Generative Feature Replay with Orthogonal Weight Modification for Continual Learning Supplemental Materials

IJCNN 2021 Submission
Gehui Shen, Song Zhang, Xiang Chen, Zhi-Hong Deng
School of Electronics Engineering and Computer Science, Peking University
{jueliangguke, songz, caspar, zhdeng}@pku.edu.cn

I. Upper Bound of Generative Feature Replay

To further explain why GFR can improve the performance of OWM, we design a training strategy to estimate the upper bound of GFR. First, we save all data of previous tasks during training. Then when training T_n, at each iteration, we sample a mini-batch $\mathcal{M}_r = \{(x_i, y_i)\}_{i=1}^{M}$ from previous data $D_{<n}$ as experience replay (ER) [1, 2] methods do. However, we use feature extractor E to compute the penultimate layer feature $\mathbf{h}_i$ for x_i and stop the gradient back-propagation through the feature $\mathbf{h}_i$. In this way, this type of replay can only alleviate the catastrophic forgetting of the last FC layer F which is identical to GFR. We call this training strategy Experience Replay above Feature (ERaF). Because in ERaF we can obtain the exact feature from previous data with the current feature extractor, its performance can be considered as the upper bound of GFR. Ideally, if there is no change of feature distribution in classifier and no catastrophic forgetting in generator, GFR can achieve the performance of ERaF.

In table I, we give a comparison of GFR and ERaF based on OWM with SSL auxiliary task on different image datasets. As we expected, we find ERaF always works better than GFR, which implies that reducing the confusion of the last FC layer, the main purpose of GFR, is potentially effective for CIL. In addition, on relatively simple datasets SVHN and CIFAR10, the gap between GFR and ERaF is small as the feature is relatively stable. On more complex CIFAR100, the gap is larger especially in the settings with more (5 and 10) tasks.

II. Implementation Details

In all experiments, we train classifier C with SGD optimizer. In preliminary experiments, we found OWM algorithm required larger learning rate to get the best performance as gradient projection reduces the stepsize of the actual update. For all methods trained with OWM, including OWM, OWM+GIR and two proposed methods (OWM+GFR and OWM+SSL+GFR), the learning rate is selected from $\{0.02, 0.05, 0.075, 0.1\}$. For other methods, the learning rate is selected from $\{0.005, 0.01, 0.03, 0.05\}$. The momentum is always set to 0.9. When training generator (G, D) for GFR or GIR methods, we use Adam [3] optimizer following Gulrajani et al. [4]. The learning rate is $1e-4$ and β_1 and β_2 are set to 0.5 and 0.9 respectively. For image data, we train classifier 50 epochs and training generator 250 epochs. For text data, we train classifier 15 epochs and training generator 50 epochs. The size of mini-batch of current data is always 64 and we select the mini-batch size of replay data from $\{16, 32, 64\}$. The temperature T of distillation loss for GFR and GIR methods is set to 2 following Castro et al. [5]. The α of SSL loss coefficient is selected from $\{0.5, 1, 2, 5\}$. We split a validation set from the original test set only for selecting hyperparameters following Hu et al. [6]. For classifier, we use dropout [7] operation with a 0.2 drop rate after each convolution layer. The activation function is ReLU [8].

Please note that we reimplement all baselines except PGMA [6] and DGM [9] as described in the main text. The results of the former are from the original paper and for the latter, we run the source code[1] released by authors.

A. Data Preprocessing

The 3 image datasets, SVHN [10], CIFAR10 and CIFAR100 [11], we use all consist of 32×32 color images. We transform the pixel values to the range $[0, 1]$ and normalize the transformed values by the mean and standard deviation of the training dataset. For text data, we use the same pre-trained word embedding following Hu et al. [6]. Specially, we use GloVe [12] embeddings for DBPedia [13] and Chinese word embeddings proposed by Li et al. [14] for THUCNews [15].

B. Code Dependencies and Hardware

The Python version is 3.7.6. We use the PyTorch deep learning library to implement the models. The version of PyTorch is 1.3.1. Other dependent libraries includes Numpy (1.17.3), torchvision (0.4.1), Keras(2.3.1). The CUDA version is 10.2. We ran all experiments on 1 NVIDIA RTX 2080ti GPU. We will publish our codes once the paper is accepted.

