# Boundary between noise and information applied to filtering neural network weight matrices

**Max Staats** STAATS@ITP.UNI-LEIPZIG.DE
*Leipzig University, Germany*

**Matthias Thamm** THAMM@ITP.UNI-LEIPZIG.DE
*Leipzig University, Germany*

**Bernd Rosenow** ROSENOW@PHYSIK.UNI-LEIPZIG.DE
*Leipzig University, Germany*

## Abstract

Deep neural networks have been successfully applied to a broad range of problems where overparametrization yields weight matrices which are partially random. A comparison of weight matrix singular vectors to the Porter-Thomas distribution suggests that there is a boundary between randomness and learned information in the singular value spectrum. Inspired by this finding, we introduce an algorithm for noise filtering, which both removes small singular values and reduces the magnitude of large singular values to counteract the effect of level repulsion between the noise and the information part of the spectrum. For networks trained in the presence of label noise, we find that the generalization performance improves significantly due to noise filtering.

## 1. Introduction

In recent years, deep neural networks (DNNs) have proven to be powerful tools for solving a wide range of problems [1–5], including many applications in physics [6–16]. DNNs are often highly over-parametrized [17–23], enabling them to generalize well beyond the training dataset and memorize large amounts of random data [24, 25]. However, overfitting to mislabeled examples in real-world datasets can significantly decrease generalization performance [26, 27].

Random matrix theory (RMT) has been successfully applied to analyzing neural networks [28–36]. Since DNN weights are initialized randomly, learned information after training manifests itself as deviations from RMT predictions. Even state-of-the-art DNNs have weights that follow RMT predictions [34, 36]. The singular value distribution of a weight matrix can be decomposed into a random bulk, described by the Marchenko-Pastur (MP) distribution, and a tail region. Singular vectors in the bulk follow the Porter-Thomas (PT) distribution, while large singular values and corresponding vectors deviate from RMT [34, 36].

We study weight matrices of various DNN architectures trained with and without label noise. Using a Kolmogorov-Smirnov test, we find a boundary between noise and information: singular vectors with small singular values match RMT predictions, while those with large singular values deviate significantly. This is confirmed by setting small singular values to zero and evaluating the impact on training and test accuracy. We find that small singular values and their associated vectors do not encode information. Networks trained with label noise require more singular values for good performance compared to those trained on pristine data, indicating that label noise is primarily

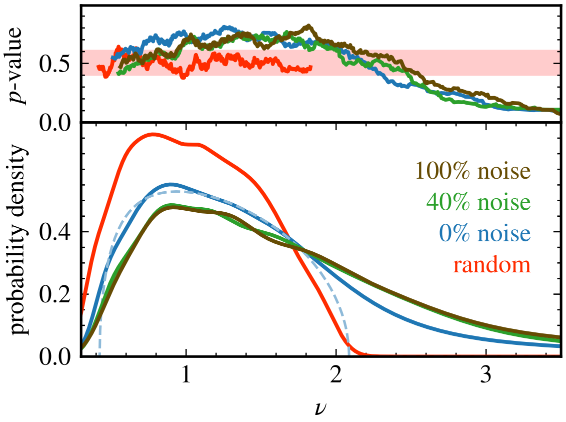

Figure 1: Analysis of singular values $\nu$ and vectors $V$ of the first weight matrix for the MLP1024 network trained with varying label noise: 0% (blue), 40% (green), and 100% (brown). Randomly initialized weights are shown in red. Upper panel: p-values of Kolmogorov-Smirnov tests for a Porter-Thomas distribution, averaged over neighboring singular values, with the $2\sigma$ region around the mean in light red. Lower panel: Corresponding singular value spectra (dashed line: fit of an MP distribution).

encoded in intermediate singular values. Motivated by these results, we propose a noise-filtering algorithm for DNN weights: (i) removing small and intermediate singular values, and (ii) reverting the shift of large singular values due to level repulsion with the noisy bulk. This algorithm improves test accuracy by up to 6% for DNNs trained with label noise. The results presented here are a concise summary of Ref. [37], which includes additional findings and detailed information.

## 2. Setup

We train several DNNs on the CIFAR-10 dataset [38] containing $N = 50000$ training images sorted into ten different classes. For training with label noise, we randomly shuffle a certain percentage of the labels. We train two kinds of architectures: (i) fully connected networks with three hidden layers, denoted as MLP1024, with layer sizes [in, 1024, 512, 512, out], and (ii) convolutional neural networks (CNNs), called miniAlexNet [25], consisting of two convolutional layers followed by max-pooling, batch normalization, and fully connected layers with regularization. We initialize the networks with a Glorot uniform distribution [39] and zero biases. For details of the training parameters refer to the Appendix A.

To show that our results are generalize to larger models, we additionally consider the two networks alexnet [40] and vgg19 [21] pretrained on the imagenet dataset [40] with 1000 classes. We compute the singular value decompositions $\mathsf{W} = \mathsf{U}\,\mathrm{diag}(\nu)\,\mathsf{V}$ with orthogonal matrices $\mathsf{U}$, $\mathsf{V}$ containing the singular vectors, and non-negative singular values $\nu$. For convolutional layers in CNNs, we first need to reshape the convolutional layer weight tensors into a rectangular shape, as shown in the Appendix F.

## 3. Boundary between noise and information

For a large random matrix, the components of its $m$-dimensional singular vectors follow a PT distribution, which is a normal distribution with zero mean and standard deviation $1/\sqrt{m}$. To account for correlations introduced by the normalization of singular vectors, we compute the Kolmogorov-Smirnov test statistic using Monte-Carlo methods. This test is applied to the empirical distribution of the singular vectors $V$ of trained networks, with details provided in the Appendix B. The resulting $p$-values, ranging from 0 to 1, indicate the likelihood of the sample being from the test distribution. We average these $p$-values over an interval of size 31, resulting in a random control fluctuating

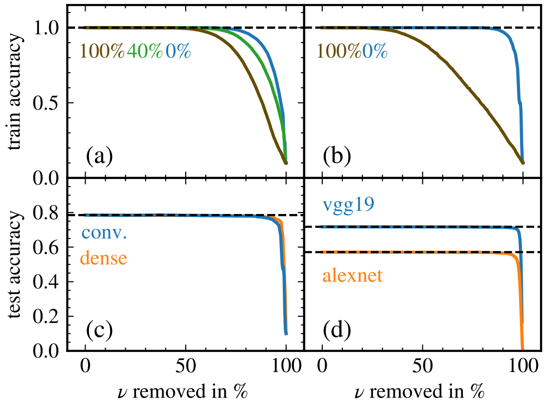

Figure 2: Boundary between information and noise, demonstrated by setting singular values to zero. Training accuracy for removing singular values from (a) the second layer of MLP1024 networks trained with various amounts of label noise (0% blue, 40% green, and 100% brown), and (b) from the second convolutional layer of miniAlexNet. Test accuracy for setting singular values to zero in (c) miniAlexNet trained without label noise in the first dense layer (orange) and the second convolutional layer (blue), and (d) in the pre-trained networks vgg19 [21] (third dense layer, blue) and alexnet [40] (second dense layer, orange).

