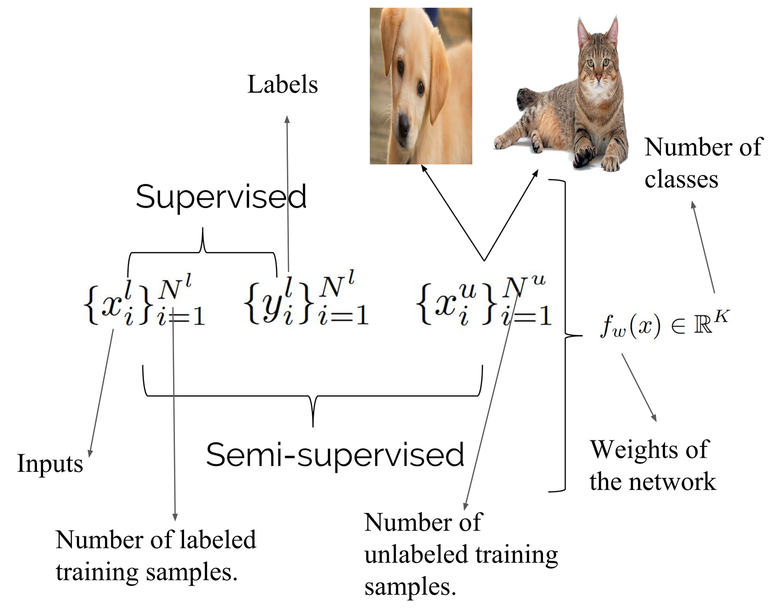
SaaS: Speed as a Supervisor for Semi-supervised Learning

Safa Cicek, Alhussein Fawzi, Stefano Soatto

UCLA VISION LAB

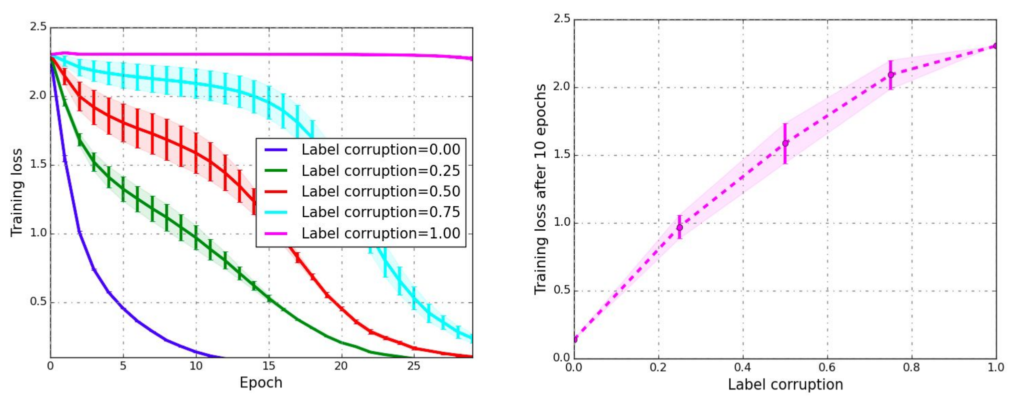
Semi-supervised learning (SSL)

- In SSL, one is given some labeled and unlabeled data.
- Goal: train a classifier, in hope of it performing better than if trained on the labeled data alone.



Motivation

- The key idea of our approach is to use speed of convergence as an inference criterion for the value of the unknown labels for SSL.



- **Supervision quality correlates to learning speed.**

Cumulative loss

- To quantify learning speed, we use the cumulative loss in a fixed time (epoch) interval:

$$\mathcal{L}_T = \frac{1}{T}\sum_{t=1}^{T}\frac{1}{|B_t|}\sum_{i=1}^{|B_t|} \underbrace{-\langle \log f_{w_t}(x_i), P_i\rangle}_{\ell(x_i, P_i; w_t)}$$

Cardinality of the mini-batch

$f_{w_t}(x_i)[k] \simeq P(y_i = k|x_i)$

$P_i = \delta_{i,k}$

Weights simulated with SGD dynamics.

