



**HAL**  
open science

## Can sparsity improve the privacy of neural networks?

Antoine Gonon, Léon Zheng, Clément Lalanne, Quoc-Tung Le, Guillaume Lauga, Can Pouliquen

► **To cite this version:**

Antoine Gonon, Léon Zheng, Clément Lalanne, Quoc-Tung Le, Guillaume Lauga, et al.. Can sparsity improve the privacy of neural networks?. GRETSI 2023 - XXIXème Colloque Francophone de Traitement du Signal et des Images, Aug 2023, Grenoble, France. hal-04062317v3

**HAL Id: hal-04062317**

**<https://hal.science/hal-04062317v3>**

Submitted on 12 Oct 2023

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

# Can sparsity improve the privacy of neural networks?

Antoine GONON\*<sup>1</sup> Léon ZHENG\*<sup>1,2</sup> Clément LALANNE<sup>1</sup> Quoc-Tung LE<sup>1</sup> Guillaume LAUGA†<sup>1</sup> Can POULIQUEN†<sup>1</sup>

<sup>1</sup>Univ Lyon, EnsL, UCBL, CNRS, Inria, LIP, F-69342, LYON Cedex 07, France

<sup>2</sup>valeo.ai, Paris, France

**Abstract** – Sparse neural networks are mainly motivated by resource efficiency since they use fewer parameters than their dense counterparts but still reach comparable accuracies. This article empirically investigates whether sparsity could also improve the privacy of the data used to train the networks. The experiments show *positive correlations between the sparsity of the model, its privacy, and its classification error*. Simply comparing the privacy of two models with different sparsity levels can yield misleading conclusions on the role of sparsity, because of the additional correlation with the classification error. From this perspective, some caveats are raised about previous works that investigate sparsity and privacy.

## 1 Introduction

Deep neural networks are state-of-the-art for many learning problems. In practice, it is possible to tune the parameters of a given network in order to perfectly interpolate the available data [18]. This overfitting regime is of practical interest since good performances can be obtained this way [2]. However, it comes with an increased risk in terms of privacy [13], since the network memorizes information about training data, up to the point of interpolating them. Among these information, some might be confidential. This raises the question of what information can be inferred given a black-box access to the model.

To detect an overfitting situation, an indicator is given by the ratio of the number of parameters by the number of data points available: the more parameters there are, the more the model can interpolate the data. In order to hinder the capacity of the model to overfit, and thus to store confidential information, this work studies the role of the number of nonzero parameters used. *Can we find a good trade-off between model accuracy and privacy by tuning the sparsity (number of nonzero parameters) of neural networks?*

Attacks such as "Membership Inference Attack" (MIA) can infer the membership of a data point to the training set [14], using only a black-box access to the targeted model. This can be problematic in case of sensitive data (medical data, etc.). Given a network, how could one reduce the risk of such attacks, while preserving its performances as much as possible?

Numerous procedures have been proposed to defend against MIAs [9]. In this work, the studied approach consists in decreasing the number of nonzero parameters used by the network in order to reduce its memorization capacity, while preserving as much as possible its accuracy. Decreasing the number of parameters can either be done by considering smaller (dense) networks as in [15], or by considering sparse subnetworks, as done here <sup>1</sup>.

\*,:†: Equal contributions. Work funded in part by the following projects: AllegroAssai ANR-19-CHIA-0009, NuSCAP ANR-20-CE48-0014, SeqALO ANR-20-CHIA-0020-01, MOMIGS of GdR ISIS and CIFRE N°2020/1643. The authors thank the Blaise Pascal center for the computational means. It uses the SIDUS [12] solution developed by Emmanuel Quemener.

1. One can empirically find sparse subnetworks that are more accurate than small dense networks: sparse subnetwork  $\simeq$  dense counterpart  $>$  smaller dense network [6].