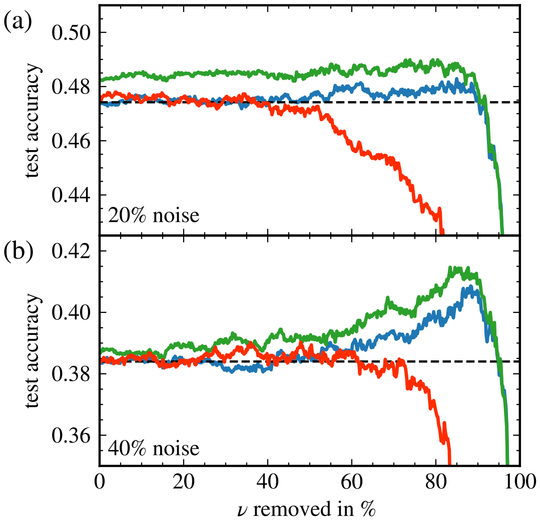

Figure 3: Dependence of the test accuracy on the removal and shifting of singular values from the second weight matrix of MLP1024 networks trained in the presence of label noise: upon setting singular values to zero (blue) and when additionally shifting them according to Eq. (3) (green) we observe a significant improvement in performance. For training with overfitting (red) no improvement is observed, indicating that information and noise are mixed in the spectrum.

around $0.5$ with a standard deviation of $\sigma = 0.05$. For the MLP1024 architecture without label noise, the averaged $p$-values drop below $2\sigma$ of the random control for singular values $\nu \gtrsim 2.3$, corresponding to $14.3$ percent of deviating singular values (blue solid line in the upper panel of Fig. 1). This indicates that information may be contained in these singular vectors. For vectors corresponding to small singular values, the $p$-values lie within or above the $2\sigma$ region due to the orthogonality requirement with vectors possessing a small mean (see Appendix C).

Additionally, we compare the empirical singular values to a MP distribution valid for random matrices [41]. The bulk of small singular values can be fitted with a MP distribution (lower panel, dashed line), which describes the spectrum of randomly initialized weight matrices. The upper end of this fit is located at singular values $\nu \approx 2$, consistent with the value $2.3$ found above, where the $p$-value falls outside the $2\sigma$ interval.

To verify that system-specific information is stored in singular vectors corresponding to large singular values, we set the smallest singular values to zero and monitored the training and test accuracy of the networks (Fig.2). The training accuracy (upper panels, blue lines) shows that learned information is stored only in the largest singular values and their corresponding vectors. Label noise, learned differently from the rule, is mainly stored in intermediate singular values, causing the training accuracy to drop significantly earlier when label noise is present. This behavior is more pronounced in a convolutional layer of miniAlexNet (Fig.2b), where the training accuracy drops sharply after removing more than 30% of the singular values, compared to about 90% for pristine training data. Examining the dependence of test accuracy on the removal of singular values (Fig.2c, d) reveals that generalization relies solely on the largest singular values and corresponding vectors. This finding holds for large convolutional networks such as AlexNet (orange) and VGG19 (blue) in Fig.2d. The observation that neural networks use only a small fraction of large singular values and vectors to learn the underlying rule explains why they generalize well despite their capacity to memorize random labels, as larger singular values and vectors store the rule, while intermediate ones can memorize random labels.

## 4. Noise filtering of weights

We next study how the generalization performance of DNNs trained with label noise depends on the removal of singular values. In Fig. 3 we show the test accuracy of an MLP1024 network when setting singular values to zero in the second layer. We show results for 20% and 40% label noise which are common in web-crawled datasets [42]. For $40\%$ noise the generalization accuracy improves by up to 2.5% when removing about 90% of singular values. See the Appendix H for differently trained models that show similar behavior. Following the improvements found for image recognition tasks when removing singular values [37], it was later found that even large transformer models can benefit from the removal of singular values [43].

In addition, we train MLP1024 networks with severe overfitting, i.e. we train for much longer than necessary to achieve 100% training accuracy, with a slower learning rate schedule. This causes an earlier drop of the test accuracy when removing singular values (red line, Fig. 3), without any noticeable improvements before the drop. We interpret this behavior as a mixing between information and noise in the spectra such that no clear boundary between these regimes exists anymore.

Level repulsion in random matrices leads to an upward shift of large singular values in the presence of a random bulk of smaller singular values [44–46]. To identify the bulk, we suggest to model the weight matrices as $\mathsf{W} = \mathsf{W}_0 + \mathsf{W}_{\text{noise}}$ with a noisy bulk $\mathsf{W}_{\text{noise}}$ and a low rank part $\mathsf{W}_0$ containing the information. This model can be justified by assuming Langevin learning dynamics as discussed in the Appendix D. The upward shift of singular values in this model can be explicitly computed [44, 47] in the limit where the dimensions of the $n_l \times n_{l-1}$ weight matrix tend to infinity with a fixed ratio $q = n_l/n_{l-1} \leq 1$. Under the assumption of i.i.d distributed elements of $\mathsf{W}_{\text{noise}}$ with standard deviation $\sigma$, the singular values $\nu$ of $\mathsf{W}$ can be shifted back to recover the unperturbed singular values $\nu_0$ of $\mathsf{W}_0$ via

$$\frac{\nu_0}{\sigma} = \frac{1}{\sqrt{2}}\sqrt{\left(\frac{\nu}{\sigma}\right)^2 - q - 1 + \sqrt{\left(\left(\frac{\nu}{\sigma}\right)^2 - q - 1\right)^2 - 4q}}\ , \tag{1}$$

where $\sigma$ is obtained from a MP fit to the spectrum [37] (for details see the Appendix E). The effect of such a transformation is shown in Fig. 4: While large values are shifted by a relatively small amount as seen in the inset, there are several singular values that get pushed far into the MP region.

When cleaning a weight matrix $W$ from noise, we use the following algorithm: i) rank order the singular values of $W$; ii) shift the large singular values outside the MP region, maintaining their rank; and iii) remove singular values from small to large based on their rank. This algorithm

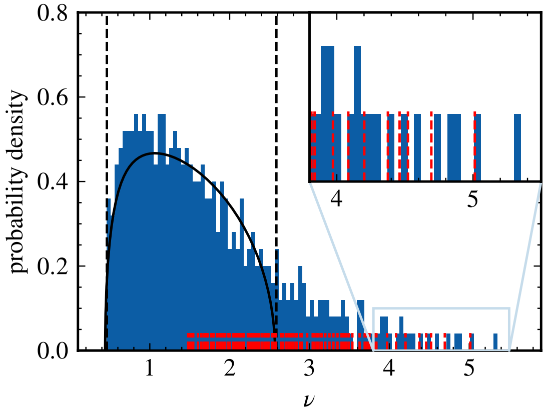

Figure 4: Singular value shift: histogram of singular values for the first weight matrix of the MLP1024 network and MP fit (solid, black) with boundaries of the MP region (dashed black lines). The dashed red lines show the locations of shifted singular values according to Eq. (1), and the inset zooms into the tail region.

