Agency for Science, Technology and Research SINGAPORE

NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

Task-Agnostic Continual Learning Using Base-Child Classifiers

Pranshu Ranjan Singh[1], Saisubramaniam Gopalakrishnan[1], Qiao ZhongZheng[1,3], Ponnuthurai N. Suganthan[3], Ramasamy Savitha[1,2], ArulMurugan Ambikapathi[1,2]

[1]Institute for Infocomm Research (I²R), A*STAR, Singapore, [2]Artificial Intelligence, Analytics And Informatics (AI3), A*STAR, Singapore, [3]Nanyang Technological University (NTU), Singapore

Poster Number 3087

Continual Learning

- **What is Continual Learning?** Continual learning allows neural networks (NN) to learn
  - A sequence of tasks **incrementally**
  - **Avoid catastrophic forgetting** of preceding tasks
- **Why Continual Learning?**
  - Past task data is **no longer available** / limited storage
  - Default training on new (task) data leads to **forgetting** of old task
- **Consider the Continual Learning (CL) context:**
  - A sequence of N **classification tasks** $\{\boldsymbol{T_1}, \boldsymbol{T_2}, \dots, \boldsymbol{T_N}\}$; Each task $\boldsymbol{T_k}$ comprising of $\boldsymbol{N_k}$ samples and N_{T_k} class labels
  - Let $\boldsymbol{D_{T_k}} = \{\boldsymbol{X_1}, \boldsymbol{X_2}, \dots, \boldsymbol{X_{N_k}}\} \subset \Re^{a \times b \times c}$ be the set of training image samples for task $\boldsymbol{T_k}$, with $\boldsymbol{a}$ rows, $\boldsymbol{b}$ columns, and $\boldsymbol{c}$ channels
  - Training data for the classification task T_k can be defined as collection of tuples, each containing input and the corresponding class label, $\{< \boldsymbol{X_1}, \boldsymbol{y_1} >, \dots, <\boldsymbol{X_{N_k}}, \boldsymbol{y_{N_k}} >\}$, with class labels $\forall \boldsymbol{y_i} \epsilon \{\boldsymbol{0}, \boldsymbol{1}, \dots, \boldsymbol{N_{T_k}} - \boldsymbol{1}\}$

Key Contributions

- Design a **task-agnostic** approach that uses **Base-Child** hybrid setup to incrementally learn tasks while mitigating forgetting
- Effective **co-existence** and **retention of knowledge**, enabling intra and inter task separation using reference points
- **Boundary points sampling** for selective latent space replay
- **Automatic task identification** using distance of features from reference points
- Outperform various **state-of-the-art** regularization and replay CL algorithms in terms of **accuracy, by 50% and 7%** with homogeneous and heterogeneous tasks, respectively, in task-agnostic scenarios

Acknowledgements

We would like to thank I²R A*STAR for the financial support through SC20/19-128310-CORE.

