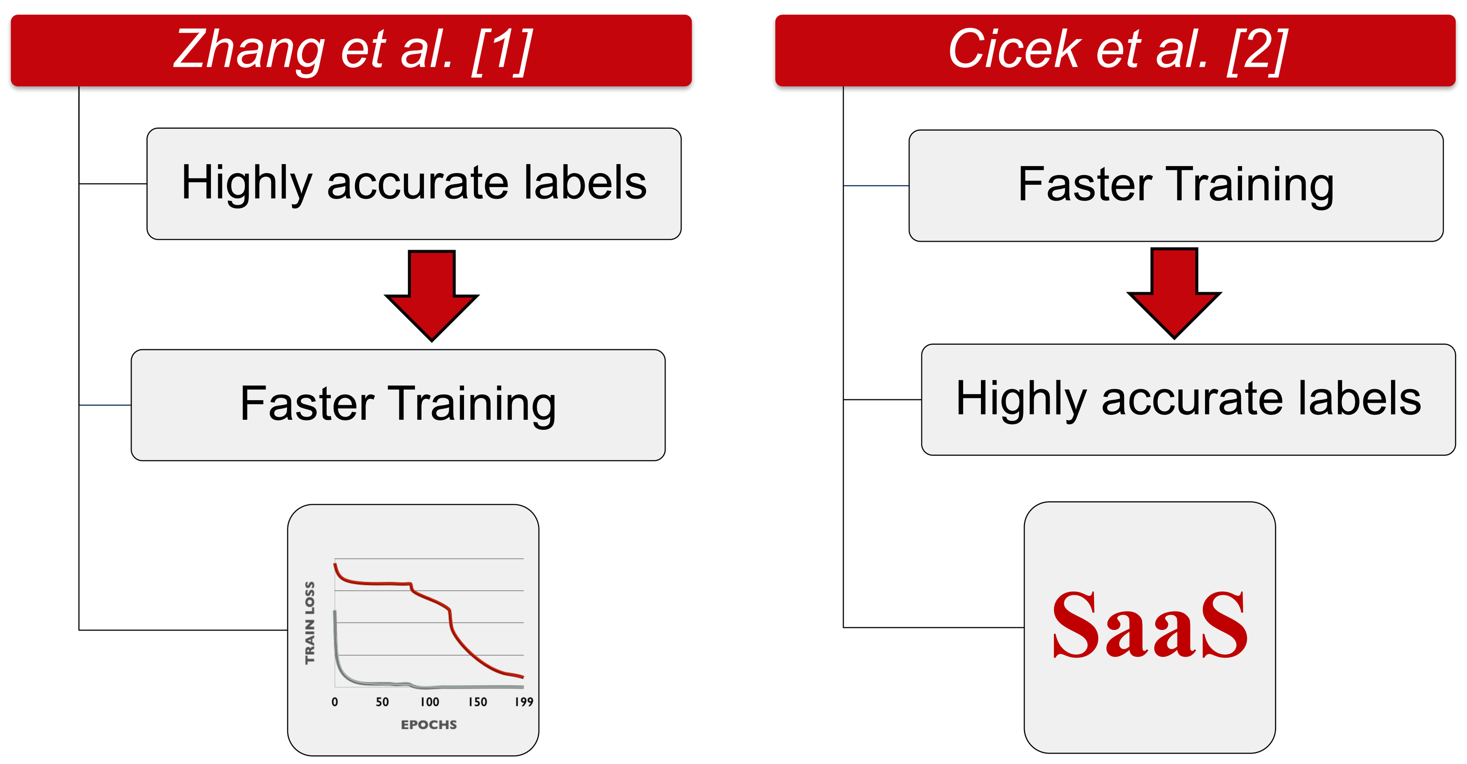
Substituting ReLUs with Hermite Polynomials gives faster convergence for SSL

Vishnu Lokhande, Sathya N. Ravi, Songwong Tasneeyapant, Abhay Venkatesh, Vikas Singh
University of Wisconsin-Madison

The Semi-supervised Learning Setup

We adopt a two phase procedure. In Phase I, we estimate the pseudolabels for the unlabeled images. In Phase II, using the estimated pseudolabels, we train a standard classifier in a supervised manner.



Speed as a Supervisor (SaaS)

SaaS seeks to find a set of pseudolabels that maximizes the decrease in the loss function over a small number of epochs.