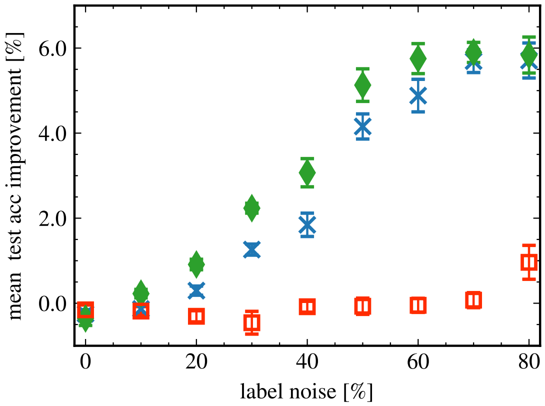

Figure 5: Average improvement of the test accuracy when removing singular values (blue, red) from all layers and when additionally shifting them (green) in MLP1024 networks, with results for both the standard learning rate schedule (blue crosses, green diamonds) and an overfitting schedule (red squares).

significantly improves generalization accuracy (see Fig. 3, green lines) when combining shifting and removing singular values, as compared to only removing them [37].

To study typical improvements from noise filtering DNN weight matrices, we train MLP1024 networks on CIFAR-10 with partially shuffled labels, keeping a validation set of 2000 and a test set of 8000 images. After training, we optimize the amount of singular values to remove for each layer, starting from the last layer and moving to the first, fixing previously filtered weights. We determine the amount to remove from the maximum validation accuracy when setting singular values to zero, starting with the smallest. We also shift singular values outside an MP-fit when this increases validation accuracy.

The results, shown in Fig. 5, indicate a 1% improvement in networks trained with the regular schedule and 20% label noise when shifting and removing singular values (green symbols). Error bars and means are computed over ten seeds. With increased label noise, improvements are more significant. However, overfitted networks (red symbols) show no improvement. Our parameter-free algorithm for removing label noise can be applied to already trained networks and combined with other methods for mitigating label noise effects, such as cleaning training data [48], modeling true labels as unknown variables [26], or using inherently label noise-robust models [49].

### 4.1. Conclusions

By comparing singular vectors to the Porter-Thomas distribution and singular values to a Marchenko-Pastur law, we argue that weight matrices of DNNs exhibit a boundary between noise and information in their spectra. We test this idea by systematically setting singular values to zero while monitoring the impact on the training and test accuracy. It turns out that (i) small singular values do not contribute to either training or test performance, (ii) large singular values encode the underlying rule, and (iii) intermediate singular values are important for the training accuracy when learning images with label noise. We suggest a filtering algorithm combining the removal of small and intermediate singular values with the downward shift of large ones, and find that it increases the generalization performance significantly in the presence of label noise. As label noise can be inherent in datasets where label annotation is difficult, we believe that filtering of weight matrices could be useful for improving the performance of DNNs in such situations.

## References

[1] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. "Deep learning". In: *Nature* 521.7553 (2015), pp. 436–444.

[2] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[3] Giuseppe Carleo et al. "Machine learning and the physical sciences". In: *Reviews of Modern Physics* 91.4 (2019), p. 045002.

[4] David Silver et al. "Mastering the game of go without human knowledge". In: *Nature* 550.7676 (2017), pp. 354–359.

[5] Yasaman Bahri et al. "Statistical mechanics of deep learning". In: *Annual Review of Condensed Matter Physics* 11 (2020), pp. 501–528.

[6] Juan Carrasquilla and Roger G Melko. "Machine learning phases of matter". In: *Nature Physics* 13.5 (2017), pp. 431–434.

[7] Evert P. L. van Nieuwenburg, Ye-Hua Liu, and Sebastian D. Huber. "Learning phase transitions by confusion". In: *Nature Physics* 13.5 (2017), pp. 435–439. ISSN: 1745-2473. DOI: \url{10.1038/nphys4037}.

[8] Kelvin Ch'ng et al. "Machine learning phases of strongly correlated fermions". In: *Physical Review X* 7.3 (2017), p. 031038.

[9] Peter Broecker et al. "Machine learning quantum phases of matter beyond the fermion sign problem". In: *Scientific reports* 7.1 (2017), pp. 1–10.

[10] Patrick Huembeli, Alexandre Dauphin, and Peter Wittek. "Identifying quantum phase transitions with adversarial neural networks". In: *Physical Review B* 97 (13 Apr. 2018), p. 134109. DOI: 10.1103/PhysRevB.97.134109. URL: https://link.aps.org/doi/10.1103/PhysRevB.97.134109.

[11] Maciej Koch-Janusz and Zohar Ringel. "Mutual information, neural networks and the renormalization group". In: *Nature Physics* 14.6 (2018), pp. 578–582. DOI: 10.1038/s41567-018-0081-4. URL: https://doi.org/10.1038/s41567-018-0081-4.

[12] Changhoon Lee et al. "Application of neural networks to turbulence control for drag reduction". In: *Physics of Fluids* 9.6 (1997), pp. 1740–1747.

[13] Xiaowei Jin et al. "NSFnets (Navier-Stokes flow nets): Physics-informed neural networks for the incompressible Navier-Stokes equations". In: *Journal of Computational Physics* 426 (2021), p. 109951.

[14] Zhihao Chen et al. "Physics-informed generative neural network: an application to troposphere temperature prediction". In: *Environmental Research Letters* 16.6 (2021), p. 065003.

[15] Javier Duarte et al. “Fast inference of deep neural networks in FPGAs for particle physics”. In: *Journal of Instrumentation* 13.07 (2018), P07027.

[16] Dan Guest, Kyle Cranmer, and Daniel Whiteson. “Deep learning and its application to LHC physics”. In: *Annual Review of Nuclear and Particle Science* 68 (2018), pp. 161–181.

[17] Ian J. Goodfellow, Oriol Vinyals, and Andrew M. Saxe. “Qualitatively characterizing neural network optimization problems”. In: *preprint arXiv:1412.6544* (2014).

[18] Daniel Soudry and Elad Hoffer. “Exponentially vanishing sub-optimal local minima in multilayer neural networks”. In: *preprint arXiv:1702.05777* (2017). URL: %5Curl%7Bhttp://arxiv.org/pdf/1702.05777v5%7D.

[19] Siyuan Ma, Raef Bassily, and Mikhail Belkin. “The Power of Interpolation: Understanding the Effectiveness of SGD in Modern Over-parametrized Learning”. In: *International Conference on Machine Learning* (2018), pp. 3325–3334. ISSN: 2640-3498. URL: %5Curl%7Bhttp://proceedings.mlr.press/v80/ma18a.html%7D.

[20] Kenji Kawaguchi, Leslie Pack Kaelbling, and Yoshua Bengio. “Generalization in deep learning”. In: *arXiv preprint arXiv:1710.05468* (2017).

[21] Karen Simonyan and Andrew Zisserman. “Very Deep Convolutional Networks for Large-Scale Image Recognition”. In: *preprint arXiv:1409.1556* (2014). URL: %5Curl%7Bhttps://arxiv.org/pdf/1409.1556%7D.

[22] Xiaohua Zhai et al. “Scaling Vision Transformers”. In: *preprint arXiv:2106.04560* (2021).

[23] Omry Cohen, Or Malka, and Zohar Ringel. “Learning curves for overparametrized deep neural networks: A field theory perspective”. In: *Physical Review Research* 3.2 (2021), p. 023034.

[24] Jake Lever, Martin Krzywinski, and Naomi Altman. “Points of significance: model selection and overfitting”. In: *Nature methods* 13.9 (2016), pp. 703–705.

[25] Chiyuan Zhang et al. “Understanding deep learning (still) requires rethinking generalization”. In: *Communications of the ACM* 64.3 (2021), pp. 107–115. ISSN: 0001-0782. DOI: \url{10.1145/3446776}.

