Supplementary material for: Class-Incremental Mixture of Gaussians for Deep Continual Learning

Lukasz Korycki
Virginia Commonwealth University
koryckil@vcu.edu

Bartosz Krawczyk
Rochester Institute of Technology
bartosz.krawczyk@rit.edu

A. Appendix

A.1. Data

We used: MNIST, FASHION, SVHN, CIFAR10 and IMAGENET10 – a subset of the tiny IMAGENET200, to gain deeper insights into our method while conducting experiments with hundreds of different configurations. Then, we extended this set with CIFAR20 – the coarse-grained version of CIFAR100, IMAGENET20A and IMAGENET20B – larger subsets of IMAGENET200 – to benchmark our method against other algorithms.

For the experiments involving fixed extractors, we used pre-trained features to construct four additional sequences – CIFAR100-PRE10, CIFAR100-PRE100, IMAGENET200-PRE20 and IMAGENET200-PRE200, which consisted of features extracted for CIAFR100 and IMAGENET200, using extractors trained on 10, 20, 100 and 200 classes of the original datasets. The summary of the used benchmarks is given in Tab. 1. Details of the feature extractors can be found in the next section.

Table 1. Summary of used datasets.

Dataset	Train	Test	Cls	Feats
MNIST	50 000	10 000	10	No
FASHION	60 000	10 000	10	No
SVHN	73 257	26 032	10	No
IMAGENET10	5000	500	10	No
CIFAR10	50 000	10 000	10	No
IMAGENET20A	10 000	1000	20	No
IMAGENET20B	10 000	1000	20	No
CIFAR20	50 000	10 000	20	No
CIFAR100-PRE10	50 000	10 000	100	128
CIFAR100-PRE100	50 000	10 000	100	512
IMAGENET200-PRE20	100 000	10 000	200	256
IMAGENET200-PRE200	100 000	10 000	200	256

A.2. Model configurations

In the first section of our experiments, we explored different configurations of our algorithm, which can be mostly seen as an ablation study. Firstly, we evaluated different **losses** (CE, MC and MCR) combined with different **classification methods** (softmax, max-component). Secondly, we checked different settings for the **tightness bound** parameter τ_p by evaluating a grid of values for inter-tightness and intra-tightness – we considered $\tau_p \in \langle$1e-06, 1e-05, 0.0001, 0.001, 0.01$\rangle$ for both. Thirdly, we analyzed how assuming different **numbers of components** affects the classification performance on different datasets. We used $K \in \langle 1, 3, 5, 10, 20\rangle$. Then we checked if it is better to maintain a whole **covariance matrix** or only its variance (FULL, VAR). Finally, we evaluated different **learning rates** for the extractor and GMM part, using $\alpha_{\mathcal{F}} \in \langle$1e-07, 1e-06, 1e-05, 0.0001, 0.001$\rangle$ and $\alpha_{\mathcal{G}} \in \langle$1e-05, 0.0001, 0.001, 0.01, 0.1$\rangle$, to check whether it may be beneficial to configure them separately, and different **memory sizes** $\mathcal{M}_c \in \langle 8, 64, 128, 256, 512\rangle$ to analyze how our method exploits limited access to class examples.

While evaluating specific parameters we kept others fixed. For our base configuration we chose a setup that was capable of providing performance comparable with a standard experience replay. We used the MCR with max-component as our loss and classification method, $K = 3$, $\tau_{p,ie} = 0.002$, $\tau_{p,ia} = 0.01$, $\beta = 0.5$, $\alpha_{\mathcal{F}}$ =0.0001, $\alpha_{\mathcal{G}}$ =0.001 and $d_{min} = 0.001$ with only variance stored per each component. We assumed a modest memory buffer per class $\mathcal{M}_c = 256$ and matched the size of a memory sample per class with the training batch size. The model was trained for 10 (MNIST, FASHION) or 25 epochs per class, with 32 (IMAGENET) or 64 instances in a mini-batch.

