南京航空航天大學

Nanjing University of Aeronautics and Astronautics

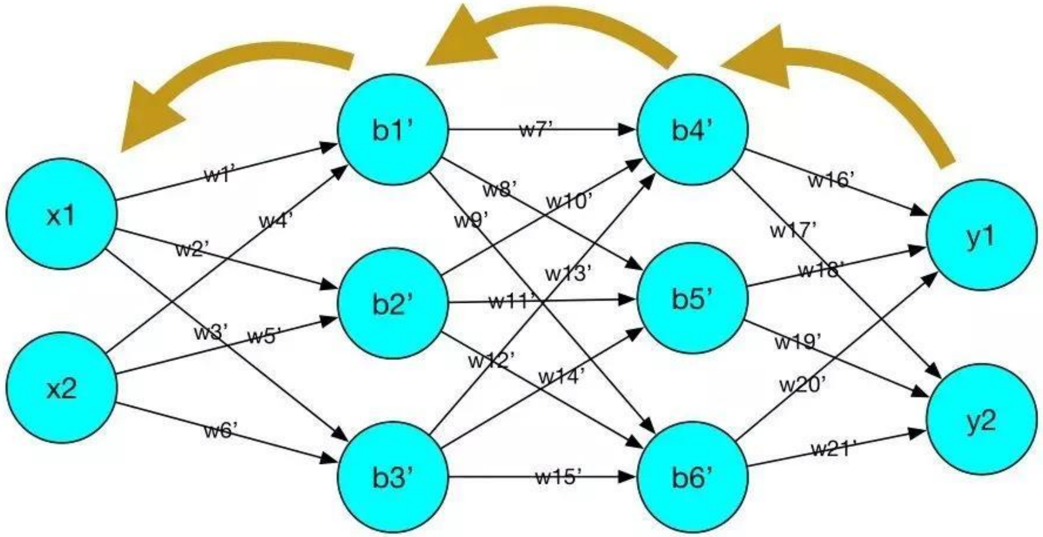
Putting An End to End-to-End: Gradient-Isolated Learning of Representations

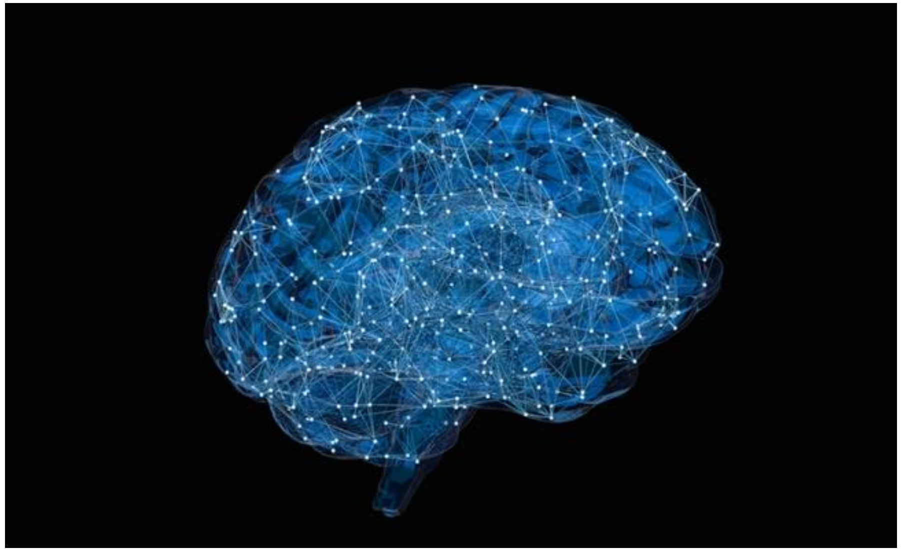
Sindy Löwe*   Peter O’Connor   Bastiaan S. Veeling*