[26] Benoıt Frénay and Michel Verleysen. “Classification in the presence of label noise: a survey”. In: *IEEE transactions on neural networks and learning systems* 25.5 (2013), pp. 845–869.

[27] Sheng Liu et al. “Robust Training under Label Noise by Over-parameterization”. In: *preprint arXiv:2202.14026* (2022).

[28] Cosme Louart, Zhenyu Liao, and Romain Couillet. “A random matrix approach to neural networks”. In: *The Annals of Applied Probability* 28.2 (2018), pp. 1190–1248. DOI: 10.1214/17-AAP1328. URL: https://doi.org/10.1214/17-AAP1328.

[29] Jeffrey Pennington and Pratik Worah. “Nonlinear random matrix theory for deep learning”. In: *Journal of Statistical Mechanics: Theory and Experiment* 2019.12 (Dec. 2019), p. 124005. DOI: 10.1088/1742-5468/ab3bc3. URL: https://doi.org/10.1088/1742-5468/ab3bc3.

[30] Andrew K. Lampinen and Surya Ganguli. “An analytic theory of generalization dynamics and transfer learning in deep linear networks”. In: *preprint arXiv:1809.10374* (2018). arXiv: 1809.10374 [stat.ML].

[31] Jeffrey Pennington, Samuel Schoenholz, and Surya Ganguli. "The emergence of spectral universality in deep networks". In: *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*. Ed. by Amos Storkey and Fernando Perez-Cruz. Vol. 84. Proceedings of Machine Learning Research. PMLR, Sept. 2018, pp. 1924–1932. URL: https://proceedings.mlr.press/v84/pennington18a.html.

[32] Nicholas P. Baskerville, Diego Granziol, and Jonathan P. Keating. "Applicability of Random Matrix Theory in Deep Learning". In: *preprint arXiv:2102.06740* (). URL: %5Curl%7Bhttps://arxiv.org/pdf/2102.06740%7D.

[33] Diego Granziol. "Beyond random matrix theory for deep networks". In: *preprint arXiv:2006.07721* (2020).

[34] Charles H Martin and Michael W Mahoney. "Implicit self-regularization in deep neural networks: Evidence from random matrix theory and implications for learning". In: *Journal of Machine Learning Research* 22.165 (2021), pp. 1–73.

[35] Charles H. Martin, Tongsu (Serena) Peng, and Michael W. Mahoney. "Predicting trends in the quality of state-of-the-art neural networks without access to training or testing data". In: *Nature Communications* 12.1 (2021), p. 4122. DOI: 10.1038/s41467-021-24025-8. URL: https://doi.org/10.1038/s41467-021-24025-8.

[36] Matthias Thamm, Max Staats, and Bernd Rosenow. "Random matrix analysis of deep neural network weight matrices". In: *Physical Review E* 106.5 (2022), p. 054124.

[37] Max Staats, Matthias Thamm, and Bernd Rosenow. "Boundary between noise and information applied to filtering neural network weight matrices". In: *Physical Review E* 108.2 (2023), p. L022302.

[38] Alex Krizhevsky, Geoffrey Hinton, et al. "Learning multiple layers of features from tiny images". In: *Tech Report* (2009).

[39] Xavier Glorot and Yoshua Bengio. "Understanding the difficulty of training deep feedforward neural networks". In: *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics* (2010), pp. 249–256. ISSN: 1938-7228. URL: %5Curl%7Bhttp://proceedings.mlr.press/v9/glorot10a%7D.

[40] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. "ImageNet classification with deep convolutional neural networks". In: *Communications of the ACM* 60.6 (2017), pp. 84–90. ISSN: 0001-0782. DOI: \url{10.1145/3065386}.

[41] V. A. Marčenko and L. A. Pastur. "Distribution of eigenvalues for some sets of random matrices". In: *Mathematics of the USSR-Sbornik* 1.4 (1967), pp. 457–483. ISSN: 0025-5734. DOI: \url{10.1070/SM1967v001n04ABEH001994}.

[42] Xuefeng Liang, Xingyu Liu, and Longshan Yao. "Review–a survey of learning from noisy labels". In: *ECS Sensors Plus* 1.2 (2022), p. 021401.

[43] Pratyusha Sharma, Jordan T Ash, and Dipendra Misra. "The truth is in there: Improving reasoning in language models with layer-selective rank reduction". In: *arXiv preprint arXiv:2312.13558* (2023).

[44] Andrew K Lampinen and Surya Ganguli. "An analytic theory of generalization dynamics and transfer learning in deep linear networks". In: *preprint arXiv:1809.10374* (2018).

[45] Laurent Laloux et al. “Noise Dressing of Financial Correlation Matrices”. In: *Physical Review Letters* 83.7 (1999), pp. 1467–1470. DOI: \url{10.1103/PhysRevLett.83.1467}.

[46] Vasiliki Plerou et al. “Universal and Nonuniversal Properties of Cross Correlations in Financial Time Series”. In: *Physical Review Letters* 83 (7 Aug. 1999), pp. 1471–1474. DOI: 10.1103/PhysRevLett.83.1471. URL: https://link.aps.org/doi/10.1103/PhysRevLett.83.1471.

[47] Florent Benaych-Georges and Raj Rao Nadakuditi. “The singular values and vectors of low rank perturbations of large rectangular random matrices”. In: *Journal of Multivariate Analysis* 111 (2012), pp. 120–135.

[48] Carla E Brodley, Mark A Friedl, et al. “Identifying and eliminating mislabeled training instances”. In: *Proceedings of the National Conference on Artificial Intelligence*. 1996, pp. 799–805.

[49] Aritra Ghosh, Himanshu Kumar, and P Shanti Sastry. “Robust loss functions under label noise for deep neural networks”. In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 31. 1. 2017.

# Appendix: Boundary between noise and information applied to filtering neural network weight matrices

## A. NEURAL NETWORK ARCHITECTURES AND TRAINING SCHEDULES

In the main text, we consider different network architectures, trained with various amounts of label noise. Tab. S1 lists the network architectures, training datasets, and accuracies achieved on each dataset. For networks trained with several different seeds, we report the average accuracy and the error of the mean. We downloaded the large pre-trained networks v) alexnet [3] via Matlab and vi) vgg19 [4] via tensorflow [5]. For the networks i)-iv), weights are initialized using the Glorot uniform distribution [6], the biases are initialized with zeros, and we standardize each image of the CIFAR-10 dataset by subtracting the mean and dividing by the standard deviation. We train networks i), iii), and iv) for 100 epochs using mini-batch stochastic gradient descent with an initial learning rate of 0.005, an exponential learning rate schedule with decay constant 0.95, momentum of 0.95, and mini-batch size 32. For the first dense layers in the CNN, we use an $L_2$ regularization with strength $10^{-4}$. For the discussion of accuracy improvements when shifting and removing singular values we also consider an overfitting training schedule ii) with 500 epochs, with a stepwise schedule starting at a learning rate 0.001, which is then reduced by a factor of 0.7 every 50 epochs. This ensures that we train for a large number of epochs after reaching 100% training accuracy. For the compressed MLP1024, we trained the network 10-times using the algorithm described in Ref. [1] using the same setup

Table S1. Neural network architectures and performance of trained networks before any filtering techniques are applied. We use d to indicate a dense layer, c for a convolutional layer, p for max pooling, f for flattening, and r for a response normalization layer (with a depth radius of 5, a bias of 1, $\alpha = 1$, and $\beta = 0.5$).