A.3. Algorithms

Based on the observations made in the first section of the experiments, in the final evaluation we used two variants of our algorithm: **MIX-CE** and **MIX-MCR** with $\tau_{p,ie}$ =0.0001, $\tau_{p,ia}$ =0.001, $\alpha_{\mathcal{F}}$ =0.0001, $\alpha_{\mathcal{G}}$ =1e-05 and, once again, $d_{min} = 0.001$ with only variance maintained per each component. The only parameter that we tuned per each dataset was the number of components K. We used Adam as the optimizer. For the memory-free sce-

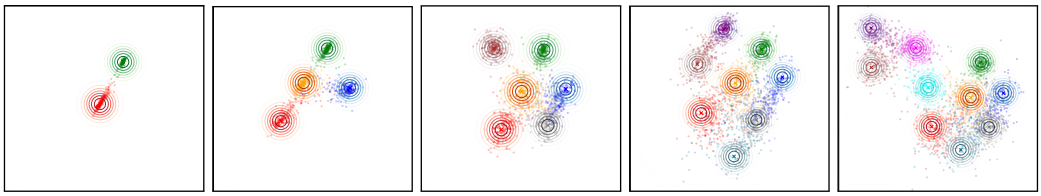


Figure 1. Learning subsequent classes of FASHION incrementally (K=1) with the inter-contrastive loss utilizing the tightness bound ($\tau_{p,ie}$=0.2).

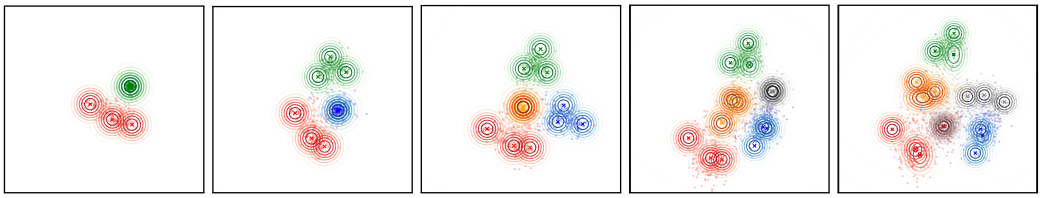


Figure 2. Learning subsequent classes of FASHION incrementally (K=3) with regionalization and the intra-contrastive loss utilizing the tightness bound ($\tau_{p,ia}$=0.25).

narios with pre-trained extractors, we turned off the inter-contrastive loss to minimize interference with previously learned classes.

The main parameters of the baselines methods were set based on the original papers and other literature, including empirical surveys or works containing vast empirical studies [1, 2, 4–6, 9]. For all memory sampling methods we matched the memory sampling size with the training batch size. For ERSB we used 10 centroids per class each containing up to either 25 or 15 instances to match the total memory size. DER used α_d=0.5, for LWF we set the softmax temperature $T = 2$ and progressively increased its distillation coefficient as suggested in [5], and SI used λ =0.0001. All of the methods utilized the Adam optimizer with a learning rate α=0.0001 as we did not observe any significant differences when changing this parameter.

Analogously to the configuration section, all of the algorithms, including ours, were trained for 10 (MNIST, FASHION) or 25 epochs per class, using 32 (IMAGENET) or 64 instances per mini-batch. The offline models were trained for either 50 or 100 epochs, until they achieved a saturation level. The memory buffer was set to $\mathcal{M}_c = 128$ (IMAGENET) or $\mathcal{M}_c = 256$ for methods supporting memory per class (ER, ERSB, iCaRL), and $\mathcal{M} = C \cdot 128$ or $\mathcal{M} = C \cdot 256$ for the remaining ones (GSS, A-GEM, DER), where C was the total number of classes. The latter group was equipped with reservoir buffers [1]. For the experiments with pre-trained extractors we wanted to check the **memory-free scenario**, therefore we set $\mathcal{M}_c = 0$ for our methods and $\mathcal{M}_c = 1$ or $\mathcal{M} = C$ for others, since most of them could not be run without storing any examples.