AMLab
University of Amsterdam
loewe.sindy@gmail.com, basveeling@gmail.com

NeurIPS 2019

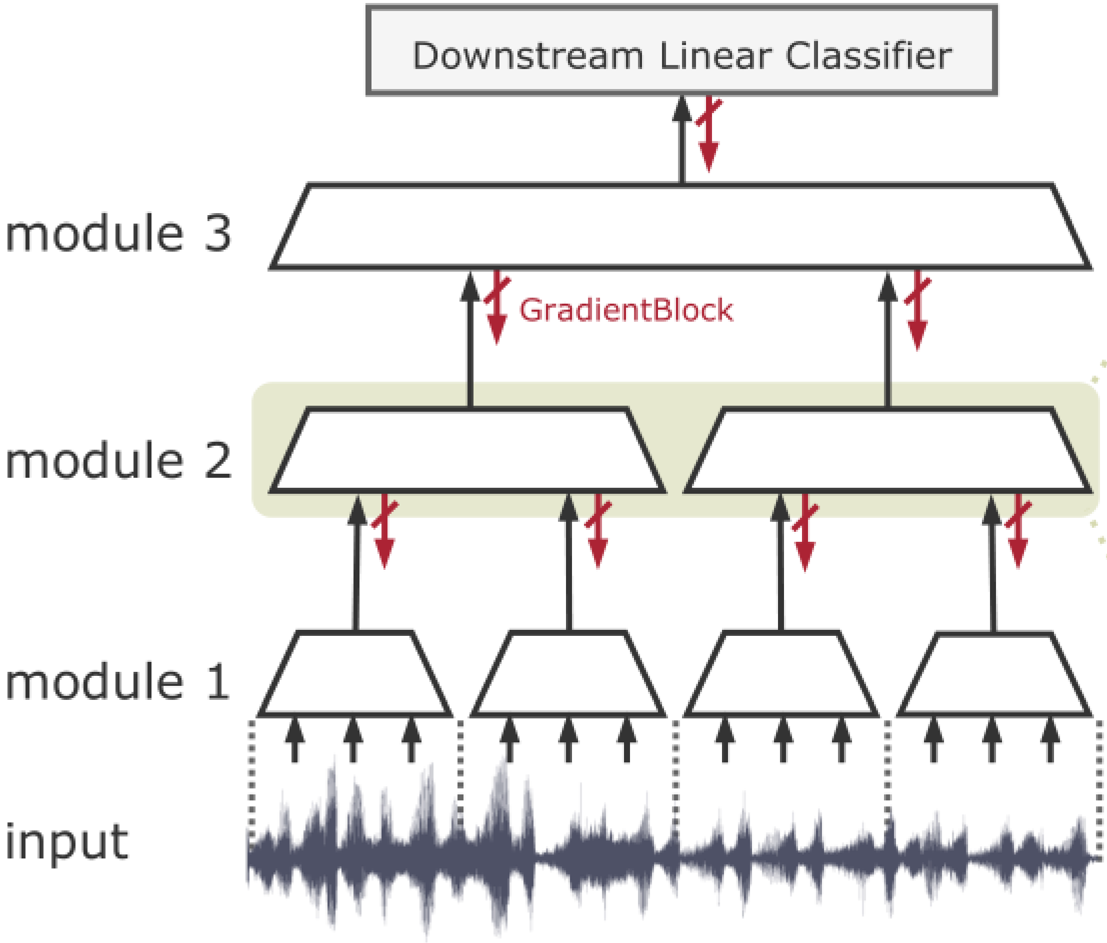
Motivation





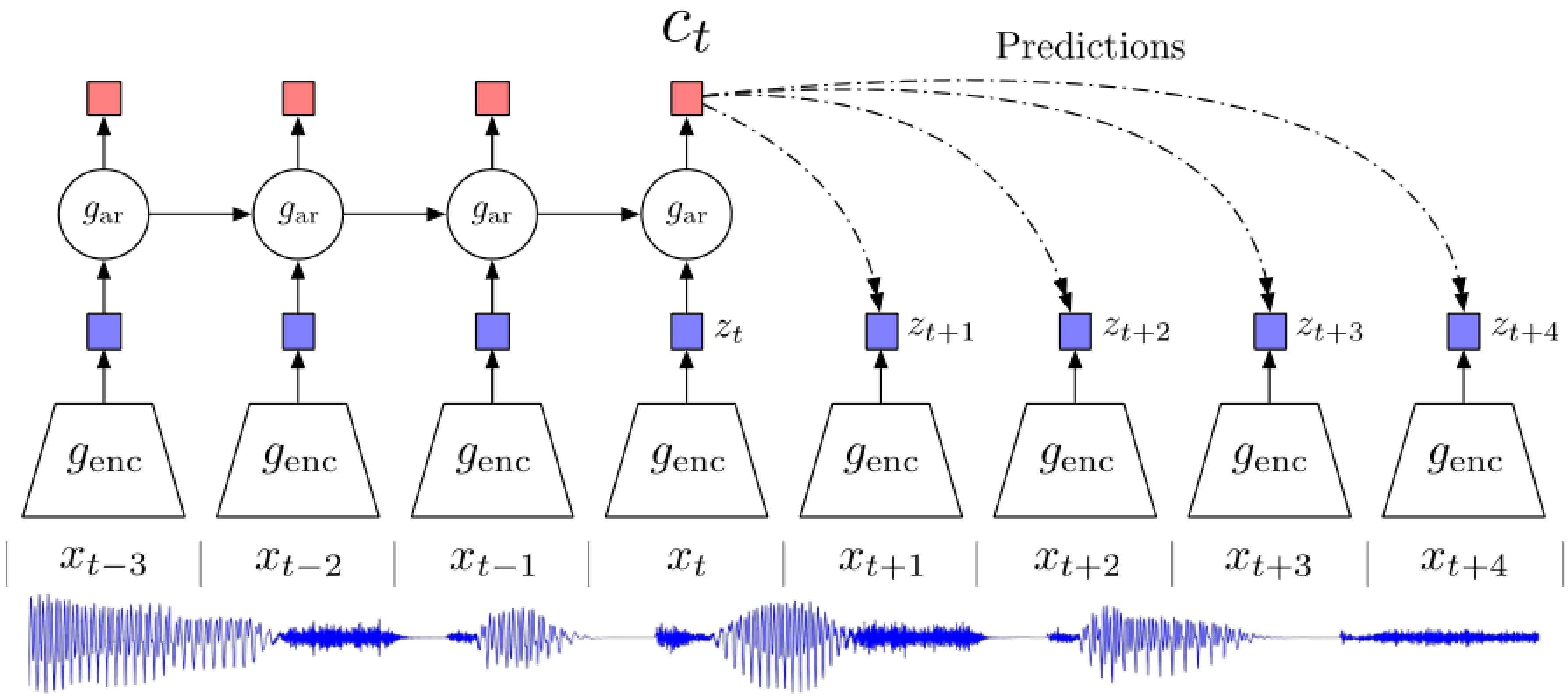
Inspired by the observation that biological neural networks appear to learn without backpropagating a global error signal, we split a deep neural network into a stack of gradient-isolated modules.

