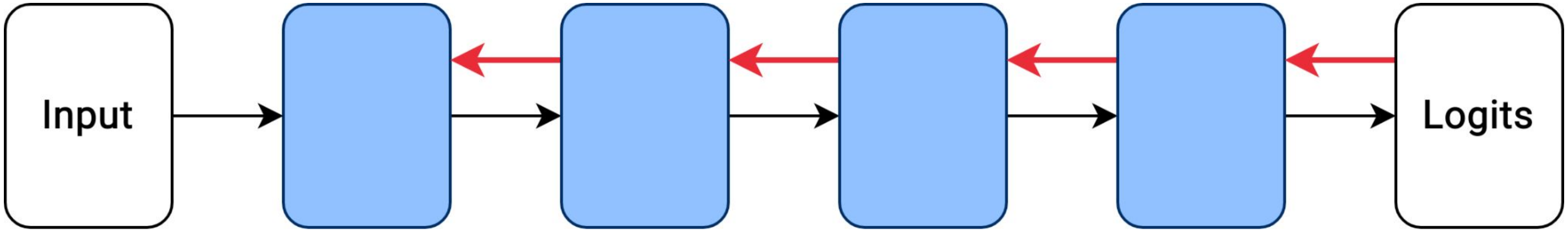
Improving the Forward-Forward Algorithm

Thomas Dooms

Prof. Dr. José M. Oramas
Dr. Inton Tsang

Backpropagation (BP)



Backpropagation

The most important discovery in machine learning?

Rummelhart David et al. (1986)

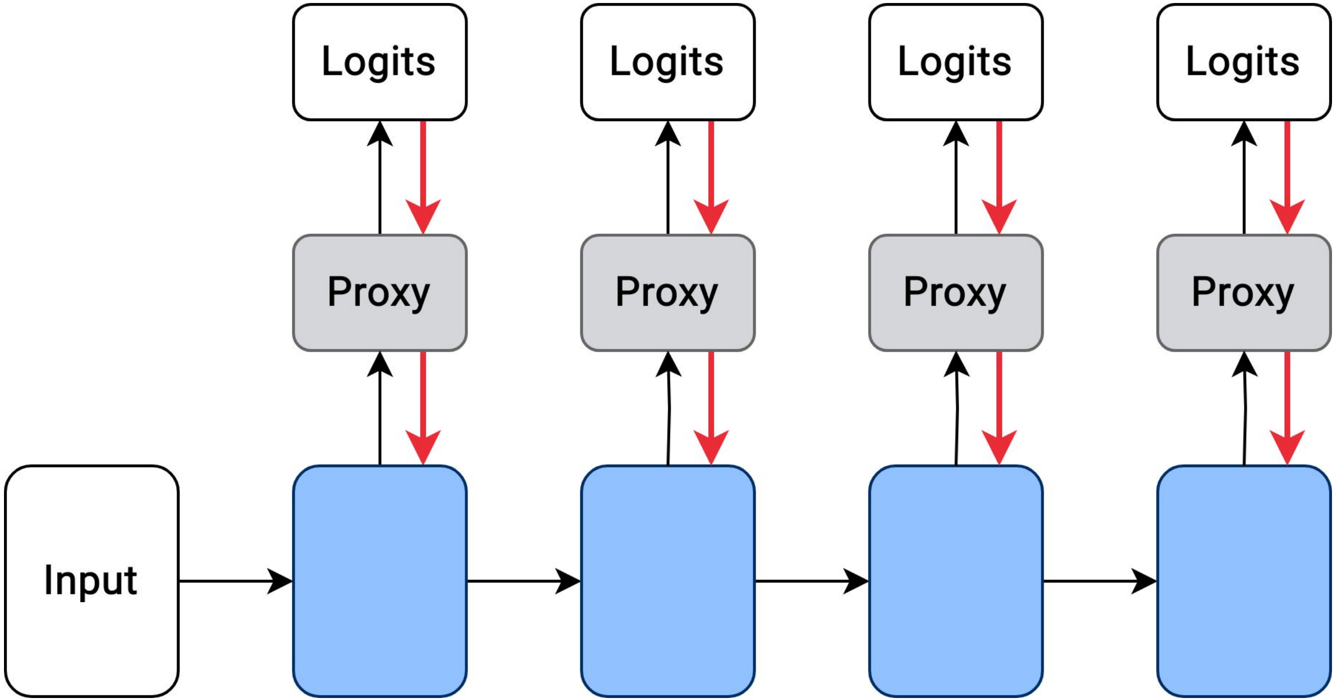
How it works

- Full error signal
- Chain rule

Downsides

- Vanishing or exploding gradients
- Fully differentiable network
- Hard to scale on hardware

Local Classifier Proxy (LCP)



Local Classifier Proxy

Duan Shiyu et al. (2021)

Notable improvements

- No update locking
- Non-differentiable operations between modules
- Easy scaling