All of the algorithms, including different configurations of our method, were combined with feature extractors. For MNIST and FASHION we used a simple CNN with two convolutional layers consisting of 32 (5x5) and 64 (3x3) filters, interleaved with ReLU, batch normalization and max pooling (2x2). For SVHN and IMAGENET we utilized ResNet18, its modified version for CIFAR10 and CIFAR20, and ResNeXt29 for CIFAR100 [10]. The classification layers consisted of the default configurations.

Finally, for our method, ER, ERSB, A-GEM and DER we disabled batch normalization, since, consistently with [8, 11], we observed a significant difference in performance when those layers were turned off for the given methods. As mentioned in Sec. A.1, for the memory-free scenarios, the extractors were pre-trained on either 10, 20, 100 or 200 classes of CIFAR100 and IMAGENET200. For this setting we trained all the models for 20 epochs per class.

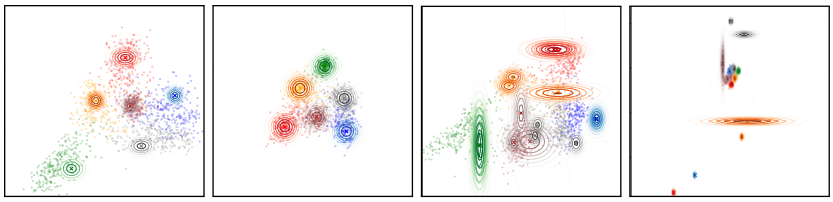
Results for the offline model were either obtained by us (learned from scratch for IMAGENET20A, IMAGENET20B and fine-tuned models for IMAGENET200), or by referring to other publications [4, 7].

B. Appendix

B.1. Additional visualizations

Fig. 1 presents an example of a single-component class-incremental mixture model learned with the inter-contrastive loss. Fig. 2 demonstrates the effectiveness of training a multi-component model with the intra-contrastive loss and regionalization.

As mentioned in the main document, the CE loss can of-



CE → 0.84 MC → 0.85 CE → 0.84 MC → **0.86**

Figure 3. Mixtures learned with cross-entropy and simple max-component strategy (K=1 and K=3) after 6 classes of FASHION.

ten achieve similar predictive performance even if its mixture models are not really fitting the data (Fig. 3). We can see it when compared with MC for K=1 or MCR for both K (Fig. 1 and 2). Furthermore, the model produced for MC with K=3 clearly shows that it is incapable of effectively utilizing multiple components for the same class. Please notice that only the Gaussians in the middle actually cover some data points, while the remaining components are completely unrelated to the observed data. These are examples of the degenerate solutions. While for FASHION this loss could still, analogously to CE, provide similar performance as MCR (the components in the middle are fitted to the data and they are sufficient to model it), the observed desynchronization of components results in its weaknesses for more complex problems. The MCR loss can provide high quality of predictive performance and of the produced mixture models.

B.2. Additional configurations

Number of components: Tab. 2 presents how many components were required to obtain the best solutions per each dataset for the given settings. We can observe that for simpler datasets (MNIST, FASHION) using a single component per class for sufficient and that introducing additional ones led to slightly worse performance, most likely due to the fact of fitting to simple concepts and overcomplicating the optimization problem. On the other hand, more complex benchmarks (SVHN, CIFAR10, IMAGENET10) preferred access to more components per class, which could provide significant improvements, e.g., for SVHN the difference between K=1 and K=10 was almost 0.3. While for these experiments we set the learning rate slightly higher for the GMM model (0.001) than for the extractor (0.0001), we observed that when the former used rate lower than the latter (as suggested by the results for learning rates that will be presented below), the optimal K tended to be lower on average. It is possible that if GMM is dominant it prefers having more flexibility (components), while when the extractor has a higher learning rate it may be more effective in adjusting representations to lower numbers of components.

Covariance: Results presented in Tab. 3, unequivocally show that our gradient-based MIX can much better adapt to data if it maintains only the variance of the covariance matrix (better by almost 0.3 when compared with full covariance). It is not surprising since previous publications related to the gradient-based GMMs for offline settings suggested a similar thing [3]. Most likely, working with a full covariance matrix leads to less stable loss values, and many more free parameters (especially if the feature space is high-dimensional) likely cause problems with convergence.

