DeepNCM: Deep Nearest Class Mean Classifiers

Samantha Guerriero — VANDAL Laboratory, Sapienza Rome University

Barbara Caputo — VANDAL Laboratory, Italian Institute of Technology

Thomas Mensink — Computer Vision Group, University of Amsterdam

Code available: github.com/tmensink/deepncm

Summary

- **Multi-class distance-based classification**
- **End-to-end learning deep representation**
- **Mean updates are too expensive**
- **Proposed solution: Online mean updates**
- **Alternatives: mean condensation and decay**
- **Experiments: On par with ResNet SoftMax**

Related Work

- Incremental Classifier & Representation Learning [2]

  Uses NCM on top of SoftMax DeepNet representation
- Prototypical Networks [3]

  Learns DeepNet representations using NCM formulation, yet in a small number of classes & examples setting such that class means can be computed exactly every batch

Current Research

- Large-Scale experiments, *e.g.* ImageNet
- Generalisation experiments

  Generalise DeepNCM representations (better) to new classes?
- Open-Set / Adversarial Examples

  Do distance based class boundaries form a protection ?
- Learning rate schedules and minimisers

  Optimise learning rate and minimisers for DeepNCM

References

[1] T. Mensink et al. – *Distance-based classification: Generalizing to new classes* – PAMI'13

[2] S. Rebuffi et al. – *ICaRL: Incremental classier and representation learning* – CVPR'17

[3] J. Snell et al. – *Prototypical networks for few-shot learning* – NIPS'17

NCM

- Class predictions (with class means $\boldsymbol{\mu}_y$)

$$y^\star = \operatorname*{argmin}_{y \in \{1,\ldots,Y\}} d(\boldsymbol{x}, \boldsymbol{\mu}_y)$$

- Maximum-Likelihood Metric Learning [1]:
  - Use parametric distance d^W_{xy}
  - Probabilistic interpretation: $p(y|\boldsymbol{x}) \propto \exp -\frac{1}{2} d^W_{xy}$
  - Learning Objective: $\frac{1}{N}\sum_i^N \ln p(y_i|\boldsymbol{x}_i)$

DeepNCM

- Distance between learned representations

$$d^\phi_{xy} = (\boldsymbol{\phi}(x) - \boldsymbol{\mu}^\phi_y)^\top (\boldsymbol{\phi}(x) - \boldsymbol{\mu}^\phi_y)$$

- Using learned representation $\boldsymbol{\phi}(x)$, *e.g.* ResNet
- Mean of learned representations

$$\boldsymbol{\mu}^\phi_y = \frac{1}{N_y} \sum_{i:y_i=y} \boldsymbol{\phi}(x_i)$$

Mean Updates:

- Class-means drift due to learning representation
- *Idea* After each mini-batch update means
- *Problem* Mean updates are too expensive (1 epoch)

Mean Updates per Batch

- Online Mean

$$\boldsymbol{\mu}^\phi_{y_i} \leftarrow \frac{n_{y_i}}{n_{y_i}+1} \boldsymbol{\mu}^\phi_{y_i} + \frac{1}{n_{y_i}+1} \boldsymbol{\phi}(x_i)$$

- Online Mean Condensation

  After each epoch, condense mean to single observation
- Decay Mean

$$\boldsymbol{\mu}^\phi_{y_i} \leftarrow \alpha\, \boldsymbol{\mu}^\phi_{y_i} + (1-\alpha)\, \boldsymbol{\phi}(x_i)$$