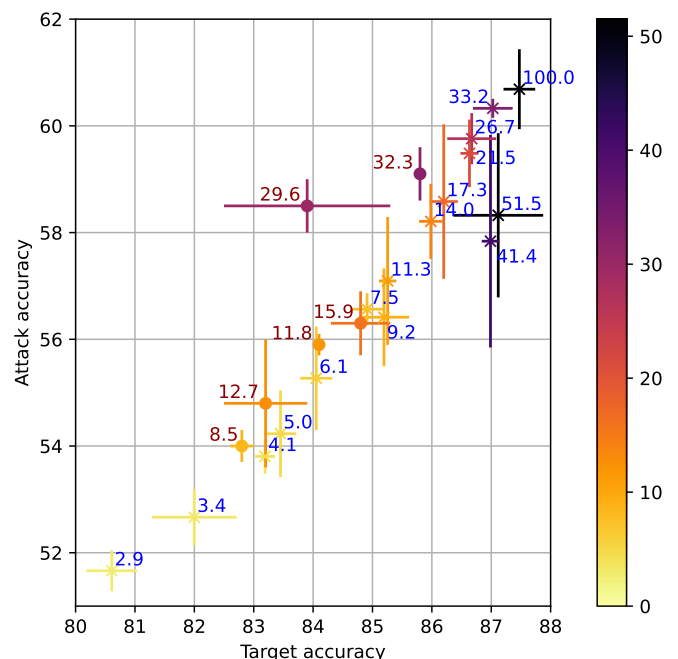


Figure 1 – Means and standard deviations of the accuracy and defense level of various sparse networks. The percentage of nonzero weights is given in blue for IMP (\* p%), and in red for Butterfly (• p%). The color indicates the percentage of nonzero parameters (in black from 50 to 100%).

**Related works.** Whether sparsity improves privacy *without further adjustment* of the training algorithm has already been partially explored [1, 16, 17]. In the light of the experiments performed in this work, shortcomings of the previous approaches will be discussed in section 4.

**Contributions and results.** This work studies two types of sparsity detailed in section 3: one without any constraints on the support (location of the nonzero weights), referred to as *unstructured* sparsity [6], the other one being *structured* sparsity [4, 11]. The results in section 4 show positive correlations between the classification error of the model, its sparsity, and its privacy (measured by the attack's error, see Figure 1).

1. This is the case for unstructured sparsity, already ex-

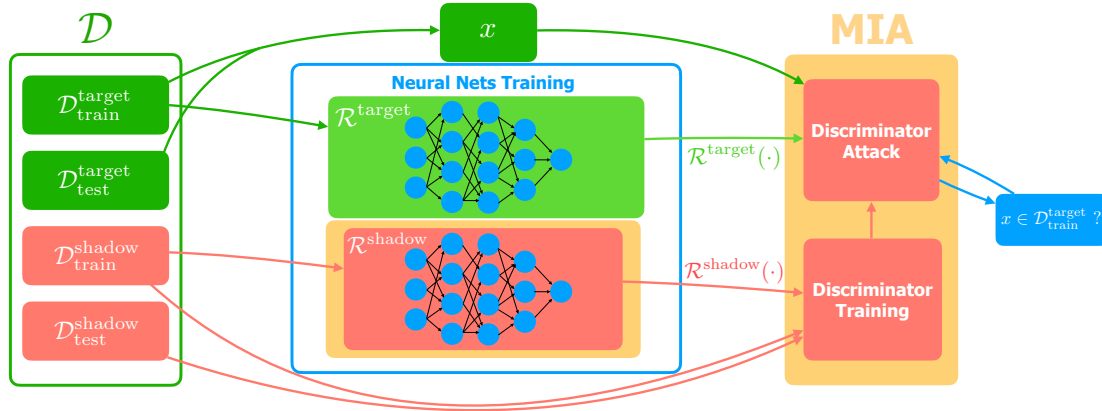


Figure 2 – Experiments obey to the following pipeline: two networks are trained in the same fashion on  $\mathcal{D}_{\text{train}}^{\text{target}}$  and  $\mathcal{D}_{\text{train}}^{\text{shadow}}$  respectively.  $\mathcal{R}^{\text{shadow}}$ ,  $\mathcal{D}_{\text{train}}^{\text{shadow}}$  and  $\mathcal{D}_{\text{test}}^{\text{shadow}}$  are then used to train a discriminator that will attack  $\mathcal{R}^{\text{target}}$  by trying to infer the membership of  $x$  in  $\mathcal{D}_{\text{train}}^{\text{target}}$ .

plored in [17], but with different conclusions. This is discussed in section 4.

