Grokking and the Geometry of Circuit Formation

Ahmed Imtiaz Humayun [1] **Randall Balestriero** [2] **Richard Baraniuk** [1]

Abstract

Grokking, or *delayed generalization*, is a phenomenon where generalization in a deep neural network (DNN) emerges after achieving near zero training error. Previous studies have reported the occurrence of grokking in specific controlled settings, such as DNNs initialized with large-norm parameters or transformers trained on algorithmic datasets. Recent studies have shown that grokking occurs for adversarial examples as well, in the form of delayed robustness. We connect the emergence of grokking with the geometric arrangement of circuits in the input space, and their size as well as proximity to the training data. We also demonstrate that grokking manifests in Large Language Models in next-character prediction tasks. We provide evidence that the arrangement of circuits in a DNN undergo a phase transition during training, migrating away from the training samples therefore increasing both robustness and generalization.

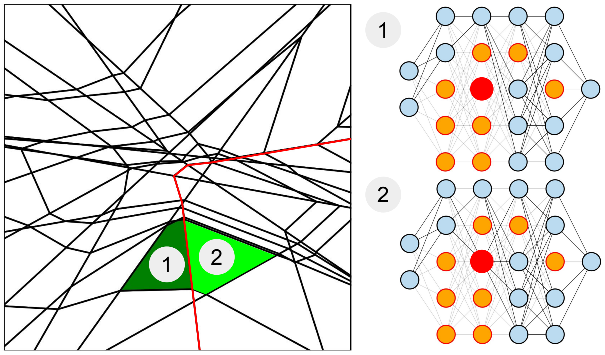

Figure 1. Geometry of unique circuits formed in the input space of a ReLU MLP $f_\theta : \mathbb{R}^2 \rightarrow \mathbb{R}$, where a circuit is defined as a unique connectivity state between the neurons of the MLP. **(Left)** Circuits in the 2D input space correspond to 'linear regions' of the DNN, a region being a set in the input space where none of the neurons change their activation state. Here we highlight two neighboring regions, between which their corresponding circuits **(right)** differ by the activation state of only the red neuron. The red neuron, like all the other neurons, can be represented as a piece-wise linear hyperplane in the input space **(left)**. Here active and inactive neurons are denoted in blue and orange.

1. Introduction

Grokking is a surprising phenomenon related to representation learning in Deep Neural Networks (DNNs) whereby DNNs may learn generalizing solutions to a task long after interpolating the training dataset, i.e., reaching near zero training error. It was first demonstrated by (Power et al., 2022) on simple Transformer architectures performing modular addition or division. Subsequently, multiple studies have reported instances of grokking for settings outside of modular addition, e.g., DNNs initialized with large weight norms for MNIST, IMDb (Liu et al., 2022), or XOR cluster data (Xu et al., 2023), or adversarial examples (Humayun et al., 2024).

In this paper, we show that grokking is a phenomenon subject to the geometric arrangement of circuits in the input space of a network. We define circuits as the unique activation states, i.e., connectivity states that a deep network can be in. We connect the notion of circuits with the linear regions formed in the input space by Deep Neural Networks with continuous piecewise linear non-linearities. Furthermore, we show that the intuition translates for non-piecewise linear DNNs, such as Transformers. We introduce the notion of circuit density as a measure for how many unique circuits are formed in a local neighborhood of the input space. Through qualitative visualizations and quantitative results on the training dynamics of circuit density, we show that there exists a strong connection between the geometric arrangement of circuits and both delayed generalization and robustness.

2. Circuits, Splines and Linear regions

A common theme in mechanistic interpretability, especially when it comes to explaining the grokking phenomenon, is the idea of 'circuit' formation during training (Nanda et al., 2023; Varma et al., 2023; Olah et al., 2020). A circuit is loosely defined as a subgraph of a deep neural network containing neurons (or linear combination of neurons) as nodes, and weights of the network as edges. In this section, we

[1] Rice University [2] Brown University. Correspondence to: Ahmed Imtiaz Humayun <imtiaz@rice.edu>.

[*] Work done while author was at Google.