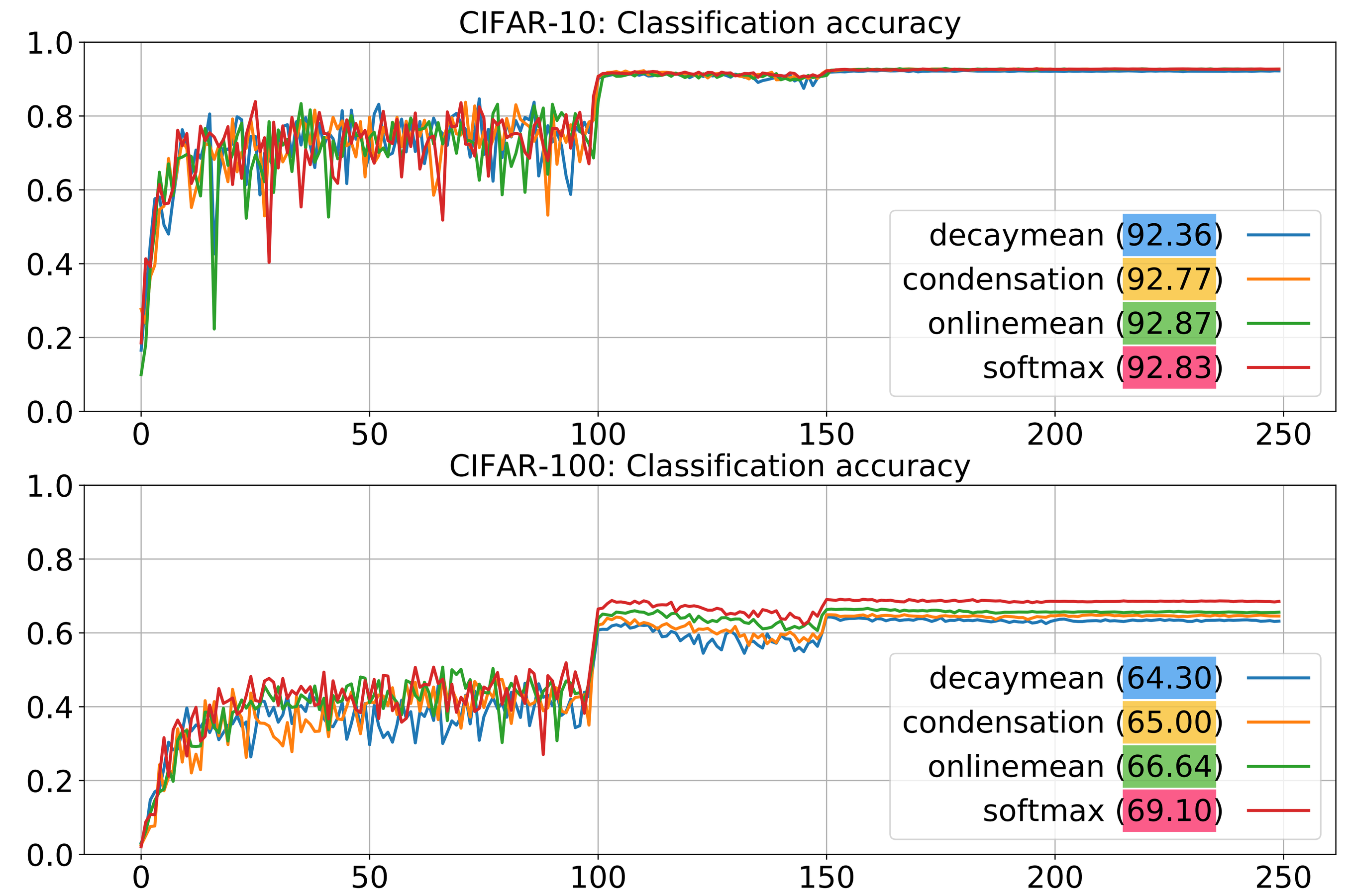
Results on CIFAR

- CIFAR-10 and CIFAR-100 dataset
- ResNet implementation in TensorFlow
  - Based on TensorFlow official ResNet model
  - Use pre-defined (softmax optimised) learning rate schedule

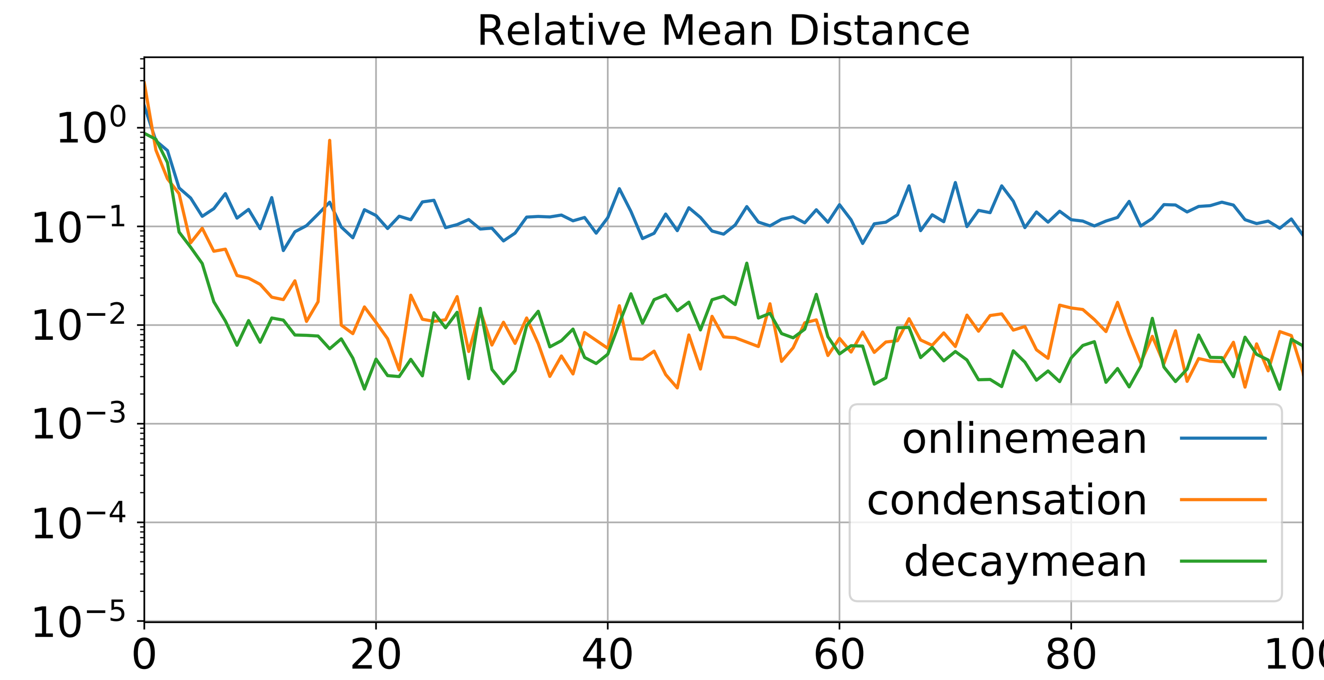
Preliminary Experiments

- Initial learning rate 0.1 works better than 0.01
- Condense after 100 iterations, .5/1/2 epochs all similar
- Decay with $\alpha = \{.5, .75, .9, .95\}$ higher is better

Accuracy over Epochs on Cifar10 and Cifar100



Relative Mean Distance on Cifar10



While mean condensation has lower RMD, accuracy is not (much) improved over online mean Possible reason: Optimiser parameters (*e.g.* momentum) are influenced by condensation