Motivation



Inspired by the observation that biological neural networks appear to learn without backpropagating a global error signal, we split a deep neural network into a stack of gradient-isolated modules.

Background---Contrastive Predictive Coding (CPC)



$$f_k(z_j, c_t) = \exp\left(z_j^T W_k c_t\right)$$

$$\mathcal{L}_N = -\sum_k \mathop{\mathbb{E}}_X \left[\log \frac{f_k(z_{t+k}, c_t)}{\sum_{z_j \in X} f_k(z_j, c_t)}\right].$$

The mutual information:

$$I(x; c) = \sum_{x,c} p(x, c) \log \frac{p(x|c)}{p(x)}.$$

$$f_k(x_{t+k}, c_t) \propto \frac{p(x_{t+k}|c_t)}{p(x_{t+k})}$$

Method---Greedy InfoMax (GIM)

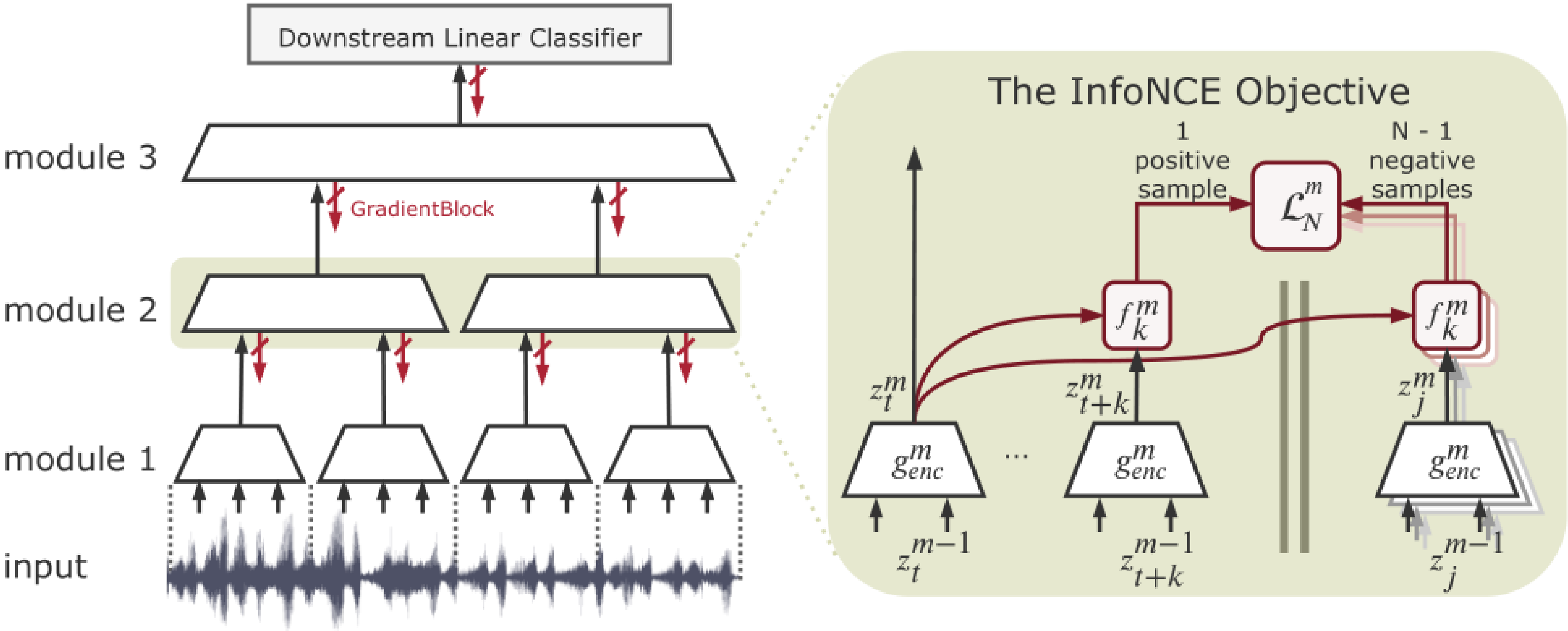


Figure 1: The Greedy InfoMax Learning Approach. **(Left)** For the self-supervised learning of representations, we stack a number of modules through which the input is forward-propagated in the usual way, but gradients do not propagate backward. Instead, every module is trained greedily using a local loss. **(Right)** Every encoding module maps its inputs z_t^{m-1} at time-step t to $g_{enc}^m(\text{GradientBlock}(z_t^{m-1})) = z_t^m$, which is used as the input for the following module. The