Algorithm 1

Input: Labeled data (x_i, y_i), unlabeled data (z_i), number of classes k, #-outer (inner) epochs $M_O(M_I)$, loss function $L = L_{CE}(x_i, y_i) + L_{CE}(z_i, y_i) + Reg_E(z_i, y_i)$, learning rates $\eta_w, \eta_P^p, \eta_P^d$. Initial Pseudolabels for unlabeled data chosen as: $y_i = e_i$ with probability $1/k$ where e_i is the one-hot vector at $i-$th coordinate.

for $O = 0, 1, 2, ..., M_O$ **do**
  Reinitialize the network parameters w^0
  $\Delta P_u = 0$
  for $I = 0, 1, 2, ..., M_I$ **do**
    (Primal) SGD Step on w: $w^{t+1} \leftarrow w^t - \eta_w \nabla L$
    (Primal) SGD Step on ΔP_u: $\Delta P_u \leftarrow \Delta P_u - \eta_P^p \nabla L$
  end for
  (Dual) SGD Step on P_u: $P_u \leftarrow P_u - \eta_P^d \Delta P_u$
end for
Output: Classification model w.

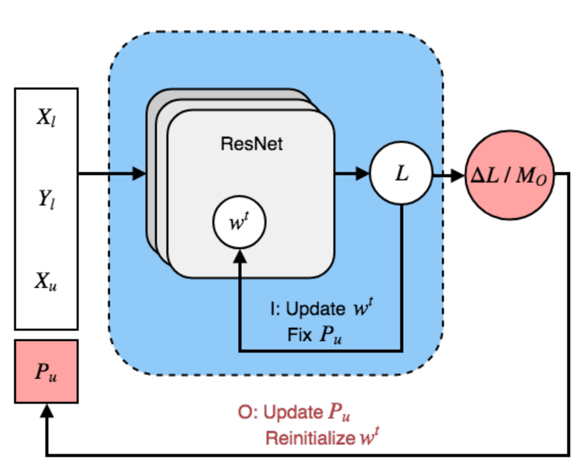


Figure 1: Illustration of the SaaS Framework. SaaS runs over two loops, in the inner loop denoted by I and the outer loop denoted by O

Conclusion: Smoothness of the objective helps in training SaaS faster

Gap in the Literature

- Ge et al. [3] showed that for one hidden layer network, one could avoid spurious local minima by utilizing an orthogonal basis expansion for ReLUs. This makes the optimization landscape well behaved.

Conclusion: Theoretical results not empirically investigated on regular computer vision architectures

- Nar et al. [4] showed that smoother landscapes enable the use of a larger range of step sizes in order for the gradient descent algorithm to converge.

Conclusion: Smoothness of the objective helps in convergence

Thanks for visiting!
See you again.

Hermite Polynomials

Hermite Polynomials as activations

- The lower order terms in the Hermite polynomial series expansion of ReLU is used as an activation function with the coefficients as trainable parameters.
- Optionally, a SoftSign function is added to handle large numerical values attained by the polynomials.