2. For structured sparsity, the experiments suggest a trade-off between privacy and performance similar to the unstructured case. This is remarkable as the structure is fixed beforehand, independently of the data.

As the presence of a correlation between the classification error of the model and its privacy is observed in the experiments, future works should design experiments that control the impact of the model’s classification error in order to conclude anything on the role of sparsity. Indeed, it could be that the sparse networks encountered in the experiments are more private than their dense counterparts just because they are less accurate.

A look at previous works on sparsity and privacy shows that this fact was not taken carefully into account, see section 4.

**Plan** Section 2 introduces the MIAs used for the experiments. Section 3 describes the types of sparsity used to defend against MIAs. Section 4 gives the results of the experiments.

## 2 MIA with a shadow model

Given a dataset  $\mathcal{D}^{\text{target}}$ , and a target network  $\mathcal{R}^{\text{target}}$  trained on a subset  $\mathcal{D}_{\text{train}}^{\text{target}}$  of  $\mathcal{D}^{\text{target}}$ , a *membership inference attack* (MIA) consists in inferring the membership function

$$m_{\text{target}} : x \in \mathcal{D}^{\text{target}} \mapsto \begin{cases} 1 & \text{if } x \in \mathcal{D}_{\text{train}}^{\text{target}}, \\ 0 & \text{otherwise.} \end{cases}$$

with only a *black-box* access to the function  $x \mapsto \mathcal{R}^{\text{target}}(x)$ . Most of the existing attacks try to measure the confidence of the model in its predictions made locally around  $x$ . If the measured confidence is high enough, then the attacker answers positively to the membership question.

In practice, the most efficient attacks consist in training a discriminator model that makes a decision based on local information of  $\mathcal{R}^{\text{target}}$  around  $x$ . This discriminator is trained from one or several *shadow* network(s) [9], as explained below (see also Figure 2). Experiments of section 4 consider the less expensive case where the attacker trains a single shadow network.

**Shadow network** Assume that the attacker has access to a dataset  $\mathcal{D}^{\text{shadow}}$  from the same distribution as  $\mathcal{D}^{\text{target}}$ . The attacker trains their own shadow network  $\mathcal{R}^{\text{shadow}}$  on a subset  $\mathcal{D}_{\text{train}}^{\text{shadow}}$  of their data. Ideally,  $\mathcal{R}^{\text{shadow}}$  is trained under the same conditions as  $\mathcal{R}^{\text{target}}$  (same architecture and same optimization algorithm). At this point, the attacker has a tuple  $(\mathcal{R}^{\text{shadow}}, \mathcal{D}^{\text{shadow}}, \mathcal{D}_{\text{train}}^{\text{shadow}})$  which is similar to  $(\mathcal{R}^{\text{target}}, \mathcal{D}^{\text{target}}, \mathcal{D}_{\text{train}}^{\text{target}})$ , and knows the shadow membership function  $m_{\text{shadow}}$ .

**Discriminator** Afterwards, the attacker trains a discriminator to approximate  $m_{\text{shadow}}$ , given a black box access to  $\mathcal{R}^{\text{shadow}}$ . This discriminator is then used as an approximation of  $m_{\text{target}}$  given a black box access to  $\mathcal{R}^{\text{target}}$ . The model for the discriminator can be any classical classifier (logistic regression, neural network, etc.) [9].

## 3 Defense and neural network pruning

Training sparse neural networks is first motivated by needs for frugality in resources (memory, inference time, training time, etc.).

This work investigates another property of sparsity: whether it can also improve the privacy of the training data. Note that a perfectly confidential network has not learned anything from the data and has no practical interest. Thus, a trade-off between confidentiality and accuracy must be made according to the task at hand. In what follows, two different types of sparsity are considered.

### 3.1 Unstructured sparsity via IMP

In the first case, no specific structure is imposed on the set of nonzero weights. The weights that are set to zero (pruned) are selected by an *iterative magnitude pruning process* (IMP) [6]: (i) train a network the usual way, (ii) prune  $p\%$  of the weights having the smallest magnitude, (iii) adjust the remaining weights by re-training the network (weights that have been pruned are masked and are no longer updated), then go back to (ii) until the desired level of sparsity is reached.