InfoNCE objective is used for its greedy optimization. This loss is calculated by contrasting the predictions of a module for its future representations z_{t+k}^m against negative samples z_j^m, which enforces each module to maximally preserve the information of its inputs. We optionally employ an additional autoregressive module g_{ar}, which is not depicted here.

Method

$$f_k^m(z_{t+k}^m, z_t^m) = \exp\left({z_{t+k}^m}^T W_k^m z_t^m\right)$$

$$\mathcal{L}_N^m = -\sum_k \mathbb{E}_X\left[\log \frac{f_k^m(z_{t+k}^m, z_t^m)}{\sum_{z_j^m \in X} f_k^m(z_j^m, z_t^m)}\right]$$

After convergence of all modules, the scoring functions $f_k^m(\cdot)$ can be discarded, leaving a conventional feed-forward neural network architecture that extracts features z_t^M for downstream tasks:

$$z_t^M = g_{enc}^M\left(g_{enc}^{M-1}\left(\cdots g_{enc}^1\left(x_t\right)\right)\right)$$

Experiments

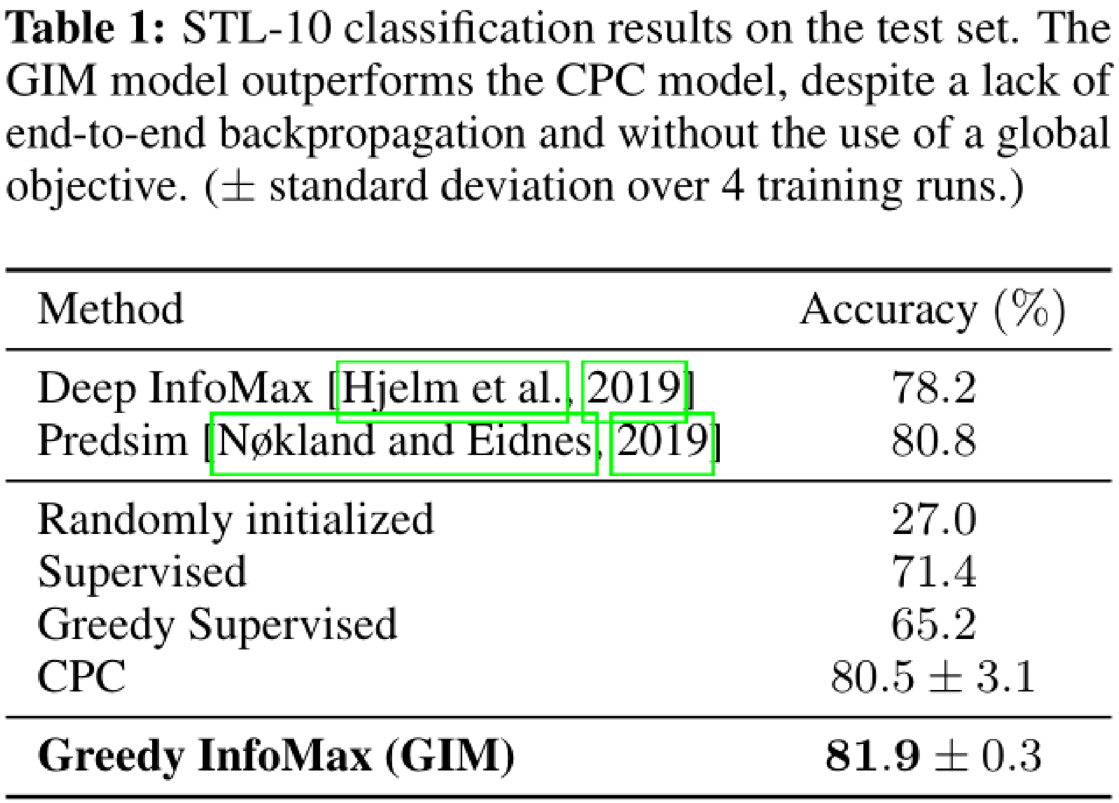


Table 1: STL-10 classification results on the test set. The GIM model outperforms the CPC model, despite a lack of end-to-end backpropagation and without the use of a global objective. (± standard deviation over 4 training runs.)