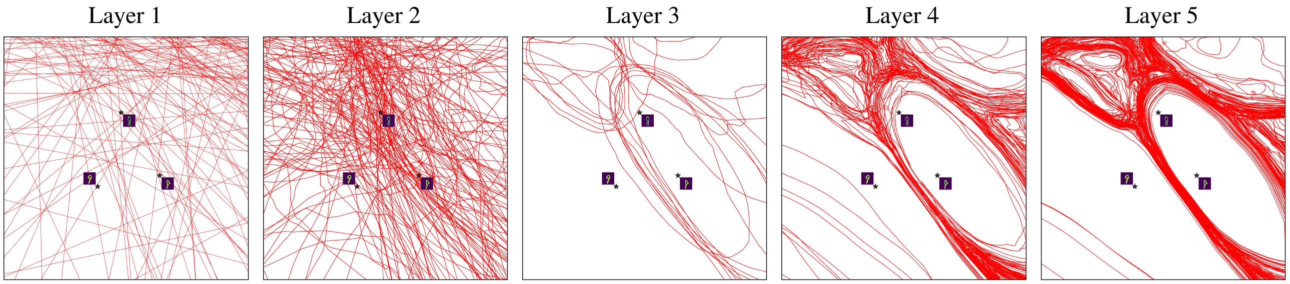

Figure 2. Layerwise visualization of the input space formation of circuits on a 2D subspace passing through a training set triad, after grokking. The circuits are visualized for an MLP with a depth of 6 and a width of 200, trained on 1,000 samples from MNIST. We see that deeper layer neurons contribute more to the formation of large circuits compared to shallower layers. This is because deeper layer neurons can be more localized in the input space due to the non-linearity induced by preceding layers. This way, a robust arrangement of circuits is formed after grokking, where large generalized circuits, i.e., those covering a large number of input space points, contain the training data points. Many of the circuits formed accumulate around and between the training set triad, thus around the decision boundary.

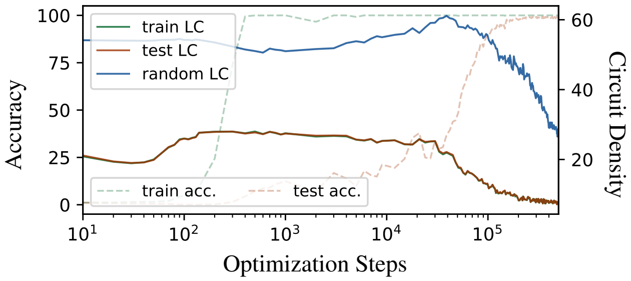

Figure 3. Accuracy and circuit density of a 2 layer Transformer trained on modular addition. We see that during grokking, the circuit density of the network drops for training, test and random points from the token embedding space as well.

connect the arrangement of circuits with the spline structure of DNNs and introduce the notion of circuit density. In Figure 1, we present the analytical spline partition formed by a deep neural network and the example circuits formed by individual linear regions in the spline partition.

2.1. Measuring Circuit Density for a Deep Network

Barak et al. (2022) introduced the notion of *progress measures* for DNN training, as scalar quantities that are causally linked with the training state of a network. We propose a geometric measure based on the arrangement of circuits in the input space. The circuit density however is equivalent to the density of unique linear operations being performed in the input space. Therefore, circuit density is equivalent to the local complexity (Hanin & Rolnick, 2019) of a DNN. We will use both of these term interchangeably in the text