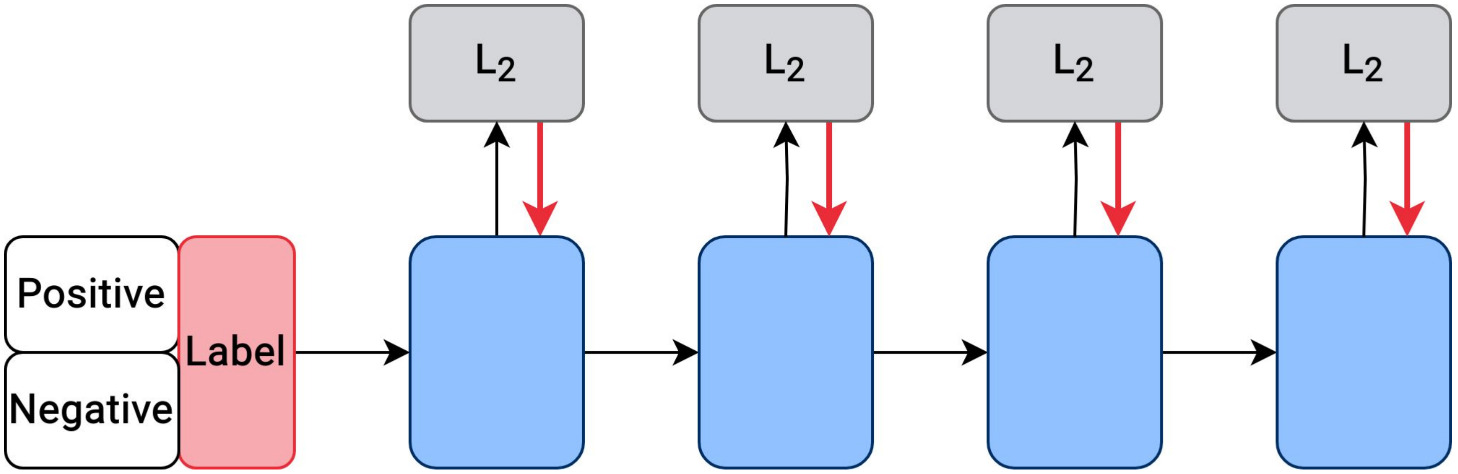
Downsides

- At evaluation time, proxies are useless
- Still need differentiation in proxy
- No error signal

Considerations

- Downstream usefulness of representations

Forward-Forward (FF)



Forward-Forward

What are the drawbacks of this learning algorithm?

Hinton Geoffrey (2022)

Notable improvements

- No update locking
- Can use non-differentiable operations
- Easy scaling

Downsides

- Difficult to evaluate / train
- No error signal

Considerations

- Downstream usefulness of representations

Goal of this Thesis

What did we set out to achieve?

Understand the layerwise relations of local learning

- Are the current representations useful?
- Are they useful downstream?

Improve these characteristics (for FF)

- Not simply improving accuracy
- But also improving depth scaling

Evaluation

How is the difference evaluated

Task

Classification of Cifar-10

Cifar-10

60,000 RGB images (32x32)

Natural images consisting of 10 classes

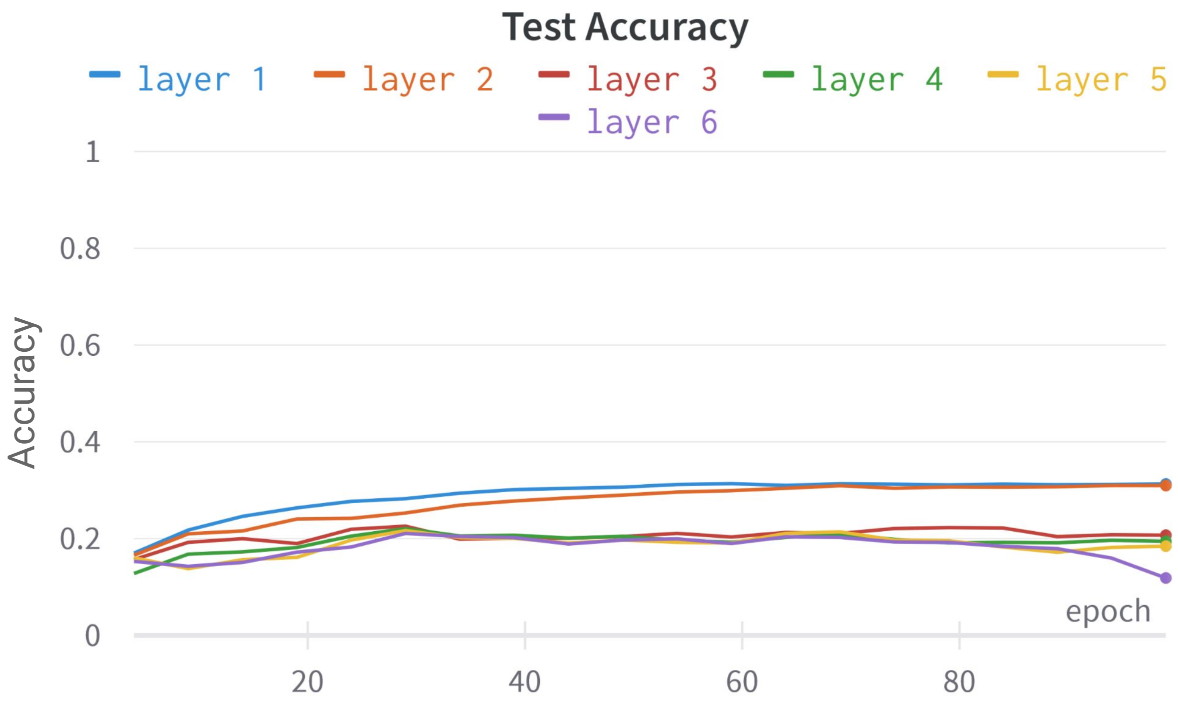
Metrics

Accuracy: $\alpha_i = \mathbb{E}_{s \in D}\left[\hat{y}_i = y\right]$

Delta: $\Delta_i = ||positive_i||_2^2 - ||negative_i||_2^2$

University of Antwerp IDLab INTERNET & DATA LAB

Forward-Forward Accuracies



Cifar-10 classification

The Trifecta

What is the main contribution of this thesis?