Task Incremental Learning Using Base-Child Classifiers

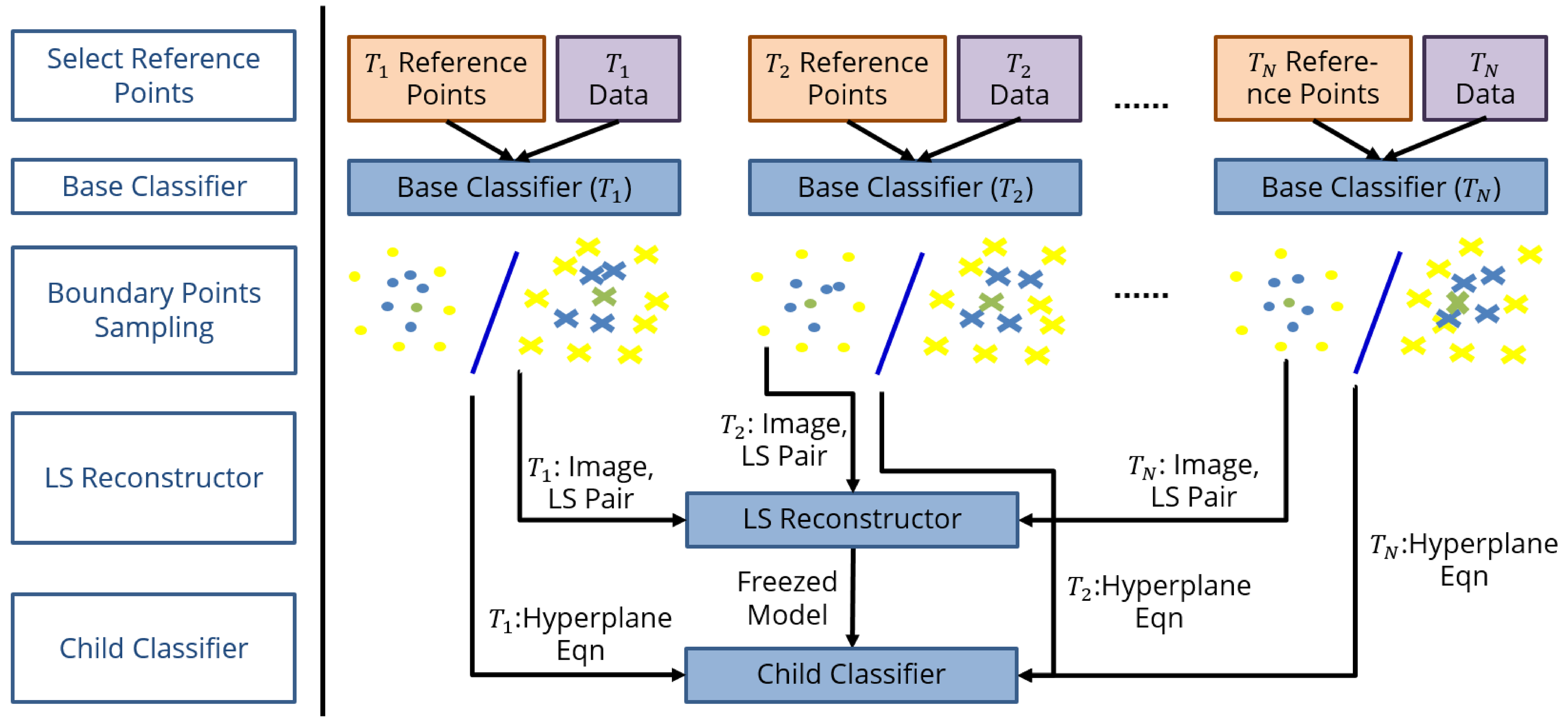

Fig. 1: *Block diagram for the proposed approach, Task-Agnostic Continual Learning using Base-Child Classifiers*

Selection of Reference Points

- Created for each class (act as **class means/centroids)** in a given task to ensure well **defined inter-class separation** as well as **inter-task separation**
- For each class in task T_k, a reference point is created; Table R_{T_k} of dimension $N_{T_k} \times s$ (latent space dimension s)

Base Classifier

- Classifier network that performs **classification exclusively for the current task $\boldsymbol{T_k}$**
- **Cross-entropy loss** (on softmax or class probabilities)

$$L_{CE} = -\frac{1}{N_k}\sum_{i=1}^{N_k} y_i.\log(\hat{y}_i)$$

- **Clustering loss** (on latent space (LS)) MAE between LS and class-specific reference point

$$L_{MAE} = \frac{1}{N_k}\sum_{i=1}^{N_k} |\widehat{ls}_i - R_{T_k}[y_i]|$$

- **Weighted loss**

$$L_{weighted} = w_1.L_{CE} + w_2.L_{MAE}$$

Boundary Points Sampling

- **Selective samples** from D_{T_k} and their **corresponding LS vectors** are stored in memory for replay to train the continual LS Reconstructor
- Locate $(\boldsymbol{X_j}, \widehat{\boldsymbol{ls}}_j)$ pairs situated at the **boundary of each class** cluster, by selecting top p% samples whose LS vectors are farthest from one another in the training set (for each class / cluster)
- Let S_{T_k} be the set of all sampled pairs, $S_{T_k} = \{(\boldsymbol{X_j}, \widehat{\boldsymbol{ls}}_j) \, \epsilon \, B_{T_k}\}$, where B_{T_k} be the set of indices lying on the class boundaries for task T_k