| | network | dataset | noise | train acc | test acc |
|---|---|---|---|---|---|
| i) | 3 hidden layers {d 3072, d 1024, d 512, d 512, d 10} (MLP1024) | CIFAR-10 | 0% | 100.0% | $(55.84 \pm 0.15)\%$ |
| | | | 10% | 100.0% | $(51.96 \pm 0.13)\%$ |
| | | | 20% | 100.0% | $(47.72 \pm 0.09)\%$ |
| | | | 30% | 100.0% | $(43.16 \pm 0.08)\%$ |
| | | | 40% | 100.0% | $(38.40 \pm 0.15)\%$ |
| | | | 50% | 100.0% | $(33.46 \pm 0.16)\%$ |
| | | | 60% | 100.0% | $(28.46 \pm 0.09)\%$ |
| | | | 70% | 100.0% | $(23.58 \pm 0.12)\%$ |
| | | | 80% | 100.0% | $(18.82 \pm 0.08)\%$ |
| | | | 100% | 100.0% | 10.3% |
| ii) | 3 hidden layer {d 3072, d 1024, d 512, d 512, d 10} (MLP1024) overfitting schedule | CIFAR-10 | 0% | 100.0% | $(56.15 \pm 0.10)\%$ |
| | | | 10% | 100.0% | $(51.89 \pm 0.11)\%$ |
| | | | 20% | 100.0% | $(47.94 \pm 0.11)\%$ |
| | | | 30% | 100.0% | $(43.85 \pm 0.18)\%$ |
| | | | 40% | 100.0% | $(38.93 \pm 0.21)\%$ |
| | | | 50% | 100.0% | $(34.01 \pm 0.18)\%$ |
| | | | 60% | 100.0% | $(29.05 \pm 0.08)\%$ |
| | | | 70% | 100.0% | $(24.10 \pm 0.10)\%$ |
| | | | 80% | 100.0% | $(19.04 \pm 0.09)\%$ |
| iii) | 3 hidden layers {d 3072, d 1024, d 512, d 512, d 10} (MLP1024), compressed during training [1] | CIFAR-10 | 0% | 100% | $(54.97 \pm 0.11)\%$ |
| | | | 40% | 100% | $(36.42 \pm 0.15)\%$ |
| iv) | CNN {c 300 5 × 5, p 3 × 3, r, c 150 5 × 5, p 3 × 3, r, f, d 384, d 192, d 10} (miniAlexNet) [2] | CIFAR-10 | 0% | 100.0% | 78.53% |
| | | | 20% | 100.0% | 66.38% |
| | | | 40% | 100.0% | 49.76% |
| | | | 100% | 100.0% | 10.15% |
| v) | alexnet [3] | ImageNet | 0% | | 57.1% |
| vi) | vgg19 [4] | ImageNet | 0% | | 71.8% |

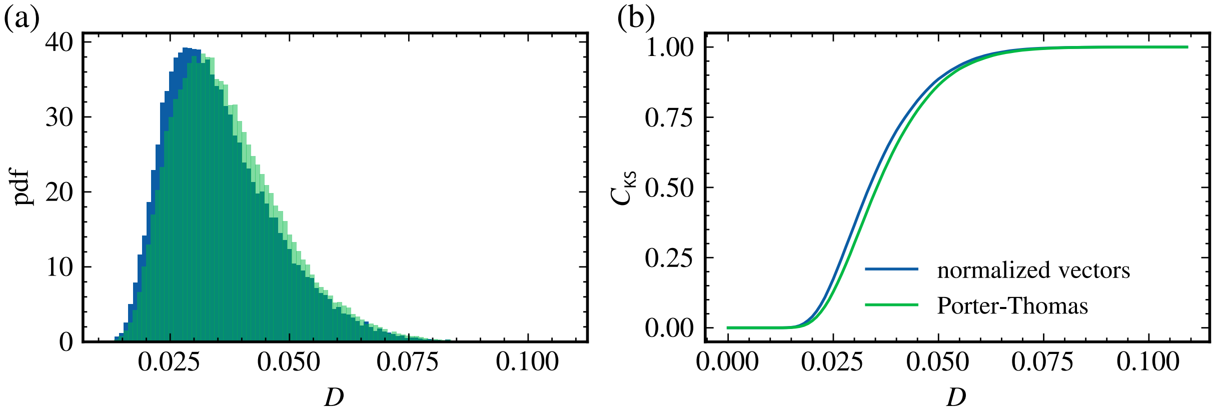

Figure S1. Comparison between Kolmogorov-Smirnov test statistics for random vectors of length 512 with i.i.d. entries (green) and for normalized vectors (blue). (a) Probability density function of the Kolmogorov-Smirnov distances Eq. (S1) obtained with Monte-Carlo sampling. (b) Cumulative distribution functions for the pdfs shown in (a). It becomes apparent that normalizing the vectors significantly changes the Kolmogorov-Smirnov test statistics even though it leaves the cumulative distribution of the vector entries unchanged.

of MLP1024 as in i). As parameters for the compression algorithm we choose $\lambda = 0.0025$ and $\mu = 0.004$.

## B. KOLMOGOROV-SMIRNOV TEST STATISTIC FOR NORMALIZED PORTER-THOMAS VECTORS

Entries of $N$-dimensional singular vectors $\xi_i$ of random matrices from the Gaussian orthogonal ensemble follow the cumulative Porter-Thomas distribution function $C_{\mathrm{PT}}(x) = 1/2 + \mathrm{erf}\left(\sqrt{N/2}\,x\right)/2$. However, their entries are not uncorrelated due to the normalization condition $\sum_i \xi_i^2 = 1$. Hence, the statistic of the usual Kolmogorov-Smirnov test which determines the $p$-values for uncorrelated data cannot be applied here. We obtain the statistic for normalized vectors using Monte-Carlo sampling of 50000 normalized random vectors $\boldsymbol{\xi}^{(k)}$ by computing the empirical cdf for each vector $C^{(k)}_{\mathrm{emp}}$ to find the corresponding Kolmogorov-Smirnov distances

$$D^{(k)} = \sup_x \left| C^{(k)}_{\mathrm{emp}}(x) - C_{\mathrm{PT}}(x) \right| \ . \tag{S1}$$

The cdf $C_{\mathrm{KS}}(D)$ for the 50000 distances $\{D^{(k)}\}$ allows to determine the $p$-values for a given new vector $\boldsymbol{\xi}$ with deviation $D(\boldsymbol{\xi})$ as $1-p = C_{\mathrm{KS}}(D(\boldsymbol{\xi}))$. The deviations between the usual Kolmogorov-Smirnov statistic (green) and the sampled statistic for normalized vectors (blue) are shown in Fig. S1.

## C. INCREASED $p$-VALUES

In main text Fig. 1 we test the singular vector entries of trained weight matrices against the Porter-Thomas distribution and find that the $p$-values in the random part of the spectrum are significantly higher than statistically expected. We argue that this is due to the presence of a few non-random singular vectors that store the information. These vectors force the random singular vectors to have a narrower distribution around the most likely part of the Porter-Thomas distribution (normal distribution with zero mean) due to the constraint of orthogonality with the deviating singular vectors with large singular values.