Primary components

- Improved loss function
- Implicit error signals
- Revisiting normalisation

Improved Loss Function

The first component of the Trifecta

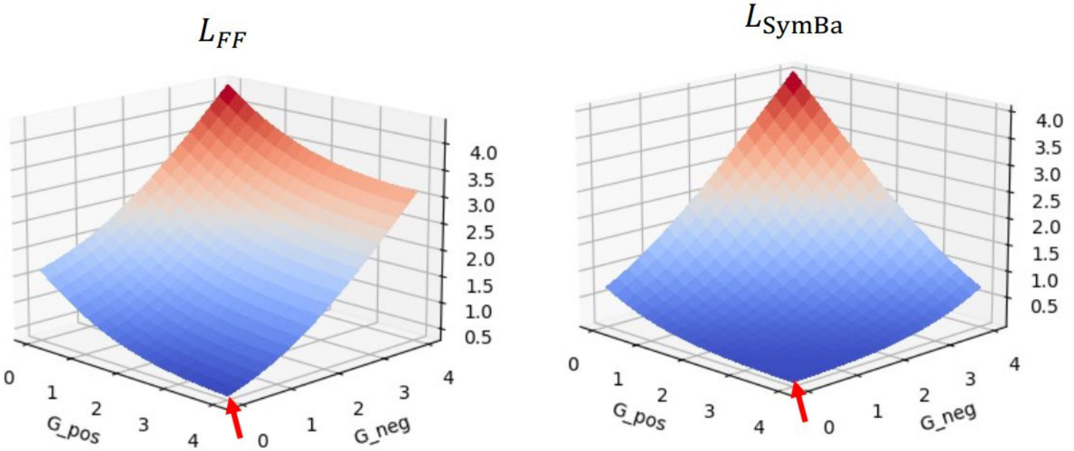
Chang-Heung Lee et al. (2023)

Issues with the vanilla loss function

- Asymmetry causes unstable gradients
- “Unnecessary” hyperparameter

Vanilla: $log(1 + exp(||negative||_2^2 - t)) + log(1 + exp(t - ||positive||_2^2))$

Symba: $log(1 + exp(||positive||_2^2 - ||negative||_2^2))$



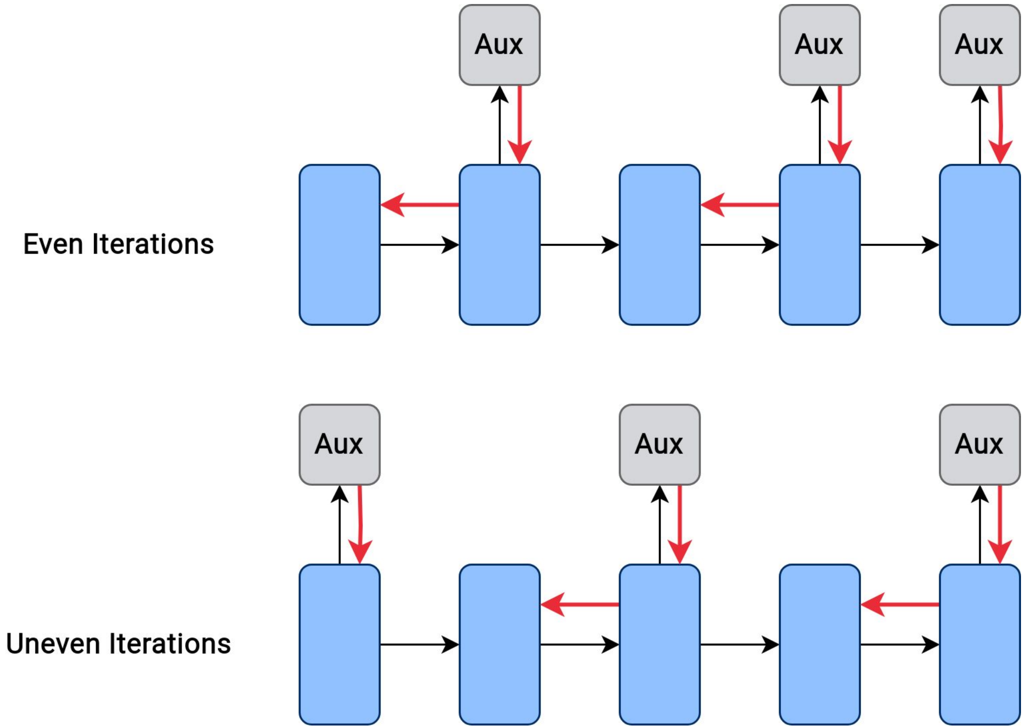
University of Antwerp

IDLab INTERNET & DATA LAB

Alternated Pairwise Learning

The second component of the Trifecta

Xiong Yuwen et al. (2020)



Ordinary Normalisation

The third component of the Trifecta

Assumption (by the vanilla FF algorithm)