This procedure finds sparse networks with empirically interesting statistical properties [6, 7].

### 3.2 Structured butterfly sparsity

In the second case, the so-called "butterfly" sparsity is structured: the weight matrices of the neural network are constrained to be a product of sparse matrices with specific supports [4, 10, 11], see Figure 3. This type of sparsity is particularly interesting for resource efficiency. Indeed, any  $N \times N$  matrix with such a factorization has a matrix-vector multiplication complexity which is sub-quadratic in theory. It is for example  $\mathcal{O}(N \log N)$  for some supports [5], against  $\mathcal{O}(N^2)$  in general.

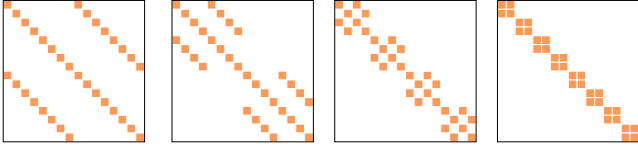


Figure 3 – Example of supports enforced to the sparse factors in a butterfly decomposition.

To enforce the butterfly structure in a neural network, the weight matrices  $\mathbf{W}$  are parameterized as  $\mathbf{W} = \mathbf{X}^{(1)} \dots \mathbf{X}^{(L)}$ , and only the nonzero coefficients of  $\mathbf{X}^{(1)}, \dots, \mathbf{X}^{(L)}$  are initialized and then optimized by stochastic gradient descent. In the case of a convolution layer, the matrix  $\mathbf{W}$  for which we impose such a structure corresponds to the concatenation of the convolutional kernels [11]. Butterfly networks can reach empirical performances comparable to a dense network on image classification tasks [4, 11].

## 4 Experimental results

All hyperparameters (including the discriminator architecture) have been chosen via grid search, with results averaged on three experiments.

**Dataset.** Experiments are performed on CIFAR-10. The dataset is randomly (uniformly) partitioned into 4 subsets  $\mathcal{D}_{\text{train}}^{\text{target}}, \mathcal{D}_{\text{test}}^{\text{target}}, \mathcal{D}_{\text{train}}^{\text{shadow}}, \mathcal{D}_{\text{test}}^{\text{shadow}}$  of 15000 images, respectively used to train and test the target and shadow networks. The membership functions are defined as in section 2, with  $\mathcal{D}^{\text{target}} := \mathcal{D}_{\text{train}}^{\text{target}} \cup \mathcal{D}_{\text{test}}^{\text{target}}$  and  $\mathcal{D}^{\text{shadow}} := \mathcal{D}_{\text{train}}^{\text{shadow}} \cup \mathcal{D}_{\text{test}}^{\text{shadow}}$ . For the target and shadow network, among their 15000 training data points, 1000 are randomly chosen and fixed for all our experiments as a validation set (used to tune the hyperparameters, and for the stopping criterion).

**Training of the target and shadow models.** The target and shadow networks have a ResNet-20 architecture [8] (272474 parameters). They are trained to minimize the cross-entropy loss by stochastic gradient descent (momentum 0.9, no Nesterov acceleration) on their respective training sets for 300 epochs, with a batch size of 256. The dataset is augmented with random horizontal flipping and random cropping. The initial learning rate is divided by 10 after 150 and 225 epochs. The weights of the neural networks are initialized with the standard method on Pytorch, following a uniform distribution on  $(-1/\sqrt{n}, 1/\sqrt{n})$  where  $n$  is the input dimension for a linear layer, and  $n$  is input dimension  $\times$  kernel width  $\times$  kernel height for a convolution.

Table 1 – Hyperparameters for the training of the target and the shadow neural networks.