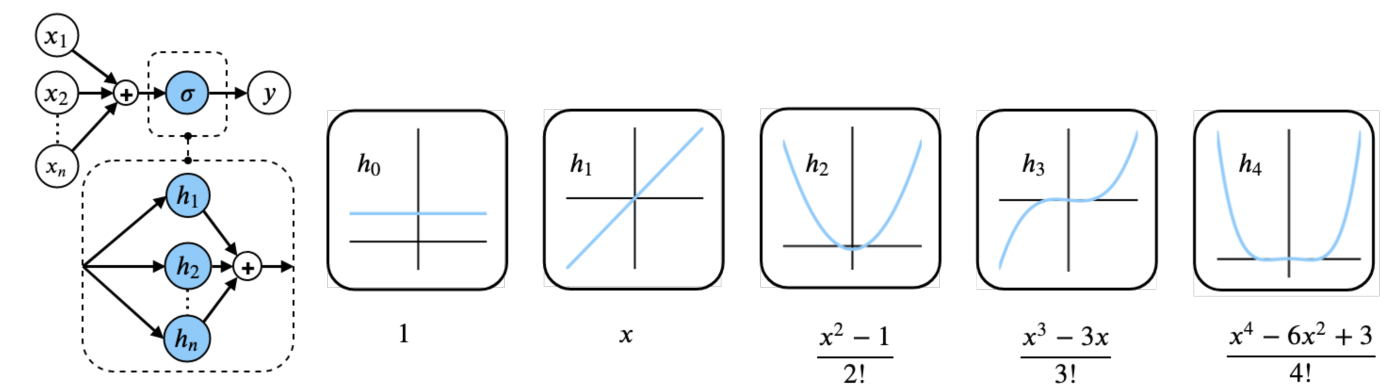


Figure 2: Hermite Polynomials as Activations **(Leftmost)**: Incorporating Hermite Polynomials as an activation function in a single hidden unit one hidden layer network.**(Middle)** The functional form of first 5 hermites.

Hermite Polynomials in ResNet 18

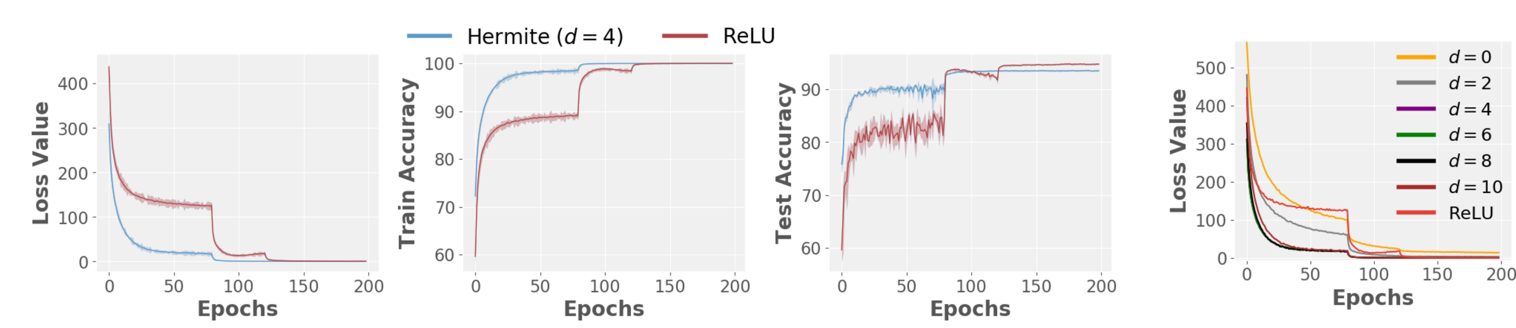


Figure 3: Hermites vs. ReLUs on ResNet18. **(Left 3 charts)** Hermites provide faster convergence of train loss and train accuracies than ReLUs. Hermites have faster convergence in test accuracies over the initial epochs but ReLU has the higher test accuracy at the end of training. **(Rightmost chart)** As we increase the number of hermite polynomials, the speed of loss convergence increases until d = 6 and then it starts to reduce. d ≥ 1 performs better than d = 0 where only softsign is used as an activation.

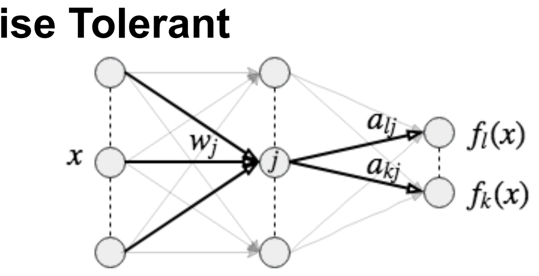
Hermite Polynomials in ResNet 152

Dataset CIFAR10	Number of Trainable Parameters	Best Test Accuracy	Epochs to reach 90% Test Accuracy
Hermite	58,145,574	95.48%	30
ReLU	58,144,842	94.5%	80

Table 1: Hermites vs. ReLUs on ResNet 152. We observe a small increase in the number of parameters. Test accuracy for hermite model converges in less than half the number of epochs.

Hermite Networks are Noise Tolerant

A generic perturbation bound is derived for for a one-hidden layer hermite network. The notations used are as outlined in the adjacent figure.



We use the perturbation bound to quantify noise resilience of Hermite polynomials. We show that if the test data is far from the train data, then hermite networks give low confidence predictions unlike ReLU network.