Subsequent layers echo the separation

Normalize length of the feature vector

No information ‘leakage’ to next layer

Rather

Subsequent layer struggle to re-separate the data

Therefore

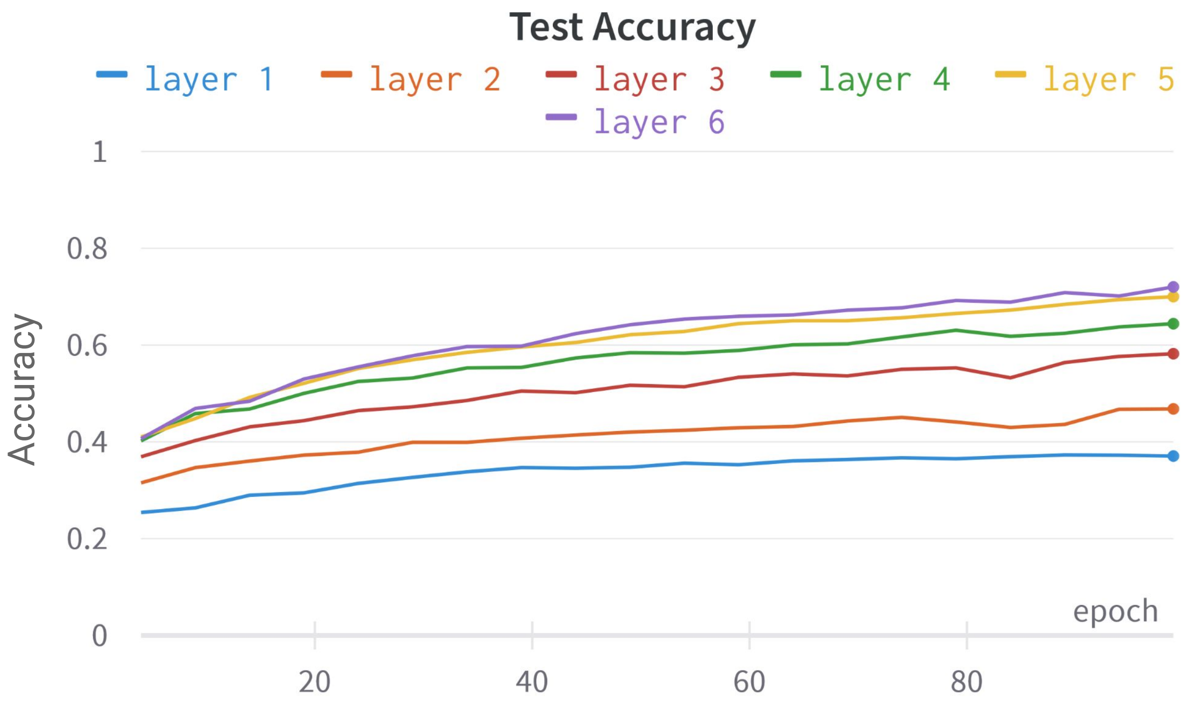
Use batch norm instead of layer norm

Results

What did we achieve?

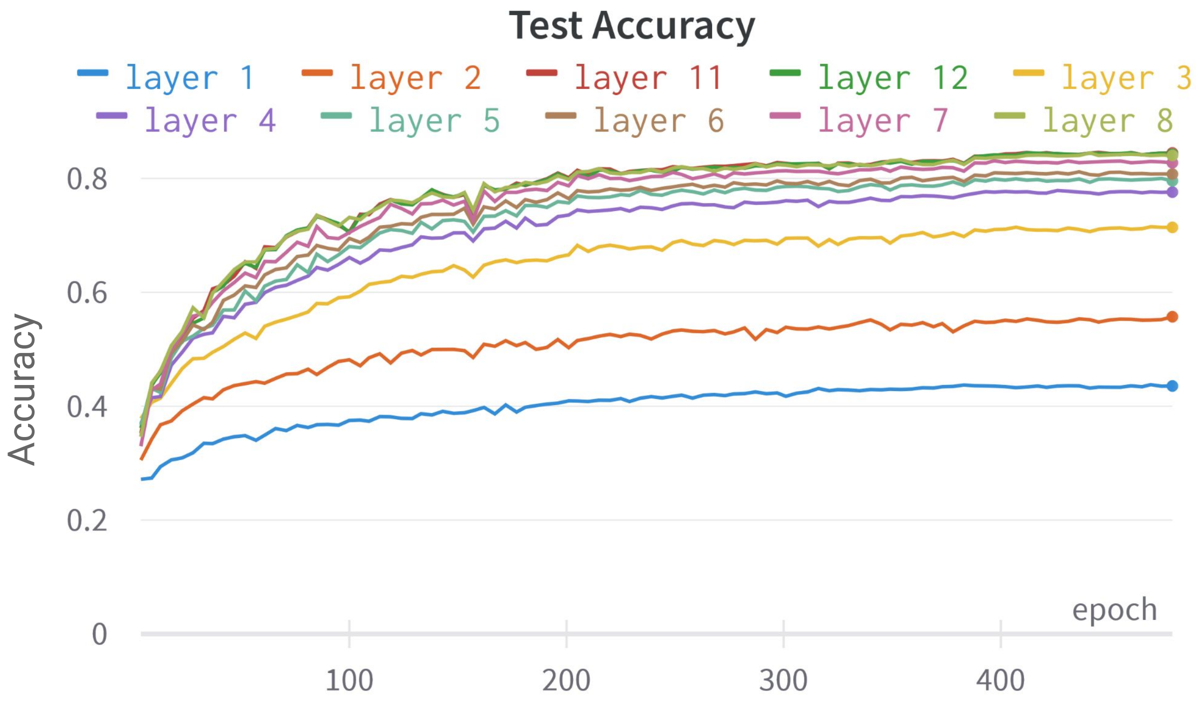
Name	Accuracy
Vanilla (reported)	59.0%
Vanilla (recreation)	34.3%
Symba	55.7%
Symba + APL	61.4%
Symba + BN	64.7%
The Trifecta	74.0%