For example, using the same test statistic as described in Sec. B such that random normalized vectors from the Porter-Thomas distribution have on average a $p$-value of 0.5, the subset of vectors with zero mean have an average $p$-value of 0.74. We show in Fig. S2(a) that the mean values of singular vector entries for small singular values of trained weight matrices (0% label noise blue, 40% green, 100% brown) are indeed smaller than the expected values ($2\sigma$ range in light red stripe) for fully random matrices (red) while the means are much larger for large singular values where the information is stored.

The increase of $p$-values can also be shown for a simple model, adding a low-rank matrix $\delta\mathsf{W}$ to a fully random matrix $\mathsf{W}_{\mathrm{random}}$ that would have singular vectors with $p$-value of 0.5 on average. For this we draw a $1024 \times 512$ matrix

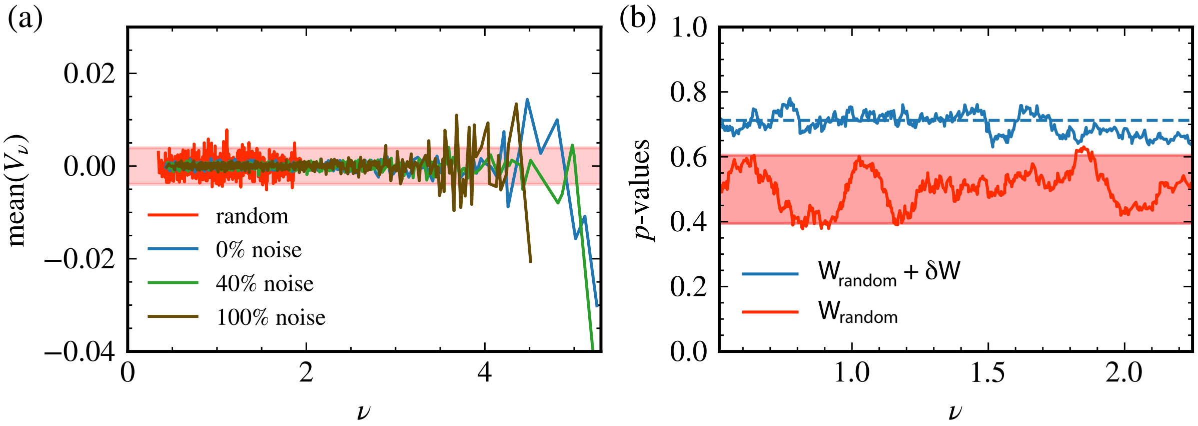

Figure S2. (a) Mean values of singular vector components for the first hidden layer of the trained MLP1024 DNN (same vectors as in Fig. 1 main text) as a function of the corresponding singular values $\nu$. The red line shows the distribution of means for singular vectors of a random weight matrix with i.i.d. Gaussian entries, with the corresponding $2\sigma$ region shown as a transparent red stripe. We observe means much closer to zero for singular vectors of the trained weight matrix in the case of small singular values, and significantly larger means for the vectors corresponding to large singular values. (b) Kolmogorov-Smirnov $p$-values (with test statistics from Sec. B) for singular vectors of a $1024 \times 512$ matrix $\mathsf{W}_{\text{random}}$ (red) with i.i.d. Gaussian entries with zero mean and variance $1/512$; $2\sigma$ region for $p$-values shown in light red. When adding a matrix $\delta\mathsf{W}$ of rank ten with entries from a Gaussian distribution with the same variance but with mean $-0.01$ (similar to mean values observed for empirical vectors corresponding to the largest singular values in (a)), the sum $\mathsf{W}_{\text{random}} + \delta\mathsf{W}$ (blue) has singular vectors with significantly increased $p$-values, due to the requirement of orthogonality between singular vectors with large and small singular values. The $p$-values are averaged over neighboring singular values with a window size of 31.

$\mathsf{W}_{\text{random}}$ with Gaussian distributed i.i.d. entries with mean zero and variance $1/512$, for which the $p$-values of singular vectors fluctuate around 0.5 (see red curve in Fig. S2(b), with most values within the $2\sigma$ region (light red stripe)). We then draw a second $1024 \times 512$ matrix with i.i.d. Gaussian distributed entries with mean $-0.01$ and variance $1/512$, compute the singular value decomposition, and reconstruct the matrix by only keeping the largest 10 singular values yielding a rank 10 matrix $\delta\mathsf{W}$. We then analyze the $p$-values of the singular vectors of $\mathsf{W}_{\text{random}} + \delta\mathsf{W}$. We find that the $p$-values are increased (blue line in Fig. S2(b)), with mean 0.71, which shows that in the presence of a few singular vectors with a distribution different from the random bulk, we expect the $p$-values in the bulk to be increased due to the enforced orthogonality to the singular vectors with a finite mean.

## D. MODELING NOISE IN THE WEIGHT MATRICES

The weight matrix $\mathsf{W}(t)$ of a DNN at time step $t$ in the training process is related to the previous weights as $\mathsf{W}(t) = \mathsf{W}(t-1) - \alpha\, \nabla_{\mathsf{W}} \mathcal{L}^{\mu(t)}$, where $\mathcal{L}^{\mu(t)}$ is the mini-batch loss function at time $t$, $\nabla_W$ is the gradient with respect to the weights $W$, and $\alpha$ is the learning rate. We can rewrite this equation in terms of the true loss function $\mathcal{L} = \langle \mathcal{L}^{\mu} \rangle_{\mu}$ such that

$$\mathsf{W}(t) = \mathsf{W}(t-1) - \alpha\, \nabla_{\mathsf{W}} \mathcal{L} - \alpha\, \nabla_{\mathsf{W}} (\mathcal{L}^{\mu(t)} - \mathcal{L}) \; . \tag{S2}$$

If we identify the gradient of the deviation of the mini-batch loss function from the true loss function as noise $\eta$ and consider the continuous time limit, we find for the neural networks dynamics

$$\frac{d\mathsf{W}}{dt} = -\alpha \nabla_{\mathsf{W}} \mathcal{L} + \eta \; , \tag{S3}$$

We assume that in the later stages of training the weights fluctuate around a minimum $W_0$ due to the stochasticity of training, and approximate $\mathcal{L}$ in this minimum to second order:

$$\mathcal{L}(\boldsymbol{W}) \approx \mathcal{L}_0 + \frac{1}{2} (\boldsymbol{W} - \boldsymbol{W}_0)^T \mathsf{H} (\boldsymbol{W} - \boldsymbol{W}_0) \; . \tag{S4}$$

Here, $\mathsf{H}$ denotes the Hessian at the minimum. For a large number of images in the training dataset and a sufficiently small batch size, the covariance matrix of the gradients is approximately equal to the Hessian of the loss landscape,