Cumulative loss as a criterion for posterior

- Cumulative loss can be written as a function of unknown label posterior to be used as a criterion.

$$\mathcal{L}_T(P^u) = \frac{1}{T}\sum_{t=1}^{T}\frac{1}{|B_t^u|}\sum_{i=1}^{|B_t^u|} \underbrace{-\langle \log f_{w_t}(x_i^u), P_i^u\rangle}_{\ell(x_i^u, P_i^u; w_t)}$$

$P^u \in \mathbb{R}^{N^u \times K}$

$P_i^u[k] = P(y_i = k|x_i), \; k = 1, \dots, K$

Overall optimization

- The overall learning can be framed as the following optimization.

$$P^u = \arg\min_{P^u} \frac{1}{T}\sum_{t=1}^{T} \underbrace{\ell(B_t^u, P^u; w_{t-1})}_{\frac{1}{|B_t^u|}\sum_{i=1}^{|B_t^u|}\ell(g_i(x_i^u), P_i^u; w_{t-1})}$$

Set of samples in mini-batch

$$\begin{aligned} P^u = &\arg\min_{P^u}\frac{1}{T}\sum_{t=1}^{T}\ell(B_t^u, P^u; w_{t-1}) \\ \text{subject to } & w_{t-\frac{1}{2}} = w_{t-1} - \eta_w \nabla_{w_{t-1}}\left(\ell(B_t^u, P^u; w_{t-1})\right) \\ & w_t = w_{t-\frac{1}{2}} - \eta_w \nabla_{w_{t-\frac{1}{2}}}\ell(B_t^l, P^l; w_{t-\frac{1}{2}}) \;\forall\, t = 1 \dots T \\ & P^u \in \mathcal{S} \end{aligned}$$

Can we just minimize cumulative loss and get correct labels?

- Supervision quality correlates with learning speed *in expectation* not in every realization.

Avoiding degenerate solutions

- Weights trained with unknown labels should have almost zero training loss on (augmented) labeled data.
- Posterior of label estimates should live in probability simplex.
- Cumulative loss should be small for augmented unlabeled data.

Entropy as an additional regularizer

- Minimizing the entropy of label estimates on unlabeled data is common in SSL literature.

$$H_Q(w) = \sum_{i=1}^{N^u} \underbrace{-\langle f_w(x_i^u), \log f_w(x_i^u)\rangle}_{q(x_i^u; w)}$$

Overall optimization with entropy

- Minimizing the entropy of label estimates on unlabeled data is common in SSL literature.

$$\begin{aligned} P^u = &\arg\min_{P^u}\frac{1}{T}\sum_{t=1}^{T}\ell(B_t^u, P^u; w_{t-1}) \\ \text{subject to } & w_{t-\frac{1}{2}} = w_{t-1} - \eta_w \nabla_{w_{t-1}}\left(\ell(B_t^u, P^u; w_{t-1}) + \beta q(B_t^u; w_{t-1})\right) \\ & w_t = w_{t-\frac{1}{2}} - \eta_w \nabla_{w_{t-\frac{1}{2}}}\ell(B_t^l, P^l; w_{t-\frac{1}{2}}) \;\forall\, t = 1 \dots T \\ & P^u \in \mathcal{S} \end{aligned}$$

Algorithm

- In the beginning of each outer epoch, label estimates are projected to the probability simplex; the posterior initialized randomly.
- The inner loop performs a few epochs of SGD to measure learning speed (cumulative loss) while keeping the label posterior fixed.
- The outer loop then applies a gradient step to update the unknown-label posterior. After each update, the weights are reset.
- After the label posterior converges, we select the maximum a-posteriori estimate and proceed with training as if fully supervised in the second phase.

$P^u \sim \mathcal{N}(0, I)$
Select learning rates η for the weights η_w and label posteriors η_{P^u}
Phase I: Estimate P^u
while P^u has not stabilized **do**
  $P^u = \Pi(P^u)$ (project posterior onto the probability simplex)
  $w_1 \sim \mathcal{N}(0, I)$
  $\Delta P^u = 0$
  // Run SGD for T steps (on the weights) to estimate loss decrease
  for $t = 1 : T$ **do**
    $w_{t-\frac{1}{2}} = w_{t-1} - \eta_w \nabla_{w_{t-1}}\left(\ell(B_t^u, P^u; w_{t-1}) + \beta q(B_t^u; w_{t-1})\right)$
    $w_t = w_{t-\frac{1}{2}} - \eta_w \nabla_{w_{t-\frac{1}{2}}}\ell(B_t^l, P^l; w_{t-\frac{1}{2}})$
    $\Delta P^u = \Delta P^u + \nabla_{P^u}\ell(B_t^u, P^u; w_t)$
  // Update the posterior distribution
  $P^u = P^u - \eta_{P^u}\Delta P^u$
Phase II: Estimate the weights.
$\hat{y}_i^u = \arg\max_i P_i^u \;\forall i = 1, \dots, N^u$
$w_1 \sim \mathcal{N}(0, I)$
while w has not stabilized **do**
  $w_{t-\frac{1}{2}} = w_{t-1} - \eta_w \nabla_{w_{t-1}} \frac{1}{|B_t^u|}\sum_{i=1}^{|B_t^u|}\ell(x_i^u, \hat{y}_i^u; w_{t-1})$
  $w_t = w_{t-\frac{1}{2}} - \eta_w \nabla_{w_{t-\frac{1}{2}}} \frac{1}{|B_t^l|}\sum_{i=1}^{|B_t^l|}\ell(x_i^l, y_i^l; w_{t-\frac{1}{2}})$

Weights are not learned in the first phase of SaaS.