Method	Accuracy (%)
Deep InfoMax [Hjelm et al., 2019]	78.2
Predsim [Nøkland and Eidnes, 2019]	80.8
Randomly initialized	27.0
Supervised	71.4
Greedy Supervised	65.2
CPC	80.5 ± 3.1
Greedy InfoMax (GIM)	$\mathbf{81.9} \pm 0.3$

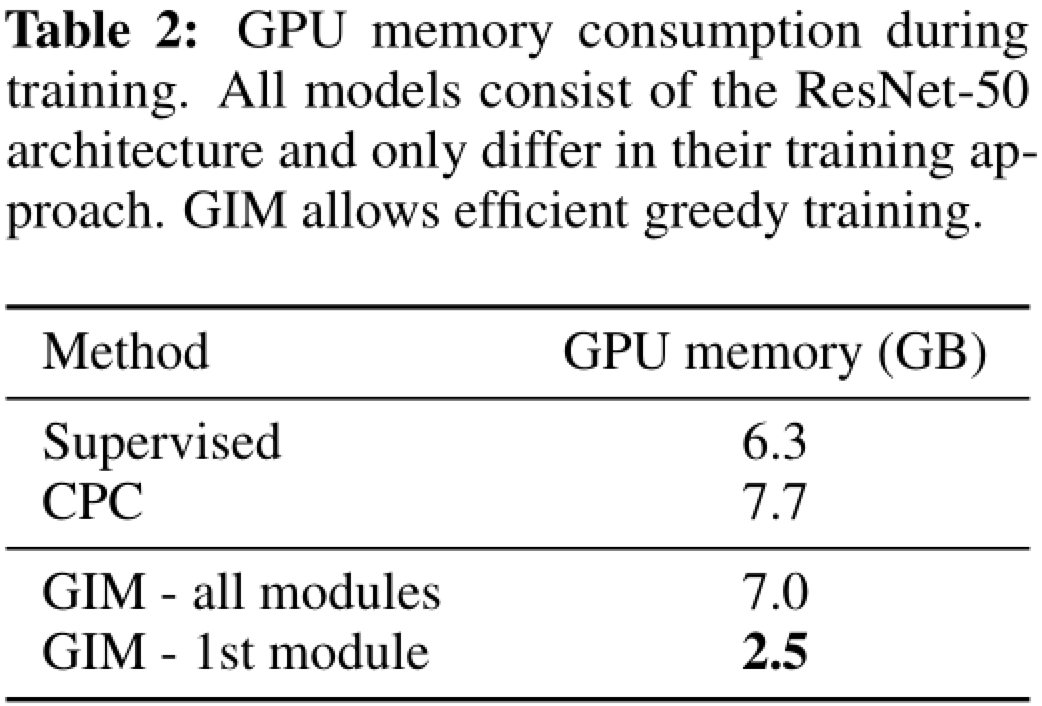
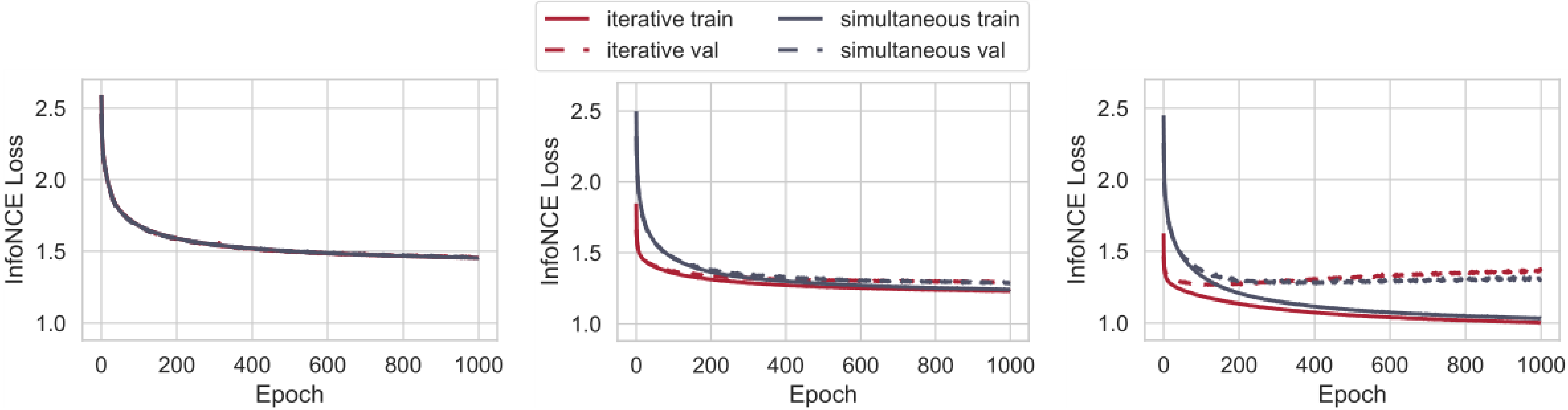