Suppose a domain is specified as the convex hull of a set of vertices $\boldsymbol{V} = [\boldsymbol{v}_1, \dots \boldsymbol{v}_p]^T$ in the DNN's input space. We wish to compute the local complexity/rugosity/circuit density (Humayun et al., 2023b; Hanin & Rolnick, 2019) for neighborhood $\mathcal{V} = conv(\boldsymbol{V})$. Let's denote the DNN layer weight as $W^{(\ell)} \triangleq [\boldsymbol{w}_1^{(\ell)}, \dots, \boldsymbol{w}_{D^{(\ell)}}^{(\ell)}]$, $b^{(\ell)}$ where ℓ is the layer index, $\boldsymbol{w}_i^{(\ell)}$ is the i-th row of $W^{(\ell)}$ or weight of the i-th neuron, and $D^{(\ell)}$ is the output space dimension of layer ℓ. The forward pass through this layer for $\boldsymbol{V}$ can be considered an inner product with each row of the weight matrix $W^{(\ell)}$ followed by a continuous piecewise linear activation function. Without loss of generality, let's consider ReLU as the activation function in our network. The partition at the input space of layer ℓ can therefore be expressed as the set of all hyperplane equations formed via the neuron weights such as $\partial\Omega = \bigcup_{i=1}^{D^{(\ell)}} \mathcal{H}_i^{(\ell)}$ and $\mathcal{H}_i^{(\ell)} = \left\{\boldsymbol{x} \in \mathbb{R}^{D^{(\ell-1)}} : \langle \boldsymbol{w}_i^{(\ell)}, \boldsymbol{x} \rangle + \boldsymbol{b}_i^{(\ell)} = 0\right\}$ which is also the set of layer ℓ non-linearities. Let, $\Phi = f_{1:\ell-1}(\mathcal{V})$ be the embedded representation of the neighborhood $\mathcal{V}$ by layer $\ell - 1$ of the network. Therefore, approximating the circuit density of $\mathcal{V}$ induced by layer ℓ, would be equivalent to counting the number of linear regions in $\Phi \cap \partial\Omega = \bigcup_{i=1}^{D^{(\ell)}} \Phi \cap \mathcal{H}_i^{(\ell)}$. The local partition inside Φ results from an arrangement of hyperplanes; therefore the number of regions is of the order $\mathcal{N}^{D^{(\ell-1)}}$ (Toth et al., 2017), where $\mathcal{N} = |\{i : i = 1, 2..D^{(\ell)} \text{ and } \mathcal{H}_i^{(\ell)} \cap \Phi \neq \emptyset\}|$ is the number of hyperplanes from layer ℓ intersecting Φ. We consider $\mathcal{N}$ as a proxy for circuit density for any neighborhood Φ. To make computation tractable, let, $\Phi \approx \widehat{\Phi} = conv(f_{1:\ell-1}(\boldsymbol{V}))$. Therefore, for $\widehat{\Phi}$, any sign changes in layer ℓ pre-activations is due to the corresponding neuron hyperplanes intersecting $conv(\boldsymbol{V})$. Therefore for a single layer, the local complexity (LC) for a sample in the input space can be approximated by the number of neuron hyperplanes that intersect $\boldsymbol{V}$ embedded to that layers input space. If we consider input space neighborhoods with the same volume, then circuit density measures the un-normalized density of non-linearity in an input space locality. We highlight that this is tied to the VC-dimension of (ReLU) DNN (Bartlett et al., 2019) where the more regions are present the more expressive the decision boundary can be (Montufar et al., 2014).

Recall that DNNs operate linearly in a region-wise fashion, i.e., for all input vectors $\{x : x \in \omega\}$, the network performs the same affine operation using parameters $(\boldsymbol{A}_\omega, \boldsymbol{b}_\omega)$ while mapping x to the output. The affine parameters for any given region, are a function of the active neurons in the network as was shown by Humayun et al. (2023a) (Lemma 1). Therefore for each region, we necessarily have a circuit or subgraph of the network performing the linear operation. Between two neighboring regions, only one node of the circuit changes. Therefore, our circuit density measure is measuring the density of unique circuits formed in a locality of the input space. While in practice a DNN might have an exponential number of circuits (Hanin & Rolnick, 2019), the emergence of robust circuit formations show that towards the end of training, the number of unique circuits get drastically reduced. This is especially true for sub-circuits corresponding to deeper layers only. In Figure 2, we show the robust partition in a layerwise fashion. We can see that for deeper layers, there exists large regions, i.e., embedding regions with only one circuit operation through the layer. This result, matches with the intuition provided by Nanda et al. (2023) on the cleanup phase of circuit formation late in training.

2.2. Deep Networks With Self-Attention: Discontinuous Piecewise Affine Operators

We recall that the self-attention mapping takes the following form

$$\operatorname{softmax}\left(\boldsymbol{X}\boldsymbol{Q}\left(\boldsymbol{X}\boldsymbol{K}\right)^\top\right)\boldsymbol{X}\boldsymbol{V}, \tag{1}$$

which is then fed into a MLP block. The same mapping is used for all layers, with per-layer parameters $(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V})$. The input to that mapping is the $T \times D$ input $\boldsymbol{X}$ where T, the sequence length, is constant across layers. The dimension D may vary, though, in practice, it is kept the same across layers. For images, T corresponds to the number of patches extracted from the image and D is the number of pixels in each patch. Optionally residual connections can be added. To start our connection between self-attention and circuits, let's consider a simplified setting where the softmax has very low temperature. That is, the matrix $\operatorname{softmax}\left(\boldsymbol{X}\boldsymbol{Q}\left(\boldsymbol{X}\boldsymbol{K}\right)^\top\right)$ has rows which contain one-hot vectors at the (per-row) $\arg\max$ of $\boldsymbol{X}\boldsymbol{Q}\left(\boldsymbol{X}\boldsymbol{K}\right)^\top$. In that setting, it is clear that mapping is discontinuous piecewise affine, where the discontinuity stems from a change in the $\arg\max$. If the following MLP layers employer ReLU activation–hence the MLP is itself a continuous piecewise affine mapping–then the entire transformer mapping will be discontinuous piecewise affine. In the more realistic case of a higher temperature, i.e., $\operatorname{softmax}\left(\boldsymbol{X}\boldsymbol{Q}\left(\boldsymbol{X}\boldsymbol{K}\right)^\top\right)$ is dense, the output is not longer a (dis)continuous affine spline due to the now smooth convex combination of the inputs.

