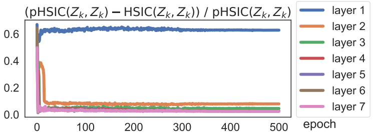
R1-5. Comparison with previous work. In the current version, we compare our work with layer-wise objectives (Sec. 1; Sec. 2-4 compare with [Nøkland, 2019]). Comparison with biologically plausible work in Sec. 5 was limited to variations of Feedback Alignment (FA) [Lillicrap, 2016; Moskovitz, 2018; Akrout, 2019]. We'll add more details, including experimental results, which are consistent with [Moskovitz, 2018]: FA performs significantly worse than backprop (80% on the 1x net from Sec. 4.3), sign-symmetry [Liao, 2016] is comparable with our methods (91% with batchnorm; our methods achieve 90-91%) but requires signs of the feedforward weights. Note that all those algorithms need a backward pass and a dedicated circuitry to propagate it, and don't perform better than backprop. We'll reference target prop [Bengio, 2014; Lee, 2015] and equilibrium prop [Scellier, 2017], but they don't seem to scale to large networks/hard problems ([Bartunov, 2018] report that target prop performs worse than FA). We'll extend the comparison with layer-wise objectives (with e.g. [Mostafa, 2018; Krotov, 2019]). We do compare our work with [Nøkland, 2019], using the same architecture in Sec. 4.3 and a slightly modified rule: cosine similarity + grouping corresponds to sim-bpf in [Nøkland, 2019], but achieves better results (88.8% w/o batchnorm and 91.3% with it (Appendix D.6) vs. 86.6% with sim-bpf and 91% when using labels in each layer, both with batchnorm). We will make the comparison more explicit (also see Appendix B.4 for why cosine similarity is implausible). Regarding Information Bottleneck (IB) and kernel approaches, we are not aware of any papers claiming good performance with IB on hard tasks; the only kernel approach to layer-wise training we know of is HSIC bottleneck, which doesn't scale well (<60% accuracy on CIFAR10 in a 5-block ResNet [Ma, 2019]; our experiments weren't successful either); both approaches are discussed in Sec. 1-2.

R1-2. HSIC/pHSIC approximation (Eqs. 6-7). We explain in Appendix A.2 why it holds mathematically: $\mathrm{pHSIC}(A, A) - \mathrm{HSIC}(A, A) = 2\mathbb{V}\mathrm{ar}_{a_1}[\mathbb{E}_{a_2} k(a_1, a_2)]$ (small for low variance of "similarity to the mean" $\mathbb{E}_{a_2} k(a_1, a_2)$). Empirically, the approximation is tight: the figure shows that the relative difference between pHSIC and HSIC stays small throughout training with the pHSIC objective (1x conv net from Sec. 4.3, Gaussian kernel with grouping and divnorm). Training with HSIC instead generally increased the test accuracy by 1-2%. We'll add those results and provide more intuition in the main text.



R1. Eq. 11 breaks the online character, but we explained how it can be computed online (as an exponential running average) in Appendix C.1 (we'll clarify this in the main text). **Sec 3.3 (eq 13 to 16):** we added it because our rule is unusual as it takes two points, raising question of how it can be plausibly computed. However, we need a batch version to use GPUs. **Adam/batchnorm:** we'll move it to the main text. **Grouping:** we'll add more clarification. Briefly, instead of computing kernels over all neurons (say 100), we arrange them into groups (say 4, with 25 neurons per group) with a single value representing each group and compute kernels over those groups (over 4-dim vectors).

R2. Tab. 1: we appreciate the references for Tab. 1 and will improve the comparisons, but it's worth noting the main result is Tab. 2 (larger networks, much better performance). **Eqs. 7-8:** we're sorry for the confusion, Eq. 8 had a typo (RHS was HSIC(A,A), not pHSIC) corrected in the appendix (in the first lines). **Conv parameters:** we overlooked this and will add them to the paper. **Eq. 12** is correct (cf. Eq. 10; derivation in Appendix B.2).

R4. Performance: while SGD outperforms our rules in small networks (which still work well with grouping+divnorm on MNIST datasets), we show it is due to the size of those networks – in Sec. 4.3 our rules nearly match SGD performance in large nets (Tab. 2: 90-91% for our methods vs. 91-92% for SGD in the 2x net). **Plausibility/performance:** the Gaussian kernel with grouping and divisive normalization performs much better than the plain one without losing plausibility. As seen in Eq. 22, the update stays Hebbian and the third factor receives a single additional multiplier, computed from the the normalizer in each group. **Large amounts of data:** we're sorry for misunderstanding, but in the abstract we mentioned large amounts of *labeled* data, which we did address with a weakly supervised rule that only requires a binary similarity signal. **Inference/training bifurcation:** our rule doesn't have a backward pass at all, as every layer performs Hebbian learning independently (without signals from the upstream layers).

R5. We agree "biological plausibility" is subjective, but we did try to address some of your points and we'll elaborate on them in the paper. **Recurrence:** networks in Sec 4.3 are "recurrent" in the sense that they use divisive normalization. While in a deep feedforward network this works as simple division, a more realistic implementation would require a recurrent computation. We don't have other types of recurrence common in the visual stream (e.g. feedback connections), and introducing them would be an interesting future direction. **CNNs** are indeed implausible, and removing weight sharing is a future direction (however, most cited work on the same topic also uses CNNs). **Negative firing rates:** as SELU saturates for negative values, it can be thought of as firing rate relative to the background. **Supervised learning:** we agree the supervised framework is implausible, and that's what we tried to address with our learning rule: instead of label information, it only needs a binary similarity signal for two consecutive points (the output layer uses labels only to measure accuracy). Such a learning signal is reminiscent of self-supervised learning (e.g. [van den Oord, 2018; Löwe, 2019]) and can likely be applied in that setting. **Complicated update rule:** the Gaussian kernel with grouping and divisive normalization (the main focus of the experiments) needs to combine only three locally accessible signals: global modulation, changes in the local activity, and changes in normalization.