Table 2: GPU memory consumption during training. All models consist of the ResNet-50 architecture and only differ in their training approach. GIM allows efficient greedy training.

Method	GPU memory (GB)
Supervised	6.3
CPC	7.7
GIM - all modules	7.0
GIM - 1st module	**2.5**

Asynchronous memory usage

Experiments



(a) First Module (b) Second Module (c) Third Module

Figure 3: Training curves for optimizing all modules *simultaneously* (blue) or *iteratively*, one at a time (red). While there is no difference in the training methods for the first module (**a**), later modules (**b, c**) start out with a lower loss and tend to overfit more when trained iteratively on top of already converged modules.

Experiments

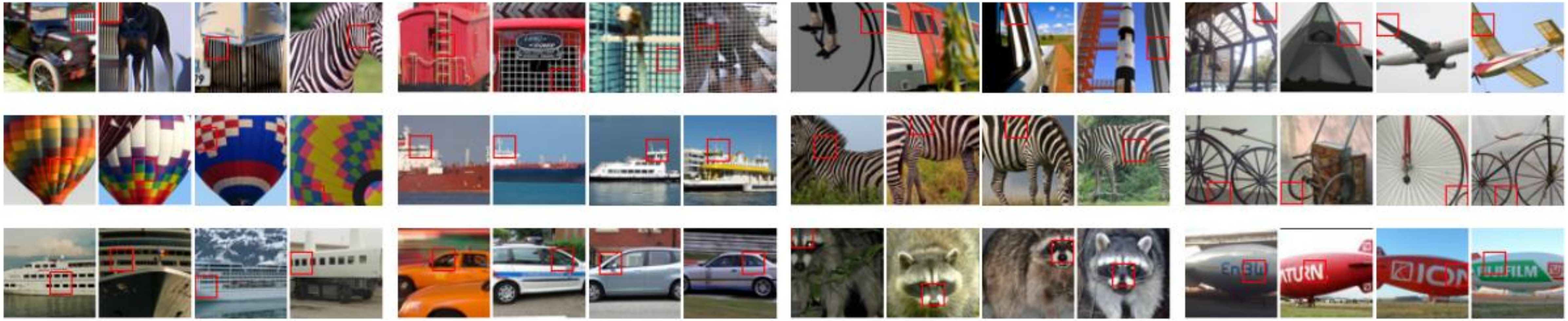


Figure 2: Groups of 4 image patches that excite a specific neuron, at *3* levels in the model (**rows**). Despite unsupervised greedy training, neurons appear to extract increasingly semantic features. Best viewed on screen.