Layerwise performance



Cifar-10 classification

Full convergence



Cifar-10 classification

Full Convergence Details

Only a single configuration was tested

500 epoch training regime

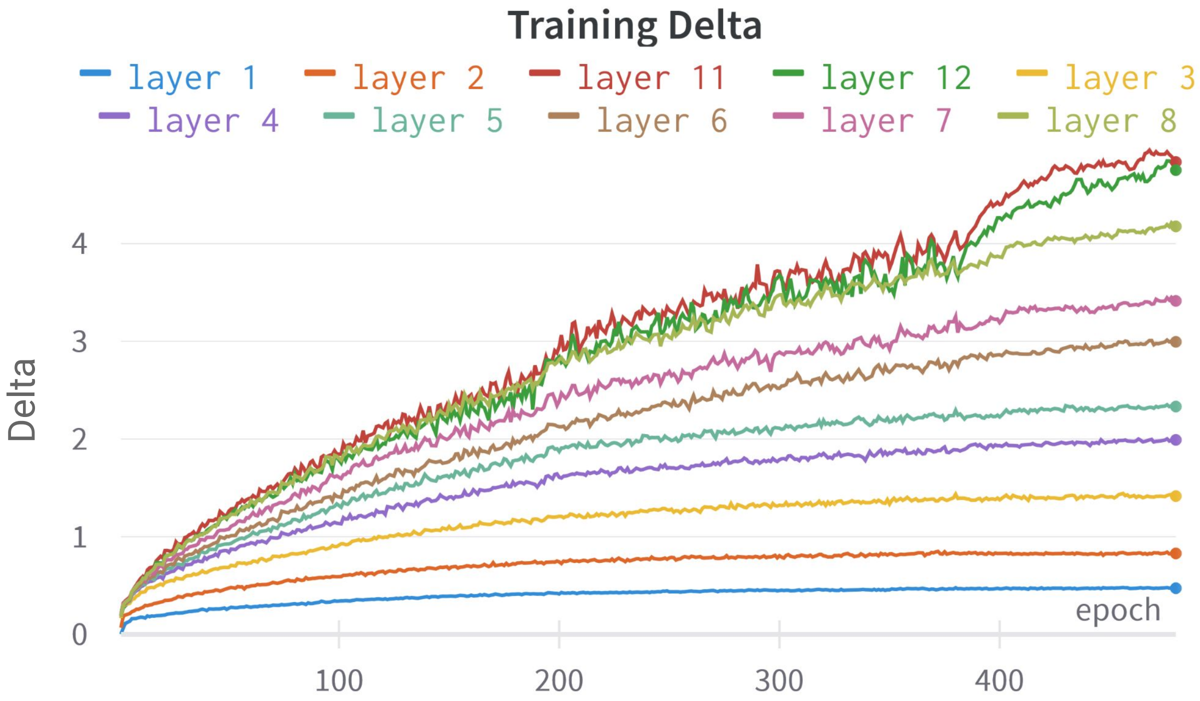
- 200 epochs lr 0.001 (0.0% → 80.0%)
- 200 epochs lr 0.0005 (80.0% → 83.3%)
- 100 epochs lr 0.0001 (83.3% → 84.3%)

Large network

- 12 layers (8.5M params)
- Layer 12 is 4% better than layer 6

	MNIST	F-MNIST	SVHN	CIFAR-10
FF (100)	98.3%	85.0%	91.8%	74.0%
FF (500)	**99.1%**	91.1%	**94.4%**	84.3%
BP (100)	**99.1%**	**93.6%**	94.1%	**88.9%**

Data separation



Cifar-10 classification

Conclusion

What have we learnt?

The Trifecta significantly improves FF

- The loss functions facilitates convergence
- APL improves current representations
- BN significantly helps future usefulness

Additional observations

- Our architecture scales to ~10 layers
- Lowering the learning rate aids later layers
- It still converges quite slowly

Short-Term Perspective

How can this algorithm be used immediately?

Non-differentiable networks

- Spiking neural networks
- Quantum networks
- Reinforcement learning (?)

Research curiosity

- Are these networks more explainable?
- Are they able to achieve high performance?
- How do these networks scale even further?

Additional theoretic analysis

Long-Term Perspective

What are the future prospects of this algorithm?

Efficient Hardware Implementations

- Solve simple tasks very efficiently
- Both in terms of power and memory

Less communication overhead

- Solve multi-accelerator tasks without overhead
- More flexible scaling of large models

University of Antwerp IDLab INTERNET & DATA LAB

Questions

References

Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification
He Kaiming et al. (2015)

Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift
Sergey Ioffe et al. (2015)

Deep Residual Learning for Image Recognition
He Kaiming et al. (2015)

Averaging Weights Leads to Wider Optima and Better Generalization
Izmailov Pavel et al. (2018)

A Simple Weight Decay Can Improve Generalization
Krogh Anders et al. (1991)

Learning representations by back-propagating errors
Rummerhart David et al. (1986)

Training Deep Architectures Without End-to-End Backpropagation: A Survey on the Provably Optimal Methods
Duan Shiyu et al. (2021)

GPipe: Efficient Training of Giant Neural Networks using Pipeline Parallelism
Huang Yanping et al. (2019)

SymBa: Symmetric Backpropagation-Free Contrastive Learning with Forward-Forward Algorithm for Optimizing Convergence
Chang-Heung Lee et al. (2023)