Network	% of nonzero params	Initial learning rate	Weight decay
ResNet-20 dense	100 %	0.03	0.005
Butterfly ( $S = 1, L = 2$ )	32.3 %	0.3	0.0005
Butterfly ( $S = 1, L = 3$ )	29.6 %	0.3	0.0001
Butterfly ( $S = 2, L = 2$ )	15.9 %	0.3	0.0005
Butterfly ( $S = 2, L = 3$ )	12.9 %	0.1	0.001
Butterfly ( $S = 3, L = 2$ )	11.8 %	0.3	0.0005
Butterfly ( $S = 3, L = 3$ )	8.5 %	0.1	0.001
IMP with $k$ prunings	$\simeq 100 \times (0.8)^k \%$	0.03	0.005

The initial learning rate and the weight decay are reported in table 1. These hyperparameters lead to the same performance as in [8] when using the whole 50000 training images of CIFAR-10 instead of 15000 of them as it is done for the target and shadow networks.

For IMP, 24 prunings and readjustments of the parameters are performed. Each readjustment is done with the same training procedure as above (300 epochs, etc.). Before pruning, the weights are rewound to the values they had at the end of the epoch of maximum validation accuracy in the last 300 epochs.

For training ResNet-20 with the butterfly structure, the original weight matrices of some convolution layers are substituted by matrices admitting a butterfly factorization, with a number  $L = 2$  or 3 of factors, following a monotonic chain minimizing the number of parameters in the factorization [11]. The substituted layers are those of the  $S = 1, 2$  or 3 last segments<sup>2</sup> of ResNet-20.

**Discriminator training.** A discriminator takes as inputs the class  $i$  of  $x$ , the prediction  $\mathcal{R}(x)$  made by a network  $\mathcal{R}$  (target or shadow), as well as an empirical approximation of  $\frac{1}{\epsilon} \mathbb{E}(|\mathcal{R}(x) - \mathcal{R}(x + \epsilon \mathcal{N})|)$  for  $\epsilon = 0.001$  and a standard normal vector  $\mathcal{N}$ , where the approximation is made by averaging over 5 samples. This expectation encodes local first order information of  $\mathcal{R}$  around  $x$ . For each pair of networks  $(\mathcal{R}^{\text{target}}, \mathcal{R}^{\text{shadow}})$ , three discriminators (perceptrons) are trained, with respectively 1, 2, 3 hidden layer(s) and 30, 30, 100 neurons on each hidden layer. The binary cross entropy is minimized with Adam for 80 epochs, without weight decay and for three different learning rates  $\{0.01, 0.001, 0.0001\}$ .

**Privacy and performance metrics** The performance of the network is measured by its accuracy (percentage of the data correctly classified), and the privacy is measured by the accuracy of the attacker. In our case, there are as much seen as unseen data during training of the network (target or shadow). Ideally, the discriminator should not do better than guessing randomly, having then an accuracy of 50%. When the discriminator has a higher accuracy, this means that the training data is less confidential.

**Results** The experiments show that the privacy improves when both the sparsity goes up and the performance goes down. This shows that experiments of this type cannot be sufficient to conclude on the role of sparsity for privacy, since the performance is clearly correlated with the privacy, and

2. A segment is three consecutive basic blocks with the same number of filters. A basic block is two convolutional layers surrounded by a residual connection.

it could be the main explicating variable. This suggests that further experiments are needed, for instance by keeping the performance of the sparse networks at a constant level. Moreover, larger scale experiments seem to be needed to reduce the standard deviations observed on the privacy metric (the attack accuracy).

**Caveats of previous works.** In the light of Figure 1, the experiments that assess the role of sparsity for privacy should take into consideration:

- (1) the possible presence of a correlation between the accuracy of the model and the privacy;
- (2) the possibly high standard deviations of the privacy metric.

To the best of our knowledge, the three papers [1, 16, 17] are the only works that investigate the role of sparsity for privacy. Unfortunately, points (1) and (2) are not taken into account in previous approaches, as discussed below.

The experimental results in [17] suggest that increasing the sparsity *degrades* privacy. However, the Figure 7 of [17] is subject to point (1) for most of the reported configurations, and subject to point (2) in all configurations. Indeed, it shows that most of the sparse networks that are less private than their dense counterparts are also more accurate. This is exactly point (1), and it is thus impossible to rule out that it is the increased accuracy, rather than the increased sparsity, that may cause the loss in privacy. For the few models of Figure 7 in [17] that do not suffer from point (1), they still suffer from point (2). Standard deviations are not reported in [17], while Figure 1 show high standard deviations in a similar setup.