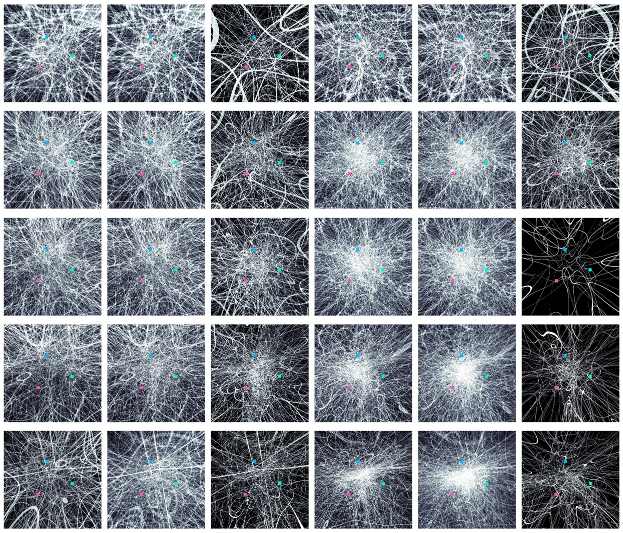

Figure 4. Layerwise circuit formation visualized for a transformer model trained on modular addition. From left to right, we present the pre-activation zero level sets for three MLPs each present in the two transformer blocks of the network. Note that only the first and fourth image, corresponds to GeLU activated MLPs. From top to bottom, we present the token embedding space circuit formations after training the network for $\{10, 12000, 30000, 108000, 498000\}$ optimization steps. In Figure 3 we present the accuracy and circuit density training dynamics for this network. The circuits are visualized on a 2D subspace in the token embedding space, that contains the inputs '`(72+65)%97=`' (pink), '`(28+93)%97=`' (green), '`(61+66)%97=`' (blue). We see that after 3×10^4 optimization steps, there is a phase change, especially visible in the deepest MLP layer. With further training we have an accumulation of the zero level-sets in between the three data points.

Curvature and Linear Regions. Formulations like that discussed above that represent DNNs as continuous piecewise affine splines, have previously been employed to make theoretical studies amenable to actual DNNs, e.g. in generative modeling (Humayun et al., 2022), network pruning (You et al., 2021), and OOD detection (Ji et al., 2022). Empirical estimates of the density of linear regions in the spline partition have also been employed in sensitivity analysis (Novak et al., 2018), quantifying non-linearity (Gamba et al., 2022), quantifying expressivity (Raghu et al., 2017) or to estimate the complexity of spline functions (Hanin & Rolnick, 2019).

3. Experiments

To visualize the circuit formation for simple DNN MLPs like that in Figure 1 we use Splinecam (Humayun et al., 2023b). To visualize circuit formation for a GPT scale large language model Figure 5 and transformers with self