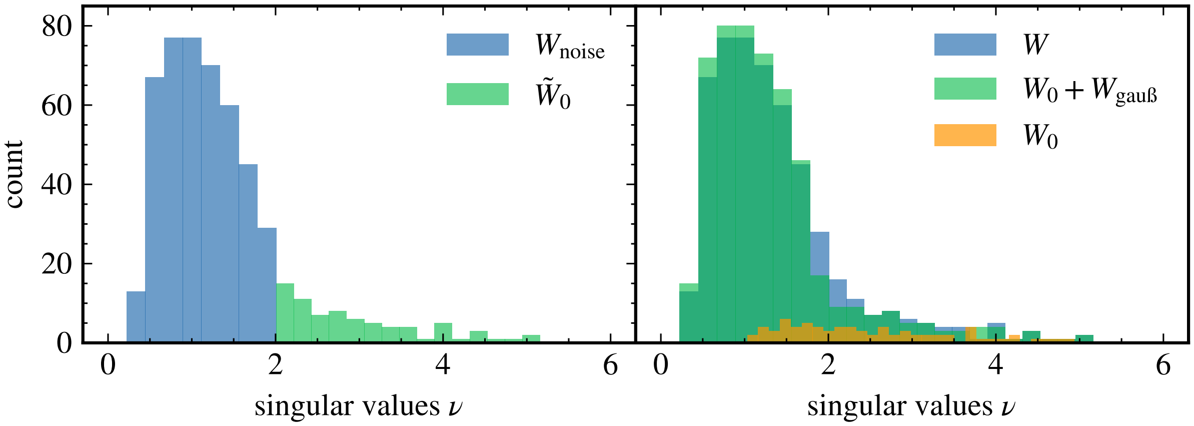

Figure S3. Decomposition of the weight matrix in a low rank and noise part. Left panel: spectrum of the second hidden layer weight of an MLP1024 network with MP part (blue) and tail (green). Right panel: spectrum of the full weight matrix $W$ (blue), corrected low rank part $W_0$ (orange), and spectrum of $W_0 + W_{\text{gauß}}$ obtained when adding a random matrix to $W_0$ (green).

$\mathsf{C} \approx \mathsf{H}$ [7–9]. We therefore assume that the noise $\eta$ follows a multivariate Gaussian distribution with zero mean and covariance matrix proportional to $\mathsf{C}$. Following the notation of Ref. [7] that $\mathsf{H} \approx \mathsf{C} = \mathsf{V}\Lambda\mathsf{V}^T$ and $\boldsymbol{z} = \mathsf{V}^T(\boldsymbol{W} - \boldsymbol{W}_0)$, Eq. (S3) is given by

$$\nabla_{\boldsymbol{W}}\mathcal{L} = \mathsf{V}\Lambda\boldsymbol{z} \ . \tag{S5}$$

As in Ref. [7], we express the noise as

$$\eta = \frac{1}{\sqrt{T}}\mathsf{V}\Lambda^{1/2}\frac{\mathrm{d}\lambda}{\mathrm{d}t} \ , \tag{S6}$$

where $\lambda$ is a *Wiener* random variable, i.e. $\mathrm{d}\lambda/\mathrm{d}t$ is an uncorrelated Gaussian random variable. This ensures that $\eta$ is equivalent to the noise in Eq. (S3) above. Here, $T$ is a constant that depends on the learning rate and the batch size and takes the role of a temperature in the Langevin dynamics. This brings the Langevin equation into the *Ornstein-Uhlenbeck* form reported in Refs. [7, 10, 11],

$$\mathsf{V}^T\frac{\mathrm{d}(\boldsymbol{W} - \boldsymbol{W}_0)}{\mathrm{d}t} = \frac{\mathrm{d}\boldsymbol{z}}{\mathrm{d}t} = -\alpha\Lambda\boldsymbol{z} + \frac{1}{\sqrt{T}}\Lambda^{1/2}\frac{\mathrm{d}\lambda}{\mathrm{d}t} \ . \tag{S7}$$

The stationary solution of this process fulfills [7]

$$\langle \boldsymbol{z}\boldsymbol{z}^T \rangle = \frac{1}{2T}\mathbb{1} \ . \tag{S8}$$

Transforming back to the original weights, using that $\mathsf{V}$ only depends on the energy landscape around the minimum and $\mathsf{V}^T\mathsf{V} = \mathbb{1} = \mathsf{V}\mathsf{V}^T$, we find

$$\langle (\boldsymbol{W} - \boldsymbol{W}_0)(\boldsymbol{W} - \boldsymbol{W}_0)^T \rangle = \frac{1}{2T}\mathbb{1} \ . \tag{S9}$$

The entries of the matrix $\boldsymbol{W}_{\text{noise}} = \boldsymbol{W} - \boldsymbol{W}_0$ must hence be i.i.d. random variables with zero mean and variance $T$. The singular value spectra for this splitting of a weight matrix into $W_0 + W_{\text{noise}}$ can be seen, based on an example, in Fig. S3.

## E. FITTING MARCHENKO-PASTUR CURVES

In the main text, we fit the singular value density of $n \times m$ weight matrices with $m \leq n$ to the Marchenko-Pastur (MP) distribution [12]

$$P(\nu) = \begin{cases} \frac{n/m}{\pi\sigma^2\nu}\sqrt{(\nu_{\max}^2 - \nu^2)(\nu^2 - \nu_{\min}^2)} & \nu \in [\nu_{\min}, \nu_{\max}] \\ 0 & \text{else} \end{cases} , \tag{S10}$$

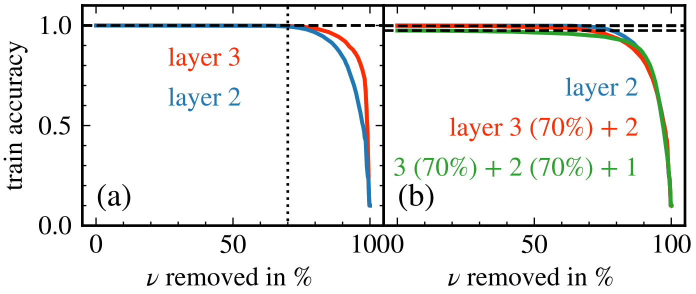

Figure S4. Training accuracy when setting singular values to zero in multiple layers of an MLP1024 network. Left panel: Results for only removing from the second hidden layer (blue) and the third hidden layer (red). Right panel: Results for removing from layer 2 (blue), layer 2 after 70% of singular values have already been removed from layer 3 (red), and from layer 1 after 70% from both layer 2 and 3 have been removed (green).

with $\nu_{\substack{\max\\\min}} = \sigma(1 \pm \sqrt{m/n})$ to obtain the standard deviation of the noise term $\sigma$ used in the shifting formula Eq. (3). As the spectrum additionally has singular values in the tail, the MP part is not normalized and the end of the MP region is not known a priori. We therefore first broaden the DNN spectrum using Gaussian broadening [13]

$$P(\nu) \approx \frac{1}{m}\sum_{k=1}^{m} \frac{1}{\sqrt{2\pi\sigma_k^2}} \exp\left(-\frac{(\nu-\nu_k)^2}{2\sigma_k^2}\right) , \tag{S11}$$

with $\sigma_k = (\nu_{k+a} - \nu_{k-a})/2$ and windows size $2a + 1 = 31$, and then fit an adjusted MP distribution, where we use $\nu_{\max}$ and the maximum height as independent fit parameters, and infer $\nu_{\min}$ from the smallest singular values. This yields an estimate for $\nu_{\min}$ and $\nu_{\max}$. We then fit the proper MP distribution Eq. (S10), only depending on $\sigma$, to a normalized histogram of the singular values between $\nu_{\min}$ and $\nu_{\max}$.