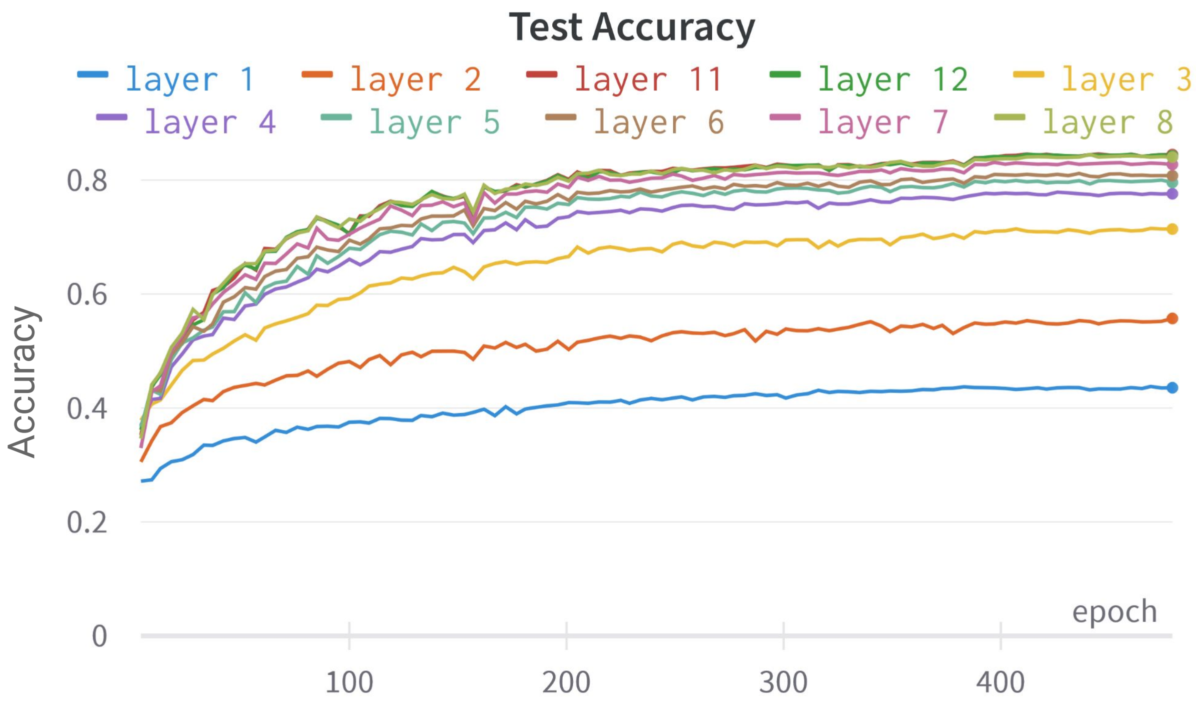
Dropout: A Simple Way to Prevent Neural Networks from Overfitting
Srivastava Nitish et al. (2014)

Visualizing the Loss Landscape of Neural Nets
Li Hao et al. (2018)

The forward-forward algorithm: Some preliminary investigations
Hinton Geoffrey (2022)

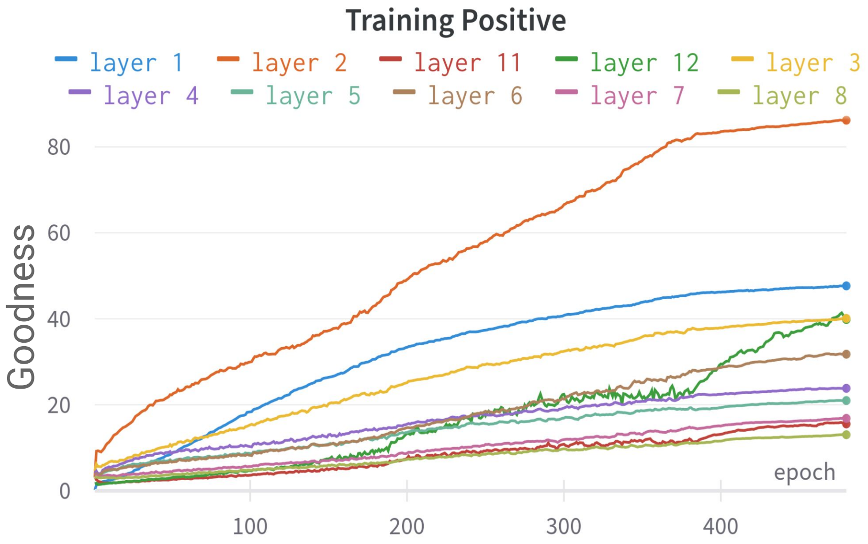
LoCo: Local Contrastive Representation Learning
Xiong Yuwen et al. (2020)

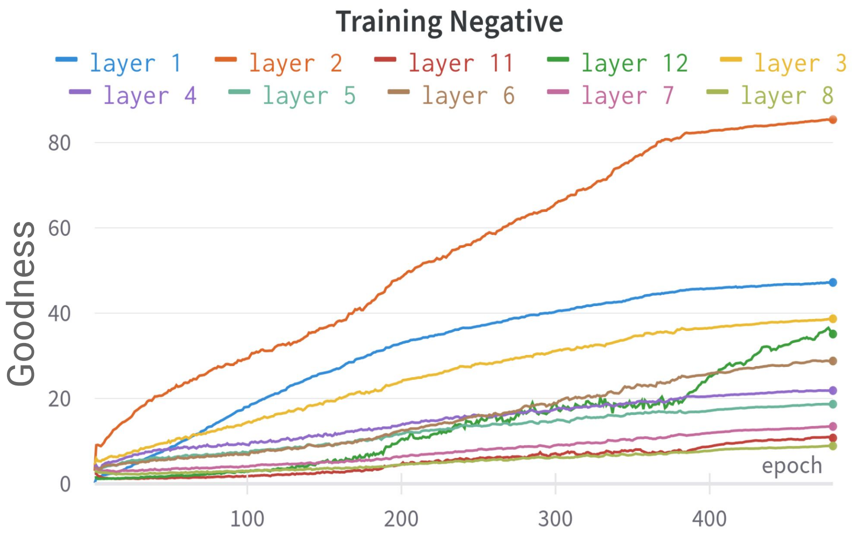
Internal Covariate Shift?



