agarwal. Neural Net Optimization by Weight-Entropy Monitoring. In Nishchal K. Verma and A. K. Ghosh, editors, *Computational Intelligence: Theories, Applications and Future Directions - Volume II*, pages 201–213, Singapore, 2019. Springer. ISBN 978-981-13-1135-2. doi: 10.1007/978-981-13-1135-2_16.
- [33] Ilya Sutskever. Talk at the simons institute: Ilya sutskever (openai), August 14 2023. URL <https://simons.berkeley.edu/talks/ilya-sutskever-openai-2023-08-14>. Relevant timestamp at 24:07.
- [34] Szymon Talaga and Konstantinos Tsampourakis. Pybdm: Python interface to the block decomposition method (0.1.0), 2024. URL <https://zenodo.org/doi/10.5281/zenodo.10652064>.
- [35] Naftali Tishby and Noga Zaslavsky. Deep learning and the information bottleneck principle. In *2015 IEEE Information Theory Workshop (ITW)*, pages 1–5, April 2015. URL <https://ieeexplore.ieee.org/abstract/document/7133169>.
- [36] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention Is All You Need. *arXiv:1706.03762 [cs]*, December 2017. URL <http://arxiv.org/abs/1706.03762>. arXiv: 1706.03762.
- [37] Simon Wiedemann, Arturo Marban, Klaus-Robert Müller, and Wojciech Samek. Entropy-Constrained Training of Deep Neural Networks. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, July 2019. doi: 10.1109/IJCNN.2019.8852119. URL <https://ieeexplore.ieee.org/document/8852119>. ISSN: 2161-4407.
- [38] Simon Wiedemann, Heiner Kirchhoffer, Stefan Matlage, Paul Haase, Arturo Marban, Talmaj Marinč, David Neumann, Tung Nguyen, Heiko Schwarz, Thomas Wiegand, Detlev Marpe, and Wojciech Samek. DeepCABAC: A Universal Compression Algorithm for Deep Neural Networks. *IEEE Journal of Selected Topics in Signal Processing*, 14(4), May 2020. ISSN 1941-0484. doi: 10.1109/JSTSP.2020.2969554. URL <https://ieeexplore.ieee.org/abstract/document/8970294>.
- [39] Simon Wiedemann, Klaus-Robert Müller, and Wojciech Samek. Compact and Computationally Efficient Representation of Deep Neural Networks. *IEEE Transactions on Neural Networks and Learning Systems*, 31(3):772–785, March 2020. ISSN 2162-2388. doi: 10.1109/TNNLS.2019.2910073. URL <https://ieeexplore.ieee.org/abstract/document/8725933>.
- [40] Hector Zenil. A review of methods for estimating algorithmic complexity: options, challenges, and new directions. *Entropy*, 22(6):612, 2020. ISSN 1099-4300. doi: 10.3390/e22060612. URL <https://www.mdpi.com/1099-4300/22/6/612>.

- [41] Hector Zenil, Narsis A. Kiani, and Jesper Tegnér. Low-algorithmic-complexity entropy-deceiving graphs. *Physical Review E*, 96(1):012308, jul 2017. ISSN 2470-0045. doi: 10.1103/PhysRevE.96.012308. URL <http://dx.doi.org/10.1103/PhysRevE.96.012308>.
- [42] Hector Zenil, Santiago Hernández-Orozco, Narsis A. Kiani, Fernando Soler-Toscano, Antonio Rueda-Toicen, and Jesper Tegnér. A Decomposition Method for Global Evaluation of Shannon Entropy and Local Estimations of Algorithmic Complexity. *Entropy*, 20(8):605, August 2018. ISSN 1099-4300. doi: 10.3390/e20080605. URL <https://www.mdpi.com/1099-4300/20/8/605>. Number: 8 Publisher: Multidisciplinary Digital Publishing Institute.
- [43] Hector Zenil, Narsis Kiani, and Jesper Tegnér. A Review of Graph and Network Complexity from an Algorithmic Information Perspective. *Entropy*, 20(8):551, jul 2018. ISSN 1099-4300. doi: 10.3390/e20080551.
- [44] Hector Zenil, Narsis A. Kiani, Francesco Marabita, Yue Deng, Szabolcs Elias, Angelika Schmidt, Gordon Ball, and Jesper Tegnér. An Algorithmic Information Calculus for Causal Discovery and Reprogramming Systems. *iScience*, 19:1160–1172, sep 2019. doi: 10.1016/j.isci.2019.07.043.
- [45] Hector Zenil, Narsis A. Kiani, Allan A. Zea, and Jesper Tegnér. Causal deconvolution by algorithmic generative models. *Nature Machine Intelligence*, 1(1):58–66, jan 2019. ISSN 2522-5839. doi: 10.1038/s42256-018-0005-0. URL <http://www.nature.com/articles/s42256-018-0005-0>.
- [46] Hector Zenil, Narsis Kiani, Felipe Abrahão, and Jesper Tegnér. Algorithmic Information Dynamics. *Scholarpedia Journal*, 15(7):53143, 2020. ISSN 1941-6016. doi: 10.4249/scholarpedia.53143.