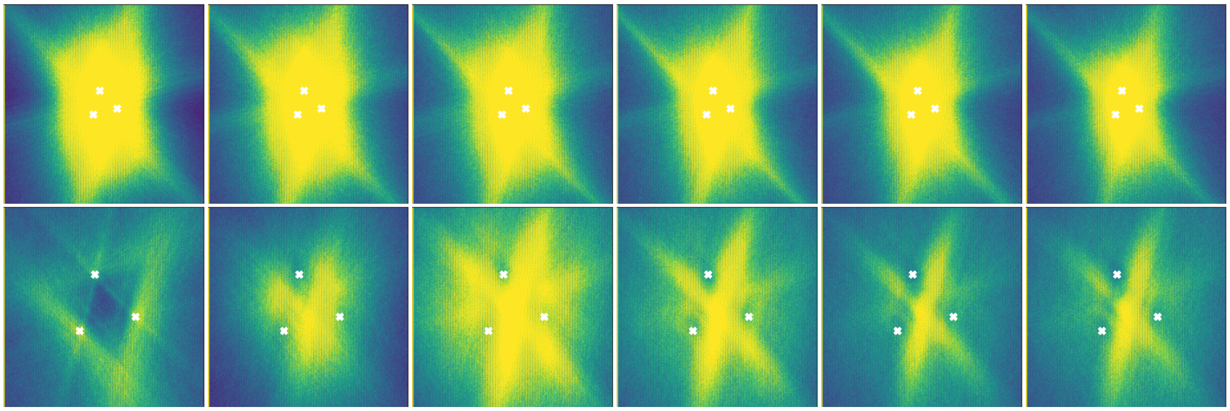

Figure 5. Circuit density visualizations for an LLM with GPT-like architecture, 6 heads and 6 transformer blocks before (top) and after (bottom) the network groks adversarial examples. We visualize a 2D subspace of the token embedding space that goes through three points (marked in white) from the training dataset. From left to right, we present the circuits formed by the pre-activation level sets of only one MLP per transformer block with GeLU activation. Prior to grokking, the circuit density is higher in the proximity of all the three examples. After grokking, we see that the circuits accumulate between the three training points, with lower circuit density in the proximity of the training points. This observation is similar to the case presented in Fig. 2 where we see low circuit density around training points, especially for deeper layers. For the first transformer block, we see that there is accumulation of non-linearities around the training points indicating that there is also a layerwise effect on the non-linearities in an LLM, much like the MLP case presented in Fig. 2.

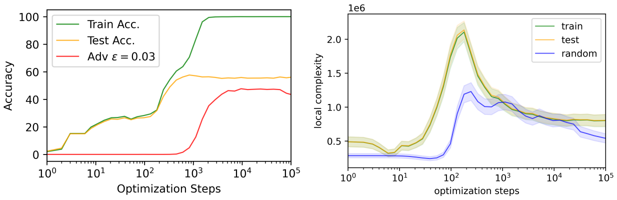

Figure 6. Accuracy (left) and local complexity (right) for an LLM with 121M parameters, 12 heads, 12 transformer blocks and a GPT-like architecture, trained on a next character prediction task using the Shakespeare dataset. We see that the GPT2 model groks adversarial examples long after training and test accuracy peaks. Adversarial examples are generated by performing non-targeted ℓ_∞ PGD (Madry et al., 2017) attacks on the token embeddings instead of the tokens directly. We see that local complexity, i.e., circuit density, undergoes a phase change here as well, that leads to grokking adversarial examples.

attention Figure 4, we use the following:

Transformers take tokens as input and use a token embedder to embed the tokens into a token embedding space. While the input space for transformers is discrete, the token embedding space can be considered continuous, with quantized bins where the transformer inputs are generally embedded to. Moreover the token embedder is trained while the whole network is being trained. Therefore, to visualize circuit formation for transformers, we consider a 2D subspace in the token embedding space that is anchored on the embeddings of three training data points. While the network is trained the 2D subspace evolves but remains anchored on the target data points. We consider a dense grid of points on the 2D subspace and measure the circuit density as detailed above for an ℓ_1 neighborhood of radius $r = 0.05$ for the transformer experiments Figure 4, and radius $r = 1e-4$ for the large language model. The area spanned by the dense grid is fixed throughout training, therefore, only the orientation of the grid changes as the network is trained. For both transformers and LLMs, we see that a phase change in the circuit formation precedes the onset of grokking.

4. Conclusions and Limitations

We present the first visualizations of circuits formed in LLMs and transformers. We present connections between circuit formation, the complexity of a network and grokking. At a high level, it is clear that the classification function being learned has its curvature concentrated at the decision boundary and approximation theory would normally dictate a free-form spline to therefore concentrate its partition regions around the decision boundary to minimize approximation error. However, it is not clear why the circuit cleanup phase occurs so late in the training process, and we hope to study that in future research. e training dynamics of stochastic gradient descent, as well as sharpness aware minimization (Andriushchenko & Flammarion, 2022) can also be studied using our framework. There can be possible connections between circuit formation and neural collapse (Papyan et al., 2020) which are not explored in this paper. The spline viewpoint of deep neural networks and its connection to circuits may provide strong geometric insights to assist in mechanistic understanding in future works as well.