Theorem 1. Let $f_k(x) = \sum_j a_{kj} \sum_{i=0}^{d} c_i h_i(w_j^T x)$ be a one-hidden layer network with the sum of infinite series of hermite polynomials as an activation function. Here, $k = 1, 2, \dots, K$ are the different classes. Define $w_J = \min w_j^T x$. Let the data x be mean normalized. If $\epsilon > 0$, the Hermite coefficients $c_i = (-1)^i$ and $||x|| \geq \frac{1}{||w_J||} \log\left(\frac{\alpha}{\log(1+K\epsilon)}\right)$ then, we have that the predictions are approximately (uniformly) random. That is, $\frac{1}{K} - \epsilon \leq \frac{e^{f_k(x)}}{\sum_{l=1}^{K} e^{f_l(x)}} \leq \frac{1}{K} + \epsilon, \forall k \in \{1, 2, \dots, K\}$.

Computational Benefits

Hermite-SaaS trains faster

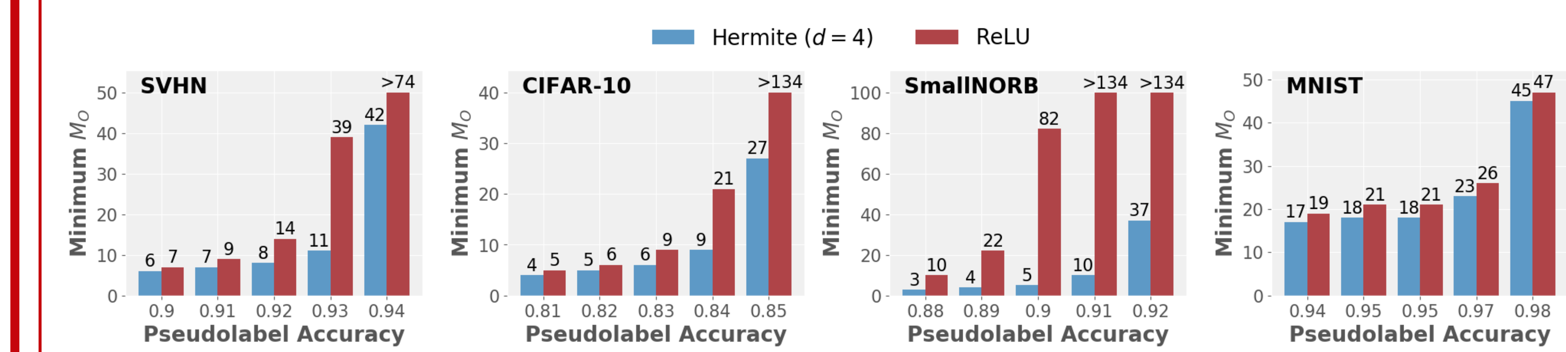


Figure 5: Hermite-SaaS trains faster. We plot the number of outer epochs M_O vs. the pseudolabel accuracy across 4 datasets. We consistently observe that the minimum number of outer epochs M_O to reach a given value of pseudolabel accuracy is always lower for Hermite-SaaS than ReLU-SaaS.

Hermite-SaaS saves time and money

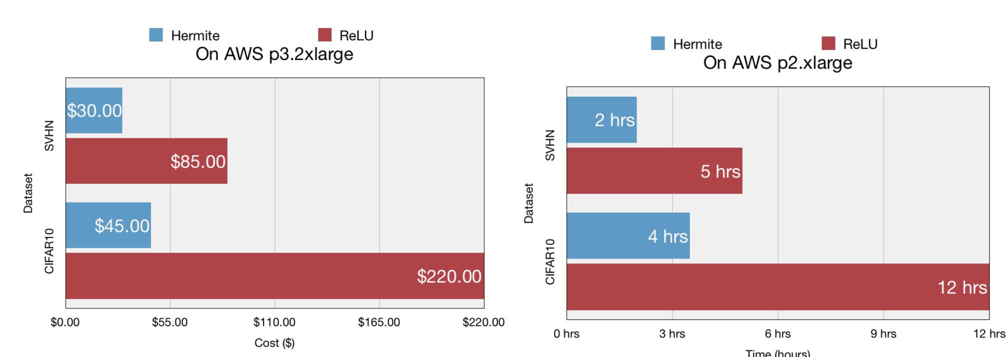


Figure 6: **(Left)** Hermite-SaaS is saving $$ on AWS p3.2xlarge. **(Right)** Hermite-SaaS is saving compute time on AWS p2.xlarge.

