Learning from the Past: Continual Meta-Learning with Bayesian Graph Neural Networks

Yadan Luo,[1] **Zi Huang,**[1] **Zheng Zhang,**[2,3*] **Ziwei Wang,**[1] **Mahsa Baktashmotlagh,**[1] **Yang Yang**[4]
[1]The University of Queensland, Australia
[2]Bio-Computing Research Center, Harbin Institute of Technology, Shenzhen, China
[3]Pengcheng Laboratory, Shenzhen, China
[4]University of Electronic Science and Technology of China, China
{lyadanluol, darrenzz219, dlyyang}@gmail.com, huang@itee.uq.edu.au,
{ziwei.wang, m.baktashmotlagh}@uq.edu.au,

Abstract

Meta-learning for few-shot learning allows a machine to leverage previously acquired knowledge as a prior, thus improving the performance on novel tasks with only small amounts of data. However, most mainstream models suffer from *catastrophic forgetting* and *insufficient robustness* issues, thereby failing to fully retain or exploit long-term knowledge while being prone to cause severe error accumulation. In this paper, we propose a novel Continual Meta-Learning approach with Bayesian Graph Neural Networks (CML-BGNN) that mathematically formulates meta-learning as continual learning of a sequence of tasks. With each task forming as a graph, the intra- and inter-task correlations can be well preserved via message-passing and history transition. To remedy topological uncertainty from graph initialization, we utilize Bayes by Backprop strategy that approximates the posterior distribution of task-specific parameters with amortized inference networks, which are seamlessly integrated into the end-to-end edge learning. Extensive experiments conducted on the *mini*ImageNet and *tiered*ImageNet datasets demonstrate the effectiveness and efficiency of the proposed method, improving the performance by 42.8% compared with state-of-the-art on the *mini*ImageNet 5-way 1-shot classification task.

Introduction

A key signature of human intelligence is the ability to quickly acquire knowledge from few examples. Despite artificial intelligence (Yang et al. 2019; Bin et al. 2019; Luo et al. 2019) has made remarkable progress in wide applications, it remains challenging to perform well in situations with little available data or limited computational resources. Such a scenario is typically referred to as few-shot learning, which has attracted vast interests recently.

Rather than simply augmenting data (Hariharan and Girshick 2017) or adding regularization to compensate for the lack of data, an emerging line of work tackles few-shot learning with *meta-learning*. By leveraging previous learning experience to obtain a prior over tasks at meta-train time, the efficiency of later learning is further improved at meta-test time. Particularly, the learned prior for discovering the transferable knowledge can act as an inductive bias to minimize generalization error.

Generally, mainstream meta-learning models follow the episodic training paradigm, where the *meta-learner* extracts domain-general information among episodes so that it can assist the *task-specific learner* to recognize unlabeled samples (*query set*) based on the few labeled points (*support set*). In this way, the meta-learner can be implemented variously: as an optimizer that gathers gradient flows from different tasks (Finn, Abbeel, and Levine 2017; Nichol, Achiam, and Schulman 2018; Lee and Choi 2018); as an classification weight generator that hallucinates classifiers for novel classes (Qiao et al. 2018; Rusu et al. 2016; Gidaris and Komodakis 2019); or as a metric that measures similarity between the query and support samples (Vinyals et al. 2016; Snell, Swersky, and Zemel 2017). Nevertheless, existing meta-learning methods are far from optimal due to the lack of *relational inductive bias* modeling (Battaglia et al. 2018), thereby failing to manipulate the structured representations of intra- and inter-task relations.

Motivated to capture more interactions among instances, another line of work has explored graph structure (Garcia and Bruna 2018; Kim et al. 2019; Liu et al. 2019) or second-order statistics like covariance (Li et al. 2019b) in meta-learning framework. More concretely, Garcia and Bruna (Garcia and Bruna 2018) cast few-shot learning as the node classification problem with graph neural networks, where nodes are represented with the images in the episodes, and edges are given by a trainable similarity kernels. In the same vein, Liu (Liu et al. 2019) et al. proposed a transductive propagation network (TPN) for label propagation and thus enabled transductive inference for all query set. Alternatively, Kim et al. (Kim et al. 2019) modeled the learning as an edge labeling problem, in order to directly predict whether the associated two nodes belong to the same class.