Cifar-10 classification

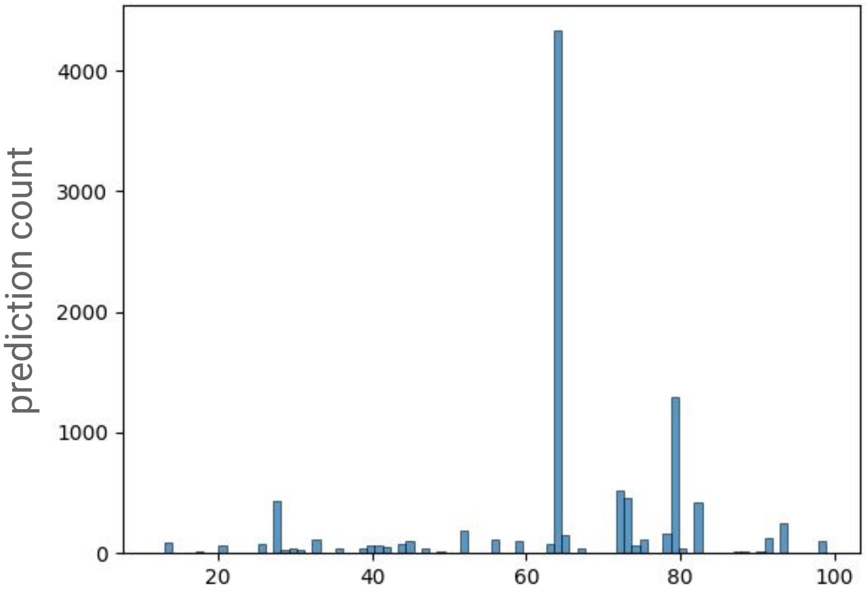
Learning rate adaption?