Hermite-SaaS generalizes better

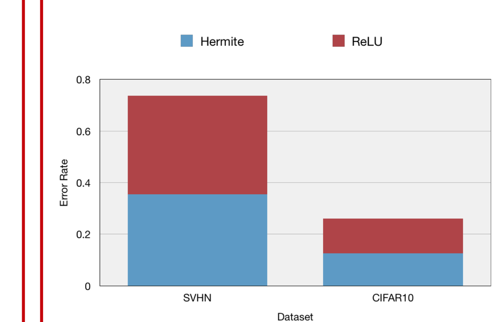


Figure 7: We obtain lower generalization error for the SSL setup.

Hermite-SaaS is more noise resilient

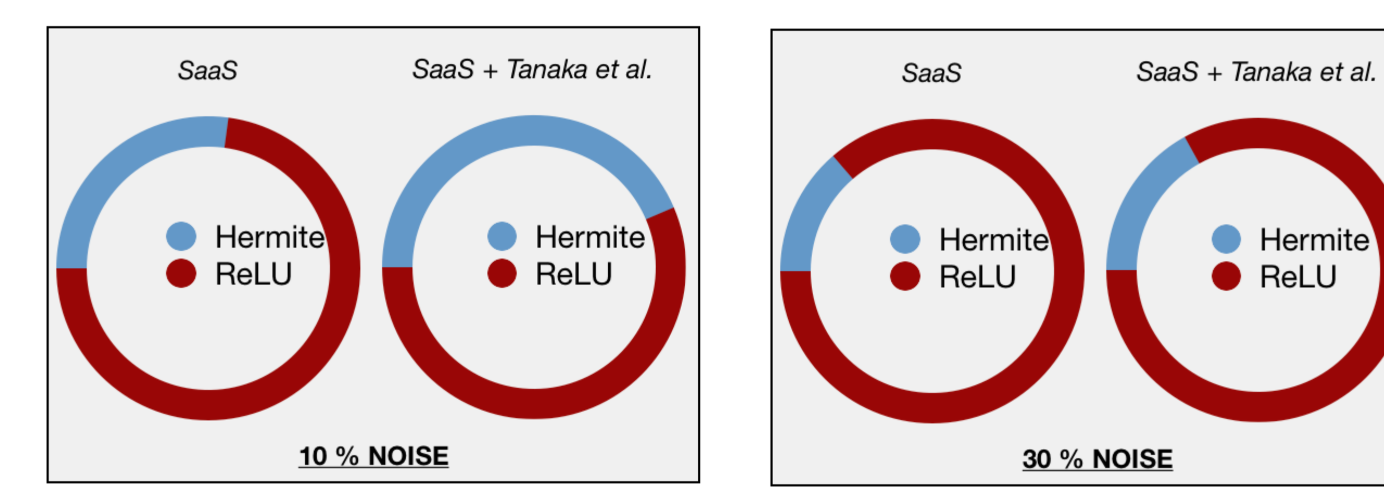


Figure 8: SaaS training with **(Left)**10% and **(Right)** 30% label corruption. We report the number of epochs to reach the best accuracy. We observe that Hermite-SaaS converges faster compared to ReLU-SaaS. Hermite activations yield estimators with low variance suggesting that they may behave well in the presence of outliers. Tanaka et al stands for noisy label processing method proposed in [5]. It is indicated from our experiments that post-processing techniques such as [5] may not always be useful from generalization perspective for an SSL setup.

References

[1] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals. Understanding deep learning requires rethinking generalization. arXiv preprint arXiv:1611.03530, 2016
[2] S. Cicek, A. Fawzi, and S. Soatto. Saas: Speed as a supervisor for semi-supervised learning. In The European Conference on Computer Vision (ECCV), September 2018
[3] R. Ge, J. D. Lee, and T. Ma. Learning one-hidden-layer neural networks with landscape design. arXiv:1711.00501, 2017
[4] Nar, K. and Sastry, S. Step size matters in deep learning. In Advances in Neural Information Processing Systems, 2018
[5] Tanaka, D. Ikami, T. Yamasaki, and K. Aizawa. Joint optimization framework for learning with noisy labels. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018