While promising, most existing graph-based meta-learning approaches suffer from two major limitations, *i.e.,* ***catastrophic forgetting*** (Kemker et al. 2018) and ***insufficient robustness*** (Zhang et al. 2019), which make it difficult to transfer knowledge over long time spans or handle

*indicates the corresponding author.

uncertainty in graph structure. On the one hand, the problem of catastrophic forgetting is that when a meta-learner gradually encounters a sequence of learning problems, it tends to attenuate past knowledge when it learns new things. On the other hand, current graph-based models incorrectly treat the pre-defined edge initialization as the reliable topology for message-passing, in the sense that inaccurate or uncertain relationships among query-support pairs could lead to severe error accumulation through multi-layer propagation.

To address the issues mentioned above, in this paper, we propose a novel Continual Meta-Learning with Bayesian Graph Neural Networks (CML-BGNN) for few-shot classification, which is illustrated in Figure 1. To alleviate the drawback of catastrophic forgetting, we jointly model the long-term inter-task correlations and short-term intra-class adjacency with the derived continual graph neural networks, which can retain and then access important prior information associated with newly encountered episodes. Specifically, the node update block aggregates the adjacent embedding from each episode and feeds context-aware node representations to gated recurrent units, which are expected to meliorate node features with previous history. Such aggregations can be naturally chained and combined into the multiple layers to enhance model expressiveness. Moreover, as uncertainty is rife in edge initialization, we provide a Bayesian approach for edge inference so that classification weights can be dynamically adjusted for discriminating specific tasks. The conceived amortization networks approximate posterior distribution of classification weights with a Gaussian distribution defined by a mean and variance over possible values. Accordingly, task-specific parameters are sampled to mitigate the bias of node embedding and further enhance the robustness of graph neural networks. Overall, our contributions can be briefly summarized as follows:

- We propose a novel Continual Meta-Learning framework that leverages both long-term inter-task and short-term intra-task correlations for few-shot learning. Different from existing graph-based meta-learning approaches, we introduce a memory-augmented graph neural network to enable flexible knowledge transfer across episodes.
- To remedy uncertainty among query-support pairs, a Bayesian edge inference is derived by amortizing posterior inference of task-specific parameters.
- We show the effectiveness of the proposed architecture through extensive experiments on the *mini*Imagenet and *tiered*Imagenet benchmark with a 42.8% relative improvement over state-of-the-art counterparts. Regarding to robustness analysis, we perform semi-supervised learning to verify the efficiency and effectiveness of the proposed method.

Related Work

Meta-Learning

Meta-learning studies how to distill prior knowledge from past experience and enable fast adaptation to novel tasks with only a limited amount of samples. Much effort has been devoted by recent work, which can be broadly categorized into several groups: (1) *Optimization-based* methods either learn a good parameter initialization or leverage an optimizer as the meta-learner to adjust model weights. Typical examples include learning to approximate gradient descent with LSTM (Ravi and Larochelle 2017), learning model-agnostic initial parameters (Finn, Abbeel, and Levine 2017) and its variants with probabilistic estimation (Finn, Xu, and Levine 2018; Yoon et al. 2018; Li et al. 2017), first-order approximation (Nichol, Achiam, and Schulman 2018), layer selection (Lee and Choi 2018), learner update direction and learning rate learning (Li et al. 2017), and relation embedding (Rusu et al. 2019); (2) *Generation-based* methods learn to augment few-shot data with a generative meta-learner (Wang et al. 2018), or learn to predict classification weights for novel classes (Rusu et al. 2016; Qiao et al. 2018; Gidaris and Komodakis 2019); (3) *Metric-based* approaches address the few-shot classification problem by learning a proper distance metrics as the meta-learner, such as cosine similarity (Vinyals et al. 2016), euclidean distance to class prototypes (Snell, Swersky, and Zemel 2017; Ren et al. 2018), ridge regression (Bertinetto et al. 2019), relation network (Sung et al. 2018), task attention (Yan, Zhang, and He 2019), category traversal (Li et al. 2019a), and graph modeling (Garcia and Bruna 2018; Liu et al. 2019; Li et al. 2019b; Kim et al. 2019). Rather than purely relying on graph-based metric learning, our methodology exploits long-term information from previous tasks and jointly models the topological uncertainty.