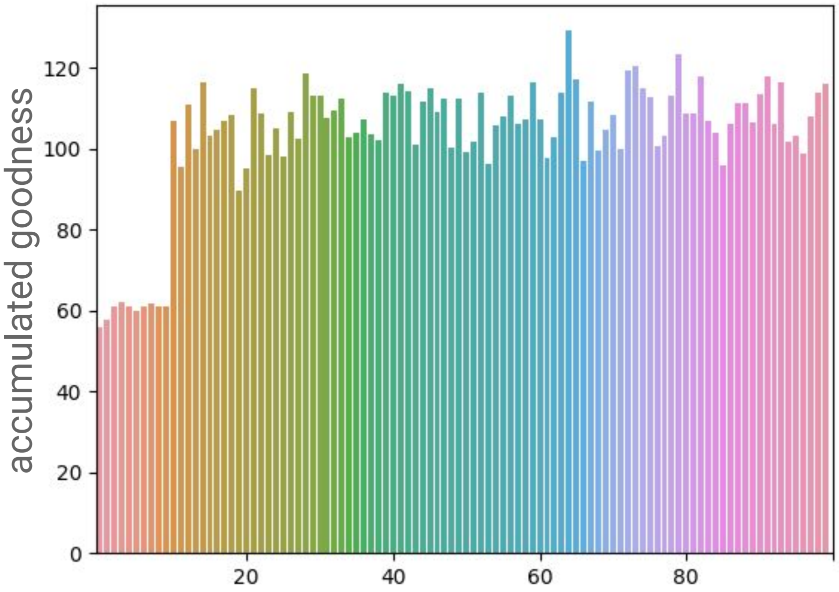




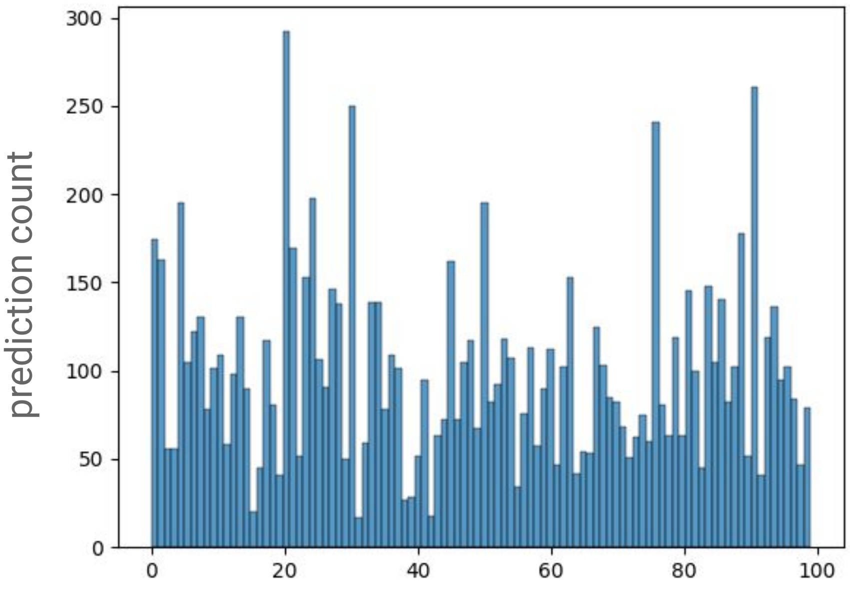
Cifar-10 classification

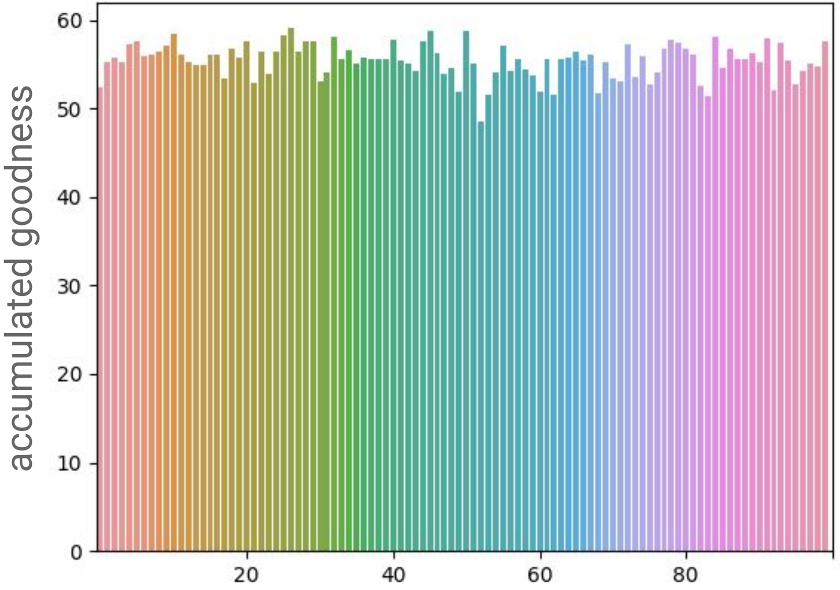
Larger Datasets?



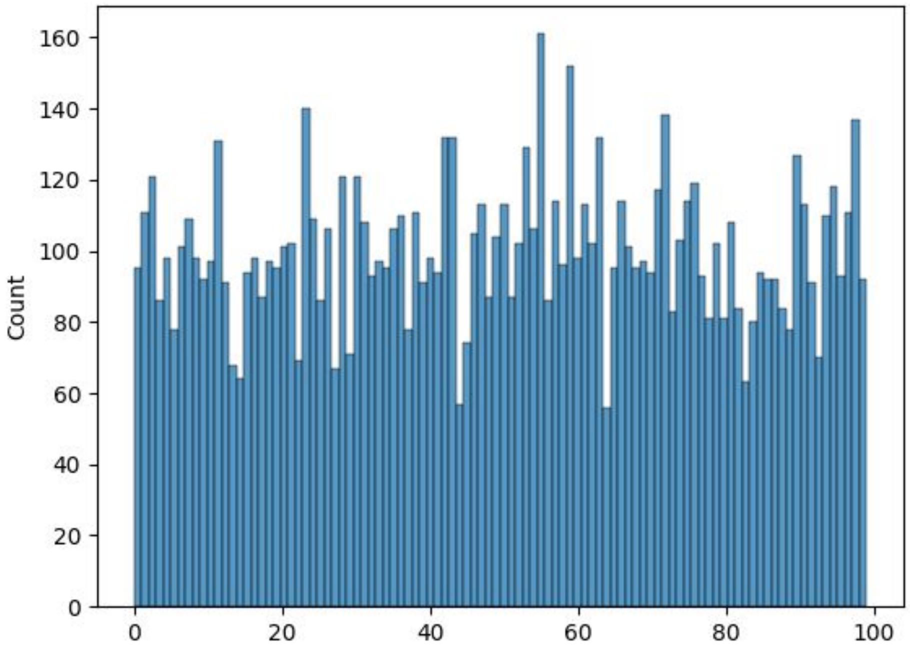


Larger Datasets!

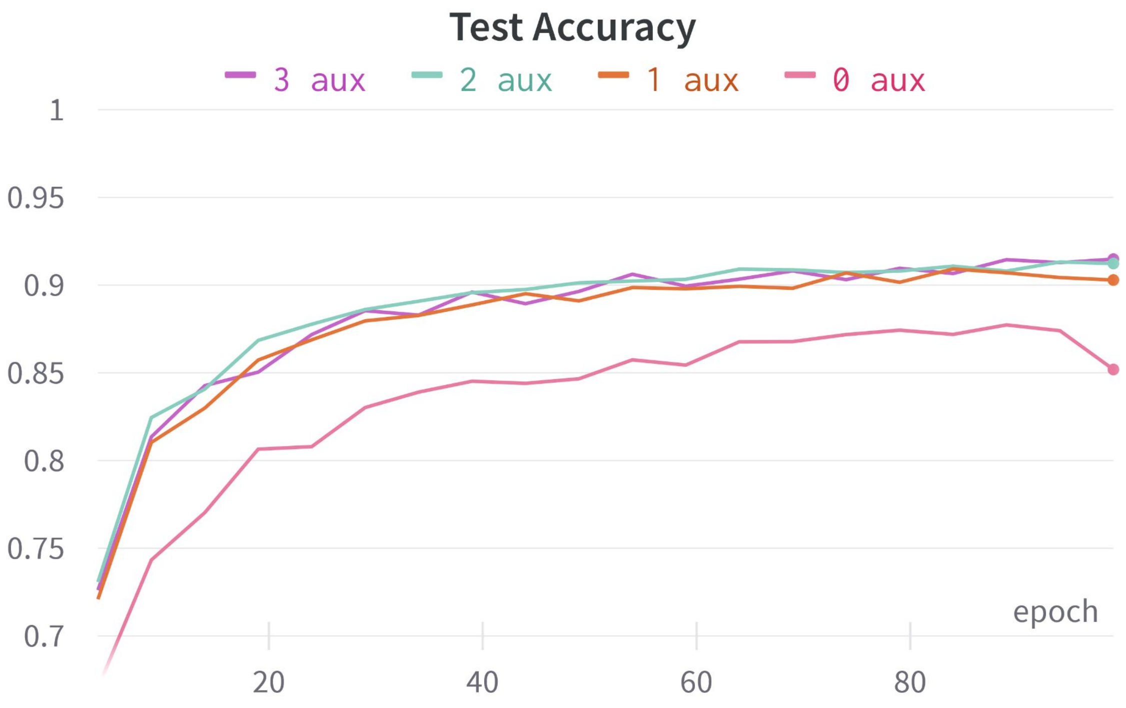




Backprop baseline



Local Classifier Proxy



Backpropagation