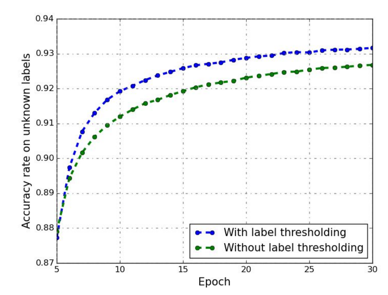
$$\min_{w, P^u}\sum_{i=1}^{N}\ell(x_i, P_i^u; w)$$

- This optimization problem has many trivial, degenerate solutions (Zhang et al., 2016). In SaaS, label posterior minimizing the *cumulative loss* is found. Weights of the network are not learnable parameters in the first phase of the SaaS; they are *simulated* with SGD dynamics.

Label thresholding on posterior

- We project label estimates to the closest probability simplex with minimum probability for a class being 0.05.

$$\mathcal{S}_\alpha = \{x \in \mathbb{R}^K : \sum_i x_i = 1, x_i \geq \alpha\}$$



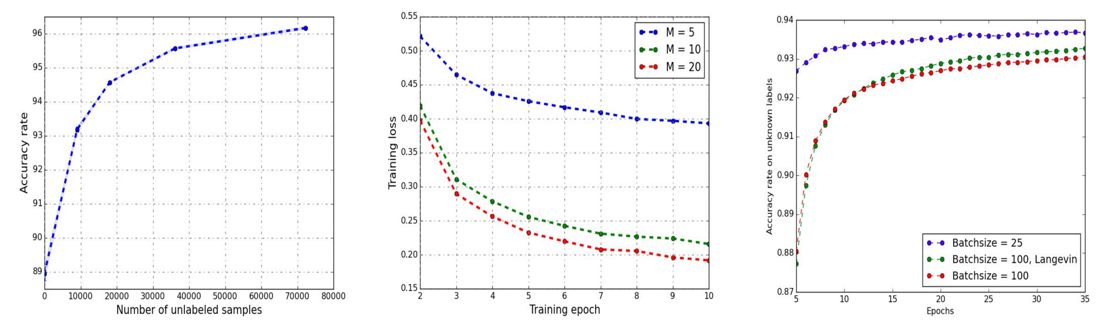
Results

- Error rates achieved by SaaS.

Dataset	CIFAR10-4k	SVHN-1k
Error rate by supervised baseline on test data	17.64 ± 0.58	11.04 ± 0.50
Error rate by SaaS on unlabeled data	12.81 ± 0.08	6.22 ± 0.02
Error rate by SaaS on test data	10.94 ± 0.07	3.82 ± 0.09

- Comparison of SaaS to other state-of-the-art SSL algorithms.

Method-Dataset	CIFAR10-4k	SVHN-1k
VAT+EntMin Miyato et al. (2017)	**10.55**	3.86
Stochastic Transformation Sajjadi et al. (2016)	11.29	NR
Temporal Ensemble Laine and Aila (2016)	12.16	4.42
GAN+FM Salimans et al. (2016)	15.59	5.88
Mean Teacher Tarvainen and Valpola (2017)	12.31	3.95
SaaS	10.94 ± 0.07	**3.82 ± 0.09**



- (Left) SaaS achieves better generalization with more unlabeled data.
- (Middle) SaaS finds labels training on which is faster.
- (Right) By using smaller batch size, SaaS achieves better generalization with the cost of low GPU utilization and slow training. A trick we use to improve the computational cost is to use Langevin dynamics with larger batchsize.

References

- Miyato, T., Maeda, S.i., Koyama, M., Ishii, S.: Virtual adversarial training: a regularization method for supervised and semi-supervised learning. arXiv preprint arXiv:1704.03976 (2017)
- Sajjadi, M., Javanmardi, M., Tasdizen, T.: Mutual exclusivity loss for semi- supervised deep learning. In: Image Processing (ICIP), 2016 IEEE International Conference on, IEEE (2016) 1908-1912
- Laine, S., Aila, T.: Temporal ensembling for semi-supervised learning. arXiv preprint arXiv:1610.02242 (2016)
- Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., Chen, X.: Improved techniques for training gans. In: Advances in Neural Information Processing Systems. (2016) 2234-2242
- Tarvainen, A., Valpola, H.: Mean teachers are better role models:Weight-averaged consistency targets improve semi-supervised deep learning results. In: Advances in neural information processing systems. (2017) 1195-1204
- Zhang, C., Bengio, S., Hardt, M., Recht, B., Vinyals, O.: Understanding deep learning requires rethinking generalization. arXiv preprint arXiv:1611.03530 (2016)