Catastrophic Forgetting

Catastrophic forgetting (Kemker et al. 2018) has been a long-standing issue in machine learning community due to the stability-plasticity dilemma (Ditzler et al. 2015). In recent literature, a number of methods have been proposed on the basis of continual learning, which can be roughly subdivided into the following groups. Regularization approaches (Li and Hoiem 2016) alleviate catastrophic forgetting by imposing constraints on the update of the neural weights in order to prevent "overwriting" what was previously encoded. Alternatively, in ensemble algorithms (Rusu et al. 2016), the architecture itself is altered to accommodate new tasks by retraining a pool of pre-trained models. In dual-memory algorithms (Soltoggio, Stanley, and Risi 2018), one estimates the distribution of the old data either by saving a small fraction of the original dataset into a memory buffer or by training a generator to mimic the lost data and labels (Shin et al. 2017). Being more related to the last group, our work, *for the first time*, tackles the catastrophic forgetting in a meta-learning framework and validates its effectiveness and efficiency on practical tasks.

Background and Problem Definition

Given the training set $\mathcal{D}_{train}$, the goal is to learn the model $f : x \rightarrow y$, which is capable of generalizing well to the unseen test set $\mathcal{D}_{test}$, where $\mathcal{D}_{train} \cap \mathcal{D}_{test} = \varnothing$. Meta-learning approaches commonly adopt *episodic training* strategy to minimize the generalization error across a series of tasks

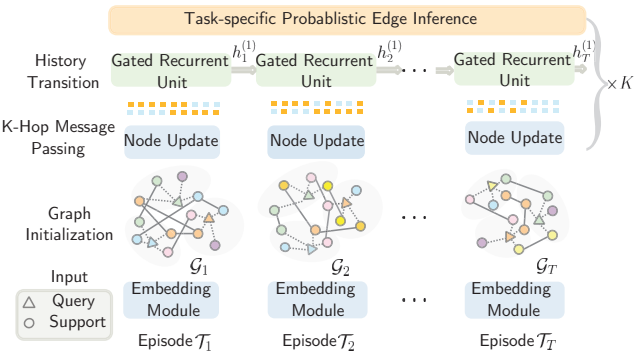

Figure 1: The general flowchart of the proposed CML.

$\mathcal{T} \sim p(\mathcal{T})$ that are randomly sampled from a task distribution. Specifically, a N-way, K-shot classification setting is used for both training and testing stage, where N indicates the number of unique classes in each episode and K denotes the number of training samples per class. Each episode or task $\mathcal{T} = (\mathcal{S}, \mathcal{Q})$ is composed of the *support set* $\mathcal{S} = \{(x_i, y_i)\}_{i=1}^{N\times K}$ and the *query set* $\mathcal{Q} = \{(\tilde{x}_i, \tilde{y}_i)\}_{i=1}^{K}$. To discover the commonalities and variability across tasks, we decouple the model f into two sub-modules, *i.e.*, the feature learner $f_\Theta(\cdot;\Theta)$ and the task-dependent classifiers $\{f_\psi(\cdot;\psi_t)\}_{t=1}^{\mathcal{T}}$, where Θ indicates the shared parameters that suit for all tasks and $\{\psi_t\}_{t=1}^{|\mathcal{T}|}$ indicate the task-specific parameters. To mathematically illustrate the proposed continual meta-learning procedure, we firstly recall the unified definition of existing meta-learning models: as discussed in literature (Grant et al. 2018; Finn, Xu, and Levine 2018), meta-learning can be viewed as an approximate inference for the posterior given the following definition.

Definition 1. ***Meta-Learning*** *Given the task $\mathcal{T}$ sampled from task distribution $p(\mathcal{T})$, the posterior predictive distribution for query points is calculated as,*

$$p(\widetilde{Y}|\widetilde{X}, \mathcal{S}; \Theta) = \prod_{i=1}^{|\mathcal{Q}|} p(\tilde{y}_i|\tilde{x}_i, \mathcal{S}_i; \Theta)$$
$$= \prod_{i=1}^{|\mathcal{Q}|} \int p(\tilde{y}_i|\tilde{x}_i, \psi_i; \Theta) p(\psi_i|\mathcal{S}_i; \Theta)\, \mathrm{d}\psi_i \approx p(\tilde{y}_i|\tilde{x}_i, \psi_i^*),$$

where ψ_i^* is the maximum a posteriori (MAP) value of ψ_i, which can be obtained via point estimates.