References

Andriushchenko, M. and Flammarion, N. Towards understanding sharpness-aware minimization. In *International Conference on Machine Learning*, pp. 639–668. PMLR, 2022.

Barak, B., Edelman, B., Goel, S., Kakade, S., Malach, E., and Zhang, C. Hidden progress in deep learning: Sgd learns parities near the computational limit. *Advances in Neural Information Processing Systems*, 35:21750–21764, 2022.

Bartlett, P. L., Harvey, N., Liaw, C., and Mehrabian, A. Nearly-tight vc-dimension and pseudodimension bounds for piecewise linear neural networks. *The Journal of Machine Learning Research*, 20(1):2285–2301, 2019.

Gamba, M., Chmielewski-Anders, A., Sullivan, J., Azizpour, H., and Bjorkman, M. Are all linear regions created equal? In *AISTATS*, pp. 6573–6590, 2022.

Hanin, B. and Rolnick, D. Complexity of linear regions in deep networks. *arXiv preprint arXiv:1901.09021*, 2019.

Humayun, A. I., Balestriero, R., and Baraniuk, R. Polarity sampling: Quality and diversity control of pre-trained generative networks via singular values. In *CVPR*, pp. 10641–10650, 2022.

Humayun, A. I., Balestriero, R., Balakrishnan, G., and Baraniuk, R. G. Splinecam: Exact visualization and characterization of deep network geometry and decision boundaries. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3789–3798, June 2023a.

Humayun, A. I., Balestriero, R., Balakrishnan, G., and Baraniuk, R. G. Splinecam: Exact visualization and characterization of deep network geometry and decision boundaries. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3789–3798, 2023b.

Humayun, A. I., Balestriero, R., and Baraniuk, R. Deep networks always grok and here is why. *arXiv preprint arXiv:2402.15555*, 2024.

Ji, X., Pascanu, R., Hjelm, R. D., Lakshminarayanan, B., and Vedaldi, A. Test sample accuracy scales with training sample density in neural networks. In *Conference on Lifelong Learning Agents*, pp. 629–646. PMLR, 2022.

Liu, Z., Michaud, E. J., and Tegmark, M. Omnigrok: Grokking beyond algorithmic data. *arXiv preprint arXiv:2210.01117*, 2022.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

Montufar, G. F., Pascanu, R., Cho, K., and Bengio, Y. On the number of linear regions of deep neural networks. In *NeurIPS*, pp. 2924–2932, 2014.

Nanda, N., Chan, L., Lieberum, T., Smith, J., and Steinhardt, J. Progress measures for grokking via mechanistic interpretability. *arXiv preprint arXiv:2301.05217*, 2023.

Novak, R., Bahri, Y., Abolafia, D. A., Pennington, J., and Sohl-Dickstein, J. Sensitivity and generalization in neural networks: an empirical study. *arXiv preprint arXiv:1802.08760*, 2018.

Olah, C., Cammarata, N., Schubert, L., Goh, G., Petrov, M., and Carter, S. Zoom in: An introduction to circuits. *Distill*, 5(3):e00024–001, 2020.

Papyan, V., Han, X., and Donoho, D. L. Prevalence of neural collapse during the terminal phase of deep learning training. *Proceedings of the National Academy of Sciences*, 117(40):24652–24663, 2020.

Power, A., Burda, Y., Edwards, H., Babuschkin, I., and Misra, V. Grokking: Generalization beyond overfitting on small algorithmic datasets. *arXiv preprint arXiv:2201.02177*, 2022.

Raghu, M., Poole, B., Kleinberg, J., Ganguli, S., and Dickstein, J. S. On the expressive power of deep neural networks. In *ICML*, pp. 2847–2854, 2017.

Toth, C. D., O'Rourke, J., and Goodman, J. E. *Handbook of discrete and computational geometry*. CRC press, 2017.

Varma, V., Shah, R., Kenton, Z., Kramár, J., and Kumar, R. Explaining grokking through circuit efficiency. *arXiv preprint arXiv:2309.02390*, 2023.

Xu, Z., Wang, Y., Frei, S., Vardi, G., and Hu, W. Benign overfitting and grokking in relu networks for xor cluster data. *arXiv preprint arXiv:2310.02541*, 2023.

You, H., Balestriero, R., Lu, Z., Kou, Y., Shi, H., Zhang, S., Wu, S., Lin, Y., and Baraniuk, R. Max-affine spline insights into deep network pruning. *arXiv preprint arXiv:2101.02338*, 2021.