Experiments

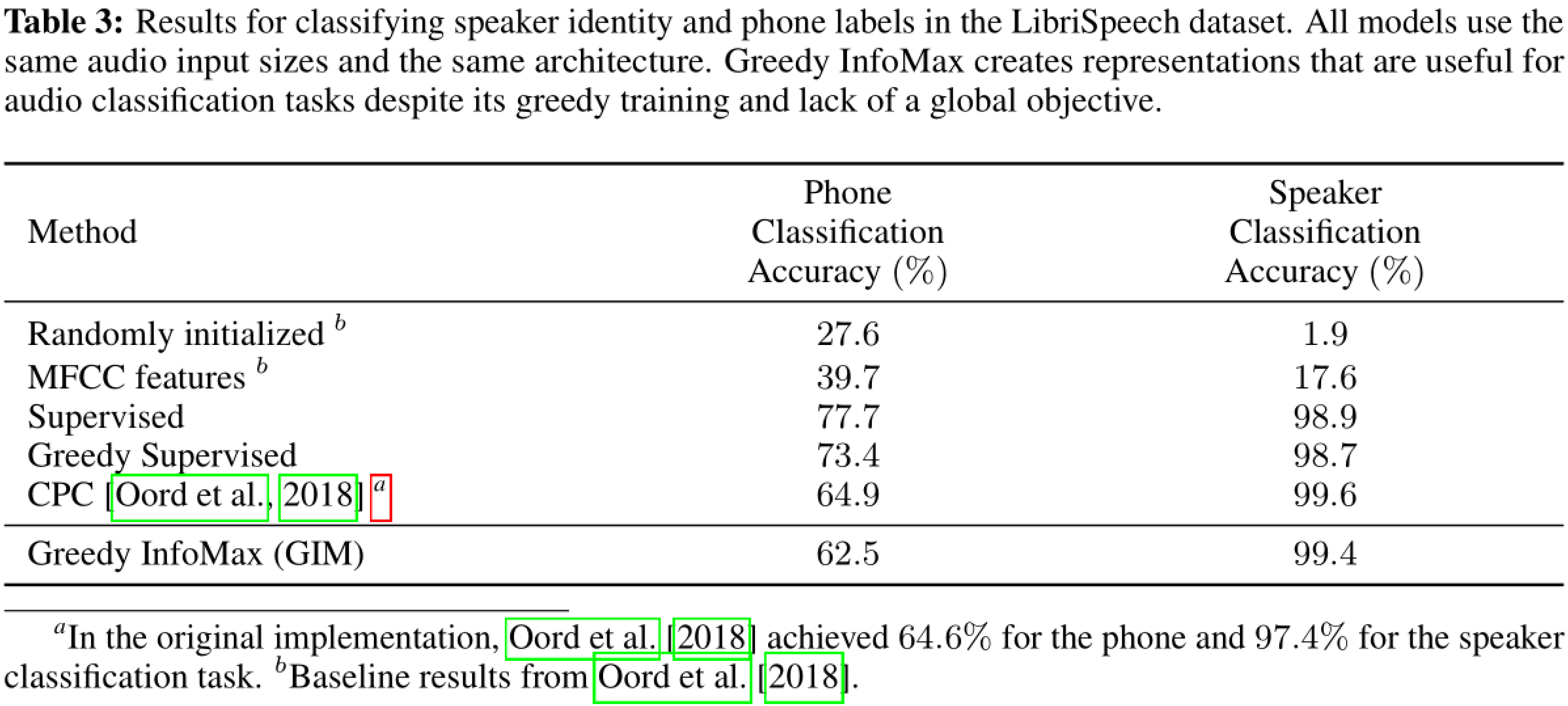
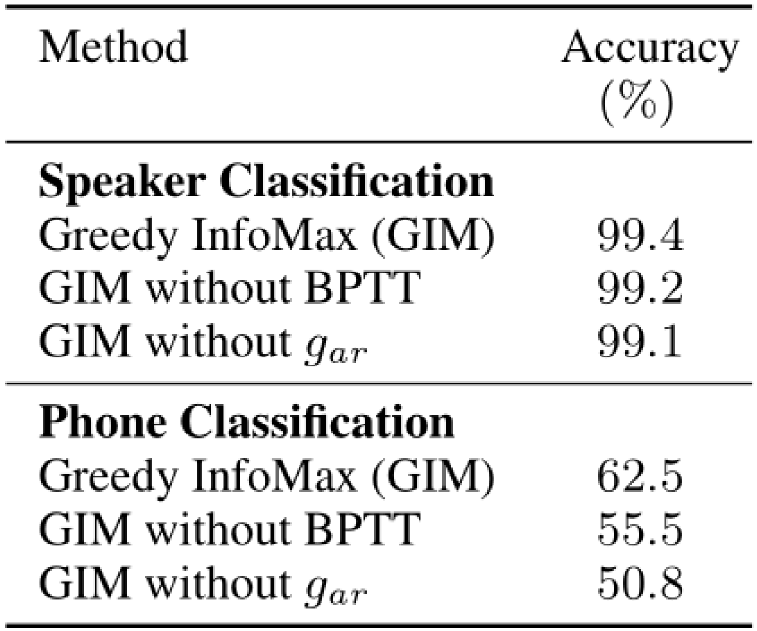


Table 3: Results for classifying speaker identity and phone labels in the LibriSpeech dataset. All models use the same audio input sizes and the same architecture. Greedy InfoMax creates representations that are useful for audio classification tasks despite its greedy training and lack of a global objective.

Method	Phone Classification Accuracy (%)	Speaker Classification Accuracy (%)
Randomly initialized [b]	27.6	1.9
MFCC features [b]	39.7	17.6
Supervised	77.7	98.9
Greedy Supervised	73.4	98.7
CPC [Oord et al., 2018] [a]	64.9	99.6
Greedy InfoMax (GIM)	62.5	99.4

[a] In the original implementation, Oord et al. [2018] achieved 64.6% for the phone and 97.4% for the speaker classification task. [b] Baseline results from Oord et al. [2018].

Experiments



Method	Accuracy (%)
Speaker Classification	
Greedy InfoMax (GIM)	99.4
GIM without BPTT	99.2
GIM without g_{ar}	99.1
Phone Classification	
Greedy InfoMax (GIM)	62.5
GIM without BPTT	55.5
GIM without g_{ar}	50.8

Table 4: Ablation studies on the LibriSpeech dataset for removing the biologically implausible and memory-heavy backpropagation through time.