For instance, the *optimization-based* meta-learning regards all model parameters as ψ and forms a point estimate by taking several steps of gradient descent initialized at ψ_0 and learning rate η, *i.e.,*

$$\psi^*(S; \Theta) = \psi_0 + \eta \frac{\partial}{\partial \psi} \log \sum_{j=1}^{|S|} p(y_j|x_j, \psi; \Theta). \quad (1)$$

While, the *generation-based* meta-learning focuses on estimating classification weight vector ψ for novel classes, given the initial value ψ_0 trained on $\mathcal{D}_{train}$ and the learning rate η, *i.e.,*

$$\psi^*(S, \Theta) = \psi_0 + \eta \frac{\partial}{\partial \psi} \log \sum_{j=1}^{|S|} p(y_j|x_j, G(x_j|\psi); \Theta), \quad (2)$$

where $G(\cdot|\cdot)$ is the learnable weight generator. The *metric-based* meta-learning takes the parameters from the top layer of neural networks as $\psi = \{w_c, b_c\}_{c=1}^{C}$ for all C classes, by averaging the top-layer activations for each class c, *i.e.,*

$$\psi^*(S, \Theta) = \{\mu_c, -\frac{\| \mu_c \|^2}{2}\}_{c=1}^{C},$$
$$\mu_c = \frac{1}{k_c} \sum_{j=1}^{k_c} f_\Theta(x_j^{(c)}), \quad (3)$$

where k_c denotes the number of samples belonging to class c, and $f_\Theta(\cdot)$ indicates the embedding network.

However, the current meta-learning suffers from catastrophic forgetting and lacks for uncertainty estimation. Without fixing these problems, a single deep model will be incapable of adapting itself to a long-run learning, since it forgets the old messages when it deals with new things. Therefore, we generalize the meta-learning to a preferable continual way, and give the following definition.

Definition 2. ***Continual Meta-Learning*** *Given the task $\mathcal{T}$ sampled from task distribution $p(\mathcal{T})$, the posterior predictive distribution for query points is*

$$p(\widetilde{Y}|\widetilde{X}, \mathcal{S}; \Theta) = \prod_{i=1}^{|\mathcal{Q}|} p(\tilde{y}_i|\tilde{x}_i, \mathcal{S}_{1:i}; \Theta)$$
$$= \prod_{i=1}^{|\mathcal{Q}|} \int p(\tilde{y}_i|\tilde{x}_i, h_i, \psi_i; \Theta) p(h_i|S_{1:i-1}; \Theta) p(\psi_i|\mathcal{S}_i; \Theta)\, \mathrm{d}\psi_i,$$

where h_i indicates the history knowledge that gives transition of long-term memory.

The first term can be interpreted as given a novel sample $\tilde{x}_i$, its respective label $\tilde{y}_i$ not only conditions on the current task ψ_i but also history information h_i, which naturally reuses supervision without pilling up complexity. The second term presents information updating and resetting from related tasks, which is jointly learned with shared parameter Θ in the continual graph neural networks, which will be elaborated in the next section. Lastly, to ensure a tractable likelihood, in the last term, the distribution $p(\psi_i|S_i; \Theta)$ over classifier weights is approximated with a few steps of Bayes by Backprop.

Proposed Approach

In this section, we detail two major components of the proposed method, the continual graph neural networks and Bayes by Backprop procedure for edge inference.

Continual Graph Neural Networks

As shown in Figure 1, the support $\mathcal{T} = (\mathcal{S}, \mathcal{Q})$ (shown in circle) and query data $\mathcal{Q} = \{(\tilde{x}_i, \tilde{y}_i)\}_{i=1}^{K}$ (shown in triangle) in each episode can form an undirected acyclic graph

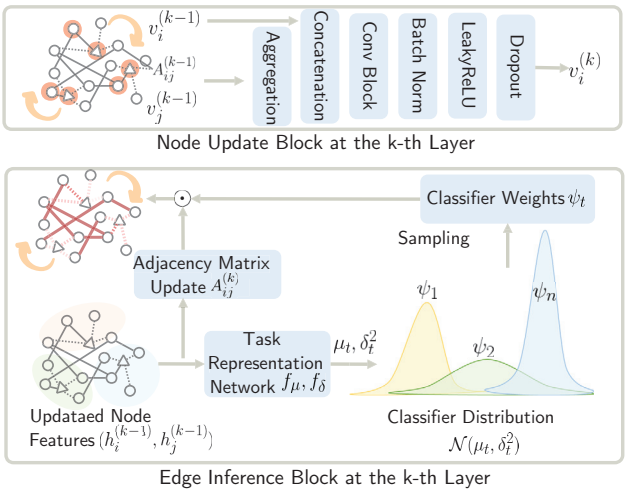

Figure 2: An illustration of the node update and edge inference block at the k-th layer.