## F. RESHAPING AND FILTERING OF CONVOLUTIONAL LAYERS

For convolutional layers, weights are four dimensional. In order to compute the singular value decomposition we first order the number of input and output channels, width and height by their size such that $D_1 \geq D_2 \geq D_3 \geq D_4$ (e.g. for a layer with shape $150 \times 300 \times 5 \times 5$ we choose $D_1 = 300$, $D_2 = 150$, $D_3 = 5$, $D_4 = 5$). Now the three smallest dimensions are grouped together such that the resulting matrix $\tilde{W}$ has dimension $(D_1, D_2 \cdot D_3 \cdot D_4)$ according to

$$\tilde{\mathsf{W}}_{k,(l\cdot D_3\cdot D_4+m\cdot D_4+n)} = \mathsf{W}_{k,l,m,n} \tag{S12}$$

with indices counted from zero. By ordering the dimensions first, instead of having a constant reshaping scheme, we ensure that there is a large enough number of singular values to perform our removal scheme. In practice, this leads to a reshaping that is most often identical to one of the procedures used in [1, 14] (see also Fig. S5 d).

## G. EFFECTS OF DEPTH

The amount of singular values which can be removed without affecting the performance varies from layer to layer (for an example see left panel of Fig. S4 where we show the removal from the second or the third layer of an MLP1024 network). We find that setting singular values to zero in a given layer is almost independent of the previous removal from other layers: In the right panel of Fig. S4, we show results for an MLP1024 network where we remove (i) from layer 2 alone, (ii) from layer 2 after already 70% of the spectrum from layer 3 has been removed, and (iii) from layer 1 after 70% of the singular values from both layer 2 and 3 were set to zero. In all cases we find similar results such that the majority of singular values and corresponding vectors can be removed without significantly influencing the performance.

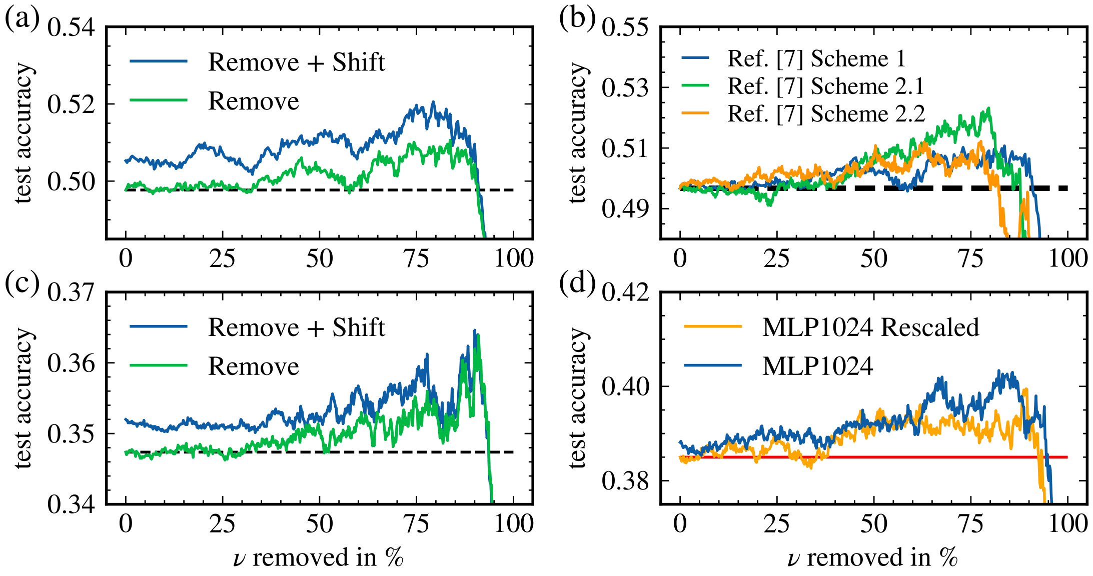

Figure S5. Effect of noise filtering for different DNN architectures, reshaping, learning rate schedule, and rescaling for the case of 40% label noise. In (a) we noise filter the second convolutional layer of miniAlexNet to significantly improve the performance. In (b) we show that the results stay consistent under different reshaping schemes. In (c) we use the learning schedule of Ref. [15] for MLP1024, while (d) shows the improvements we obtain for a rescaled representation [16] of MLP1024 as compared to the original one.

## H. GENERALITY OF THE RESULTS

To show that the noise filtering results presented in the main text are typical for other network architectures and representations, we show additional results in Fig. S5. In panel (a) we show the increase of the test accuracy of miniAlexNet trained with 40% label noise when removing or when shifting and removing singular values from the second convolutional layer. In panel (b) we show that the improvements in (a) are independent of the way we reshape the convolutional layer. Using the reshaping schemes proposed in Ref. [1], we find similar or even slightly larger improvements. We note that scheme 1 and our reshaping algorithm are equivalent in this case. In panel (c) we train MLP1024 using a learning rate schedule which keeps the learning rate high in the first epochs to find a wider minima as proposed by Ref. [15]. For 40% label noise we again find significant improvements of the test accuracy when noise filtering is applied. In panel (d) we rescale the network to a different representation by normalizing the weights $w_i$ leading into a neuron to unity ($\sqrt{\sum w_i^2} = 1$) and rescaling the output accordingly [16]. This changes the singular value decomposition significantly, however, the improvement of the test accuracy remains similar to the case without rescaling.

[1] Y. Idelbayev and M. A. Carreira-Perpinán, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2020) pp. 8049–8059.

[2] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, Communications of the ACM **64**, 107 (2021).

[3] A. Krizhevsky, I. Sutskever, and G. E. Hinton, Communications of the ACM **60**, 84 (2017).

[4] K. Simonyan and A. Zisserman, preprint arXiv:1409.1556 (2014).

[5] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," (2015), software available from tensorflow.org.

[6] Xavier Glorot and Yoshua Bengio, Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics , 249 (2010).

[7] S. Jastrzebski, Z. Kenton, D. Arpit, N. Ballas, A. Fischer, Y. Bengio, and A. Storkey, arXiv preprint arXiv:1711.04623 (2017).

[8] L. Sagun, U. Evci, V. U. Guney, Y. Dauphin, and L. Bottou, arXiv preprint arXiv:1706.04454 (2017).

[9] Z. Zhu, J. Wu, B. Yu, L. Wu, and J. Ma, arXiv preprint arXiv:1803.00195 (2018).

[10] T. Poggio, K. Kawaguchi, Q. Liao, B. Miranda, L. Rosasco, X. Boix, J. Hidary, and H. Mhaskar, arXiv preprint arXiv:1801.00173 (2017).

[11] S. Mandt, M. D. Hoffman, and D. M. Blei, arXiv preprint arXiv:1704.04289 (2017).

[12] V. A. Marčenko and L. A. Pastur, Mathematics of the USSR-Sbornik **1**, 457 (1967).

[13] V. Plerou, P. Gopikrishnan, B. Rosenow, L. A. N. Amaral, T. Guhr, and H. E. Stanley, Physical Review E **65**, 066126 (2002).

[14] Y. Yoshida and T. Miyato, preprint arXiv:1705.10941 (2017).

[15] N. Iyer, V. Thejas, N. Kwatra, R. Ramjee, and M. Sivathanu, arXiv preprint arXiv:2003.03977 (2020).

[16] F. Pittorino, A. Ferraro, G. Perugini, C. Feinauer, C. Baldassi, and R. Zecchina, arXiv preprint arXiv:2202.03038 (2022).