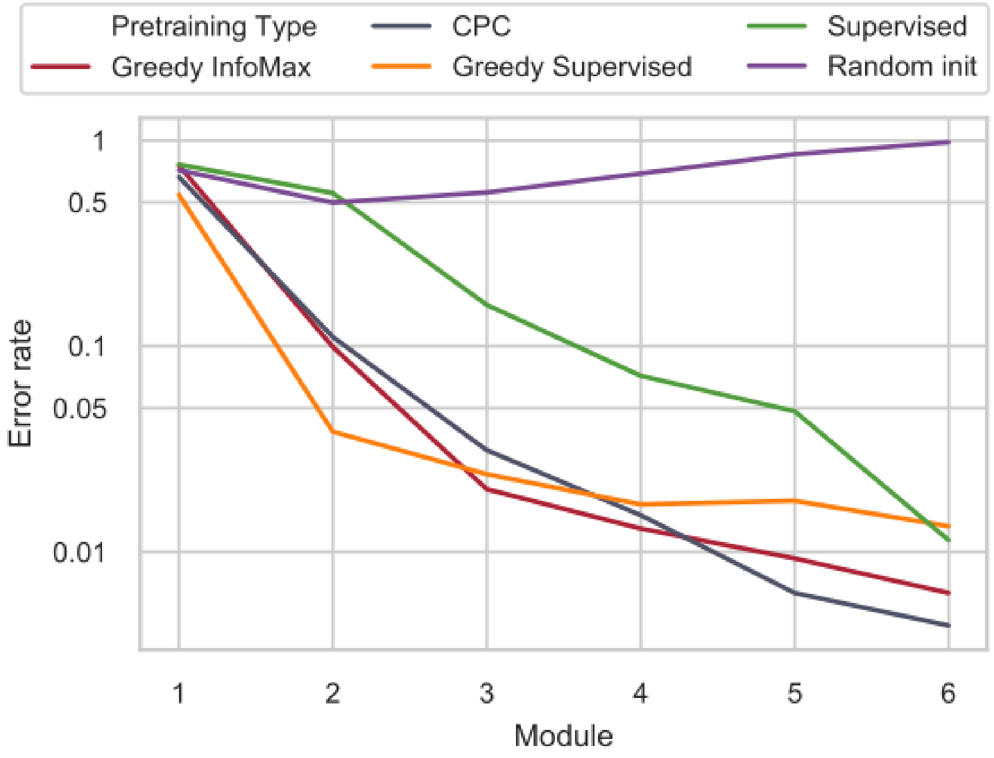
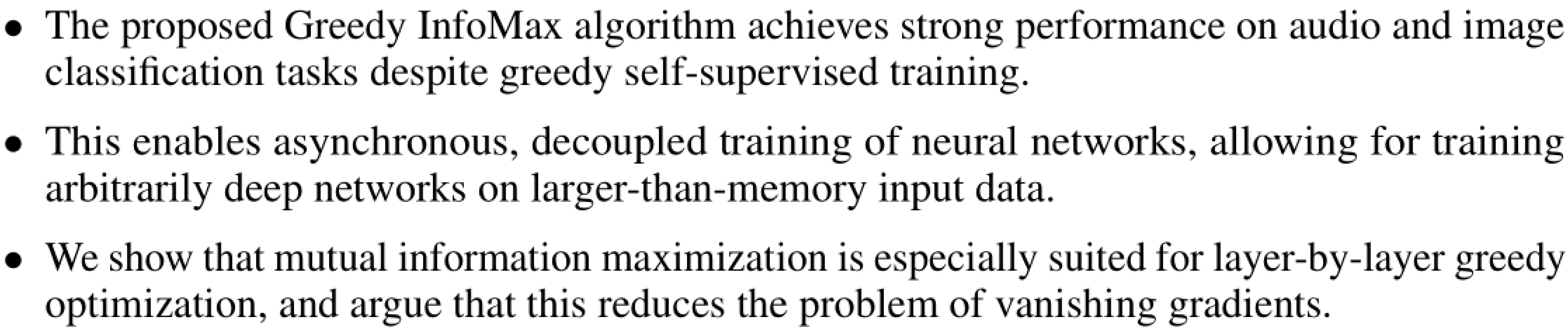


Figure 4: Speaker Classification error rates on a log scale (lower is better) for intermediate representations (layers 1 to 5), as well as for the final representation created by the autoregressive layer (corresponding to the results in Table 3).

Conclusion



- The proposed Greedy InfoMax algorithm achieves strong performance on audio and image classification tasks despite greedy self-supervised training.
- This enables asynchronous, decoupled training of neural networks, allowing for training arbitrarily deep networks on larger-than-memory input data.
- We show that mutual information maximization is especially suited for layer-by-layer greedy optimization, and argue that this reduces the problem of vanishing gradients.

THANKS