In the results of [16], no obvious correlations can be inferred between the accuracy of the model and its privacy. But the absence of standard deviations makes any conclusion on the role of sparsity hard to draw.

The results of [1] suggest that lottery ticket networks have identical privacy risks as their dense counterparts. Yet the classification accuracy of the networks was not reported and not taken into account. Moreover no standard deviations are displayed in the figures presenting the results. Again, it is hard to conclude from these experiments.

## 5 Conclusion

Sparsity is firstly motivated by resource efficiency. This work investigates whether it can also be used for privacy concerns in the case of unstructured sparsity (IMP) and structured sparsity (butterfly). The experiments show a positive correlation between the classification error of the model, its sparsity and its privacy. These results led to the identification of two important points that should be taken into account in future works that investigate sparsity and privacy: 1) the potential presence of a correlation with the classification error and 2) the potentially high variability in the privacy metric. Moreover, attacks that are more statistically relevant, but also more expensive, should be considered in the future [3].

## References

- [1] Aadesh Mahavir Bagmar, Shishira Maiya, Shruti Bidwalkar, and Amol Deshpande. Membership inference attacks on lottery ticket networks. In *ICML 2021 Workshop on Adversarial Machine Learning*, 2021.

- [2] M. Belkin, D. Hsu, S. Ma, and S. Mandal. Reconciling modern machine-learning practice and the classical bias-variance trade-off. *National Academy of Sciences USA*, 2019.
- [3] Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, Andreas Terzis, and Florian Tramèr. Membership inference attacks from first principles. In *43rd IEEE Symposium on Security and Privacy, SP 2022, San Francisco, CA, USA, May 22-26, 2022*, pages 1897–1914. IEEE, 2022.
- [4] T. Dao, B. Chen, N. S. Sohoni, A. Desai, M. Poli, J. Grogan, A. Liu, A. Rao, A. Rudra, and C. Ré. Monarch: Expressive structured matrices for efficient and accurate training. In *ICML*, 2022.
- [5] T. Dao, A. Gu, M. Eichhorn, A. Rudra, and C. Ré. Learning fast algorithms for linear transforms using butterfly factorizations. In *ICML*, 2019.
- [6] J. Frankle and M. Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *ICLR*, 2019.
- [7] J. Frankle, G. K. Dziugaite, D. Roy, and M. Carbin. Pruning neural networks at initialization: Why are we missing the mark? In *ICLR*, 2021.
- [8] K. He, X. Zhang, Sh. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*. IEEE, 2016.
- [9] H. Hu, Z. Salcic, L. Sun, G. Dobbie, P. S. Yu, and X. Zhang. Membership inference attacks on machine learning: A survey. *ACM Computing Surveys*, 2022.
- [10] Q.-T. Le, L. Zheng, E. Riccietti, and R. Gribonval. Fast learning of fast transforms, with guarantees. In *ICASSP*. IEEE, 2022.
- [11] R. Lin, J. Ran, K. H. Chiu, G. Chesi, and N. Wong. Deformable butterfly: A highly structured and sparse linear transform. In *NeurIPS*, 2021.
- [12] E. Quemener and M. Corvellec. SIDUS—the Solution for Extreme Deduplication of an Operating System. *Linux Journal*, 2013.
- [13] Maria Rigaki and Sebastian Garcia. A survey of privacy attacks in machine learning. *CoRR*, abs/2007.07646, 2020.
- [14] R. Shokri, M. Stronati, C. Song, and V. Shmatikov. Membership inference attacks against machine learning models. In *SP*. IEEE, 2017.
- [15] J. Tan, D. LeJeune, B. Mason, H. Javadi, and R. G. Baraniuk. A blessing of dimensionality in membership inference through regularization. In *AISTATS*, 2023.
- [16] Y. Wang, C. Wang, Z. Wang, S. Zhou, H. Liu, J. Bi, C. Ding, and S. Rajasekaran. Against membership inference attack: Pruning is all you need. *IJCAI*, 2021.
- [17] X. Yuan and L. Zhang. Membership inference attacks and defenses in neural network pruning. In *USENIX*. IEEE, 2022.
- [18] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals. Understanding deep learning (still) requires rethinking generalization. *ACM*, 2021.