Table 2. Average incremental accuracy for MIX using different numbers of components K.

Config	MNIST	FASHION	SVHN	CIFAR10	IMGNET10
K=1	**0.9885**	**0.8859**	0.4862	0.4282	0.6466
K=3	0.9875	0.8782	0.5978	0.5407	0.6584
K=5	0.9463	0.8562	0.6994	0.5522	**0.6604**
K=10	0.9393	0.8577	**0.7438**	**0.5620**	0.6252
K=20	0.9521	0.8517	0.6868	0.5532	0.4270

Table 3. Average incremental accuracy for MIX with diagonal and full covariance.

Config	MNIST	FASHION	SVHN	CIFAR10	IMGNET10
FULL	0.7304	0.6577	0.2931	0.3298	0.3255
VAR	**0.9888**	**0.8849**	**0.6393**	**0.5777**	**0.6865**

Learning rates: Analogously to the experiments for tightness, in Fig. 4 we presented the grid of results for different extractor (horizontal) and mixture (vertical) learning rates. The obtained results suggest that the former part is more important – once the optimal rate is set (0.0001 for the given settings) tuning the latter seems less significant, although overall it should be set to a similar or slightly lower value.

Memory size: Finally, if we look at the results of class-incremental learning using different memory sizes, given in Fig. 5, we will see that MIX can effectively utilize larger buffers and that it seems to be quite memory-dependent, especially for SVHN where the difference between subsequent sizes ranged from 0.1 to 0.2. Still, the gap was much

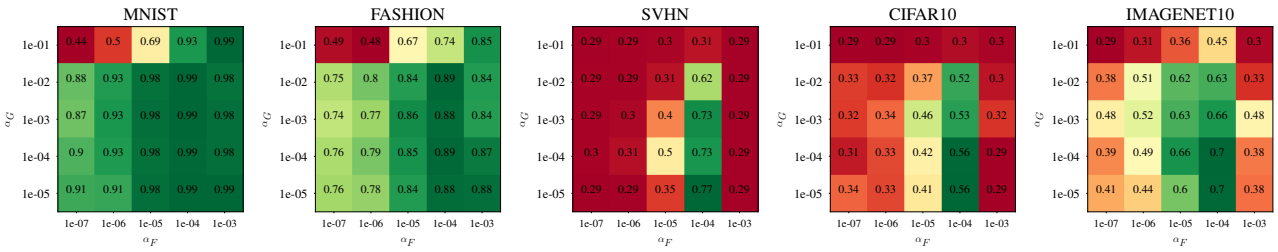


Figure 4. Average incremental accuracy for different learning rates.

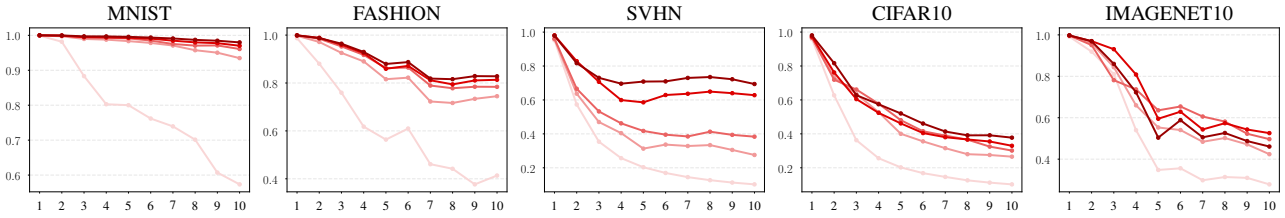


Figure 5. Incremental accuracy after each class batch for different sizes of the replay buffer.

smaller for all of the remaining datasets. While this characteristic of the algorithm may be problematic (the fewer examples we need, the better), it is still valid that if we can use a pre-trained extractor, the whole model does not need to use the memory buffer at all.

B.3. Lessons learned

Based on the theoretical and empirical analysis presented for this work we can conclude the following.