Latent Space Reconstructor

- S_{all} be the **collection of boundary points** (across all classes in a task and for all tasks $T_1, T_2, \dots, T_k$), $S_{all} = \bigcup_{i=1}^{k} S_{T_i}$
- LS Reconstructor is **continually trained** that takes a sample $\boldsymbol{X_j}$ as input and is optimized to provide the corresponding LS $\widehat{\boldsymbol{ls}}_j$ using MAE loss

$$L_{rec} = \frac{1}{n(S_{all})}\sum_{j=1}^{n(S_{all})} |\acute{ls}_j - \widehat{ls}_j|$$

Child Classifier

- Extension to the LS Reconstructor
- Takes the latent space embedding/vector $\acute{ls}_j$ (output of **frozen LS Reconstructor model** trained till task T_k), and **attaches the classification head** of the specified base classifier
- Compute test accuracies for tasks $T_1, T_2, \dots, T_k$

Automated Task Inference

- Child classifier requires the **knowledge of task identifier (ID)** to select the respective classification head (from base classifiers)
- Automatic ***Task Inference*** **(TI)** using Reference points **(*task-agnostic*):** Task ID corresponding to smallest distance between the current test sample' latent space vector $\acute{ls}_j$ and all reference points in $R_{T_1}, R_{T_2}, \dots, R_{T_k}$

Experimental Results and Discussion

Dataset

- **Split-Cifar10** (homogeneous); 5 classification tasks, each comprising of 2 classes (binary classification).
- **Cifar10-MNIST** (heterogeneous); 2 classification tasks, each comprising of 10 classes

Experimental Setup and Metrics

- **Network architecture for Base classifier and LS Reconstructor:** 5 conv. layers (stride 2, except first), batch-norm and ReLU activation followed by 2 dense layers (softmax for classifier)
- LS dimension, s = *128*, sampling percentage p = 10%, Loss weights (w1, w2): (0.1, 1) for Split-Cifar10 and (1, 1) for Cifar10-MNIST, Adam optimizer (with lr 0.001)
- **Evaluation:** Standard CL metrics **Average Accuracy** (ACC) and **Backward Transfer** (BWT)

Experimental Results

Table 1: Split-Cifar10 results averaged over 3 runs.

Method	Without Task ID		With Task ID	
	ACC	BWT	ACC	BWT
SFT	0.1818	−0.8669	0.6258	−0.3130
JT	0.5984	−0.1274	0.8697	0.0003
EWC [11]	0.1799	−0.8567	0.8004	−0.0615
SI [20]	0.1795	−0.8552	0.8217	−0.0419
LwF [16]	0.1854	**−0.0003**	**0.8549**	**−0.0132**
GR with VAE	0.1797	−0.7769	0.7521	−0.1211
A-GEM [14]	0.2454	−0.2834	0.8170	−0.0901
Ours - argmax	0.2196	−0.3122	-	-
Ours - TI	**0.7328**	−0.1071	0.7416	−0.1215

Table 2: Cifar10-MNIST results averaged over 3 runs.

Method	Without Task ID		With Task ID	
	ACC	BWT	ACC	BWT
SFT	0.4941	−0.6165	0.6020	−0.4007
JT	0.8115	0.0306	0.8122	0.0046
EWC [11]	0.5027	−0.5553	0.6650	−0.2324
SI [20]	0.5060	−0.5840	0.6342	−0.2957
LwF [16]	0.3177	−0.2121	**0.7726**	−0.1487
GR with VAE	0.6997	−0.1582	0.7412	−0.0998
A-GEM [14]	0.6401	−0.3371	0.7527	−0.1336
Ours - argmax	**0.7688**	**−0.0978**	-	-
Ours - TI	**0.7335**	−0.1690	0.7697	**−0.0965**

Proposed approach **outperform all baseline methods** for **without task ID** scenario on both Split-Cifar10 **(by 50%)** and Cifar10-MNIST **(by 7%)**

Conclusion

Proposed a **task-agnostic** CL classification method using Base-Child hybrid networks

- Learn **shared representations** across tasks
- Effective co-existence and **retention of knowledge**
- Enable **intra-task and inter-task separation** using reference points
- Best performance on both **homogeneous and heterogeneous tasks** in task-agnostic setting as compared to baseline methods