III. More Analysis Results

A. Feature Stability Analysis

We plot the averaged change of feature on the valid dataset of the first task of other datasets which are not reported in the

[1] https://github.com/SAP-samples/machine-learning-dgm

Methods	SVHN (5 tasks)	CIFAR10 (5 tasks)	CIFAR100 (2 tasks)	CIFAR100 (5 tasks)	CIFAR100 (10 tasks)
OWM	73.92	54.52	42.28	34.16	30.54
OWM+SSL+GFR	78.36(+4.44)	57.81(+3.29)	43.51(+1.23)	36.32(+2.16)	33.22(+2.68)
OWM+SSL+ERaF	78.57(+4.65)	58.94(+4.42)	45.41(+3.13)	39.04(+4.88)	35.61(+5.07)

TABLE I: The comparison between OWM, OWM+SSL+GFR and OWM+SSL+ERaF. OWM+SSL+ERaF can be considered as the upper bound of OWM+SSL+GFR. The number in parentheses is the improvement relative to OWM baselines.

main text due to space limit. The results on CIFAR100 are in Figure 1 and the results on two text datasets are in Figure 2.

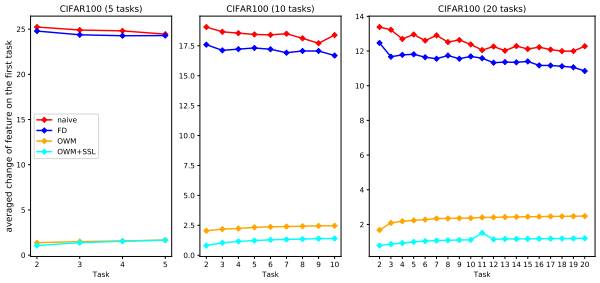

Fig. 1: The averaged Δ_i on different tasks of CIFAR100.

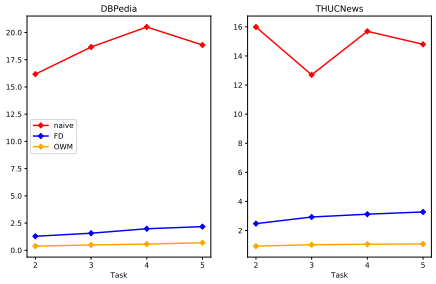

Fig. 2: The averaged Δ_i on different tasks of DBPedia and THUCNews.

We can find in different settings on CIFAR100, the phenomenon is similar with on SVHN and CIFAR10. OWM leads to very small change of feature compared to naive and FD. On the basis of OWM, SSL can further reduce the change. On two text datasets, FD can maintain the feature stable, only slight worse than OWM. We think this may attribute to the network we use for text datasets is relatively shallow (only one 1-D CNN layer before 3 layer MLP). However, the performance of FD+GFR [16] is much worse than OWM+GFR according to Table 2 in the main text.

B. Error Analysis

In Table II we list the Inter-Task Error and Inner-Task Error of other settings which are not reported in the main text due to space limit. In 5 tasks CIFAR100, 10 tasks CIFAR100, THUCNews amd DBPedia settings, the improvement of OWM+GFR and OWM+SSL+GFR over OWM is also mainly from Inter-Task Error which is consistent with the results in SVHN and CIFAR10. In 20 tasks CIFAR100, OWM+GFR is not capable of reducing Inter-Task Error thus performs similar with OWM. We think it attributes to the instability of feature as the number of task is large. However, with SSL auxiliary loss, the Inter-Task Error is reduced which can explain why OWM+SSL+GFR can improve OWM significantly.

Dataset	Method	Inter-Task Error(%)	Inner-Task Error(%)
SVHN	OWM	21.59	4.58
	OWM+GFR	20.12(-1.47)	3.99(-0.59)
	OWM+SSL+GFR	17.87(-3.72)	3.73(-0.85)
CIFAR10	OWM	41.52	3.83
	OWM+GFR	38.72(-2.80)	4.87(+0.96)
	OWM+SSL+GFR	38.06(-3.46)	4.07(+0.24)
CIFAR100 (2 tasks)	OWM	27.65	30.06
	OWM+GFR	29.31(+1.66)	28.15(-1.91)
	OWM+SSL+GFR	27.90(+0.25)	28.70(-1.36)
CIFAR100 (5 tasks)	OWM	52.66	13.59
	OWM+GFR	51.23(-1.43)	13.49(-0.10)
	OWM+SSL+GFR	50.93(-1.73)	12.84(-0.75)
CIFAR100 (10 tasks)	OWM	62.67	6.91
	OWM+GFR	60.36(-2.31)	7.53(+0.62)
	OWM+SSL+GFR	50.93(-1.73)	12.84(-0.75)
CIFAR100 (20 tasks)	OWM	69.54	3.21
	OWM+GFR	69.66(+0.12)	2.91(-0.30)
	OWM+SSL+GFR	68.21(-1.33)	3.04(-0.17)
THUCNews	OWM	15.83	4.25
	OWM+GFR	14.20(-1.63)	4.38(+0.13)
DBPedia	OWM	5.79	2.78
	OWM+GFR	4.48(-1.31)	2.64(-0.14)