1. **Class-incremental learner.** Regardless of many combined challenges, it is possible to successfully hybridize the gradient-based mixture models on top of convolutional feature extractors, and use them in class-incremental end-to-end continual learning scenarios. The presented results show that MIX is capable of providing competitive results when compared with well-known incremental baselines.
2. **Dedicated losses.** It has been shown that the training of the mixture models combined with dynamic feature extractors requires the inter-contrastive loss to effectively distinguish components of different classes from each other. In addition to that, to ensure diversity among same-class components and avoid degenerate solutions, such techniques as regionalization combined with the intra-contrastive loss are required. We showed that not only do the proposed approaches deliver what was intended, but also that they can translate into significant performance gains for more complex datasets. Finally, although the more generic high-level cross-entropy loss may provide good solutions in many cases, only the most advanced variant (MIX-MCR) delivers both high predictive performance and high quality of generated mixture models, which may be important from the perspective of interpretability or potential Gaussian-based extensions.
3. **Effective tightness.** The tightness bound plays a crucial role in stabilizing the mixture learning procedure. Setting the optimal values of inter- and intra-tightness leads to striking a balance between pushing different components from each other and actually fitting them to the data. Intuitively, the inter-tightness prefers slightly lower values than intra-tightness.
4. **Recommended configurations.** By analyzing other different hyperparameter settings and combinations of our methods we could observe that: (i) the CE loss works much better with the softmax classification method, while MC and MCR should be combined with the max-component approach, (ii) different numbers of components may be required for different data and different learning rates may also affect the optimal number, (iii) maintaining only the diagonal of the covariance matrices leads to more stable optimization and better results, (iv) the learning rate for the feature extractor dominates over the one for the mixture model, and that (v) MIX is quite memory-dependent in general end-to-end scenarios.
5. **Memory-free scenarios.** At the same time, MIX is capable of learning without a memory buffer if we use a fixed pre-trained extractor and disable the contrastive loss that is not needed in this case. Our method stands out as the best model for such class-incremental scenarios which can be very important if there are any data privacy concerns or strict memory limits.

References

[1] Pietro Buzzega, Matteo Boschini, Angelo Porrello, Davide Abati, and Simone Calderara. Dark Experience for General Continual Learning: a Strong, Simple Baseline. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 2

[2] Arslan Chaudhry, Marc'Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient Lifelong Learning with A-GEM. In *International Conference on Learning Representations (ICLR)*, 2019. 2

[3] Alexander Rainer Tassilo Gepperth and Benedikt Pfülb. Gradient-Based Training of Gaussian Mixture Models for High-Dimensional Streaming Data. *Neural Process. Lett.*, 53:4331–4348, 2021. 3

[4] Łukasz Korycki and Bartosz Krawczyk. Class-Incremental Experience Replay for Continual Learning under Concept Drift. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 3644–3653, 2021. 2

[5] Davide Maltoni and Vincenzo Lomonaco. Continuous learning in single-incremental-task scenarios. *Neural Networks*, 116:56–73, 2019. 2

[6] Marc Masana, Xialei Liu, Bartlomiej Twardowski, Mikel Menta, Andrew D. Bagdanov, and Joost van de Weijer. Class-incremental learning: survey and performance evaluation. *CoRR*, abs/2010.15277, 2020. 2

[7] Massimiliano Patacchiola and Amos J. Storkey. Self-supervised relational reasoning for representation learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 2

[8] Quang Hong Pham, Chenghao Liu, and Steven C. H. Hoi. Continual Normalization: Rethinking Batch Normalization for Online Continual Learning. *arXiv*, abs/2203.16102, 2022. 2

[9] Gido M. van de Ven and Andreas Savas Tolias. Three scenarios for continual learning. *arXiv*, abs/1904.07734, 2019. 2

[10] Saining Xie, Ross B. Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated Residual Transformations for Deep Neural Networks. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5987–5995, 2017. 2

[11] Minghao Zhou, Quanziang Wang, Jun Shu, Qian Zhao, and Deyu Meng. Diagnosing Batch Normalization in Class Incremental Learning. *arXiv*, abs/2202.08025, 2022. 2