TABLE II: Comparison of two types of error between OWM and proposed methods in different settings. The gaps of each error are in parentheses.

IV. Joint Training with SSL Auxiliary Task

Methods	SVHN	CIFAR10	CIFAR100
OWM	92.34	76.97	53.02
OWM+SSL	90.32	78.63	51.77

TABLE III: Accuracy of joint training on 3 datasets.

In order to exploit SSL auxiliary task, we actually conduct a data augmentation strategy in which the images obtained by rotation transformations are used to train and test. To explore whether the classifier benefits from data augmentation, we evaluate and report the accuracy of joint training on three image datasets in Table III. We find that incorporating SSL auxiliary task can not always improve the joint training performance. We conjecture that it is attributed to the difficult of training the classifier caused by introducing SSL auxiliary task. Moreover, data augmentation enlarges the subspace spanned by input data thus restricts its orthogonal subspace so that potentially has negative effects on OWM. However,

as displayed in the main text, SSL auxiliary task can indeed improve GFR based on OWM consistently in CL settings. We think this phenomenon can further support our motivation that SSL auxiliary loss can help the classifier extract more general features which is beneficial to GFR.

REFERENCES

[1] A. V. Robins, "Catastrophic forgetting in neural networks: the role of rehearsal mechanisms," in *First New Zealand International Two-Stream Conference on Artificial Neural Networks and Expert Systems*, 1993.

[2] S. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert, "icarl: Incremental classifier and representation learning," in *Proceedings of CVPR*, 2017.

[3] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proceedings of ICLR*, 2015. [Online]. Available: http://arxiv.org/abs/1412.6980

[4] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of wasserstein gans," in *Proceedings of NIPS*, 2017.

[5] F. M. Castro, M. J. Marín-Jiménez, N. Guil, C. Schmid, and K. Alahari, "End-to-end incremental learning," in *Proceedings of ECCV*, 2018, pp. 241–257.

[6] W. Hu, Z. Lin, B. Liu, C. Tao, Z. Tao, J. Ma, D. Zhao, and R. Yan, "Overcoming catastrophic forgetting for continual learning via model adaptation," in *Proceedings of ICLR*, 2019.

[7] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *JMLR*, vol. 15, no. 1, pp. 1929–1958, 2014.

[8] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of ICML*, 2010, pp. 807–814.

[9] O. Ostapenko, M. M. Puscas, T. Klein, P. Jähnichen, and M. Nabi, "Learning to remember: A synaptic plasticity driven framework for continual learning," in *Proceedings of CVPR*, 2019, pp. 11 321–11 329.

[10] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, "Reading digits in natural images with unsupervised feature learning," 2011.

[11] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," 2009.

[12] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proceedings of EMNLP*, 2014, pp. 1532–1543.

[13] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. N. Mendes, S. Hellmann, M. Morsey, P. van Kleef, S. Auer, and C. Bizer, "Dbpedia - A large-scale, multilingual knowledge base extracted from wikipedia," *Semantic Web*, vol. 6, no. 2, pp. 167–195, 2015.

[14] S. Li, Z. Zhao, R. Hu, W. Li, T. Liu, and X. Du, "Analogical reasoning on chinese morphological and semantic relations," in *Proceedings of ACL*, 2018, pp. 138–143.

[15] J. Li, M. Sun, and X. Zhang, "A comparison and semi-quantitative analysis of words and character-bigrams as features in chinese text categorization," in *Proceedings of ACL*, 2006.

[16] X. Liu, C. Wu, M. Menta, L. Herranz, B. Raducanu, A. D. Bagdanov, S. Jui, and J. v. de Weijer, "Generative feature replay for class-incremental learning," in *Proceedings of CVPR Workshops*, 2020.