$\mathcal{G} = (\mathcal{V}, \mathcal{E})$. Each vertex $v_i \in \mathcal{V}$ is associated with a node embedding vector $\mathcal{X} = \{x_i\}_{i=1}^{|\mathcal{S}|+|\mathcal{Q}|}$ and each edge $e_{ij} \in \mathcal{E}$ represents interaction between nodes. The adjacency matrix $\mathcal{A} = \{A_{ij}\}_{i,j=1}^{|\mathcal{V}|}$ is defied as semantic similarity between two connected nodes v_i and v_j, which will be updated dynamically. Considering the given label of support set, the adjacency matrix can be initialized as,

$$A_{ij}^{(0)} = \begin{cases} 1 & \text{if } x_i, x_j \in \mathcal{S} \text{ and } y_i = y_j, \\ 0 & \text{if } x_i, x_j \in \mathcal{S} \text{ and } y_i \neq y_j, \\ 0.5 & \text{otherwise,} \end{cases} \tag{4}$$

where support-support pairs are initialized *w.r.t* their labels and query-support pairs are softly assigned with an uncertain value. Even though ambiguity is explicitly injected, we remedy the noise via probabilistic edge inference presented in the next section. For efficient implementation, we split the meta-train $\mathcal{D}_{train}$ and meta-test dataset $\mathcal{D}_{test}$ into several sequences of episodes $\{\mathcal{G}_t\}_{t=1}^{T}$, which is learned by node updating (see Figure 2), history transition and edge inference module consecutively, where T is the length of the sequence.

Node Interactions Modeling To make each node aggregate more information from neighbors K hops away, the proposed graph model stacks K aggregation blocks. Following the generic propagation rule, the node vectors in arbitrary graph $\mathcal{G}_t$ at the k-th layer can be updated as,

$$\begin{aligned} \hat{v}_i^{(k-1)} &= \sum_{j \in \mathcal{N}(i)} (v_i^{(k-1)} A_{ij}^{(k-1)}), \\ v_i^{(k)} &= f_n([v_i^{(k-1)}; \hat{v}_i^{(k-1)}]; \theta_n), \end{aligned} \tag{5}$$

where $\mathcal{N}(i)$ indicates the neighbor set of the node v_i, $[\cdot;\cdot]$ is the concatenation operation and $f_n(\cdot;\theta_n)$ is a transformation block consisting of two convolutional layers, one LeakyReLU activation and one dropout layer. The node embedding is initialized with the extracted representation from the backbone embedding model, *i.e.,* $v_i^{(0)} = x_i$. Here we leave out the layer mark $^{(k)}$ in the next section.

Task History Transition After receiving messages from current episode $\mathcal{G}_t$, each node embedding v_i^t is further transformed with a gate updater. Different from the Gated Graph Neural Network (Li et al. 2016) that utilizes the gate updater to extend the depth of graph neural networks for the same batch of data, we feed different embedding at each time step in order to capture the long-term task correlations. Specifically, the hidden state for each node is transferred as follows,

$$\begin{aligned} z_i^t &= \sigma(W_z v_i^t + U_z h_i^{t-1} + b_z), \\ r_i^t &= \sigma(W_r v_i^t + U_r h_i^{t-1} + b_r), \\ \tilde{h}_i^t &= \tanh(W_h v_i^t + U_h (r_i^t \odot h_i^{t-1}) + b_h), \\ h_i^t &= \tilde{h}_i^t \odot z_i^t + h_i^{t-1} \odot (1 - z_i^t), \end{aligned} \tag{6}$$

where h_i^t is the updated feature of node v_i^t, σ is the sigmoid function, W_z, U_z, W_r, U_r, W_h, U_h are learnable weights, and b_z, b_r, b_h are biases of the updating function. z_i^t and r_i^t are update gate vector and reset gate vector, respectively. Here we let all parameters as θ_h for shorthand. The hidden state h_i^0 is initialized as a zero vector.

Adjacency Feature Update After T-step hidden transition, we can obtain a set of final node representations at the k-th layer. In this way, the adjacency features A_{ij} at the k-th layer is calculated as,

$$\begin{aligned} \widetilde{A}_{ij}^{(k)} &= f_e(\|h_i^{(k-1)} - h_j^{(k-1)}\|; \theta_e), \\ A_{ij}^{(k)} &= D^{-\frac{1}{2}} \widetilde{A}_{ij}^{(k)} D^{-\frac{1}{2}}, \end{aligned} \tag{7}$$

where D is the degree matrix of adjacency matrix, $f_e(\cdot;\theta_e)$ is the non-linear transformation network parameterized by θ_e, which includes four convolutional blocks, a batch normalization, a LeakyReLU activation and a dropout layer.

Bayes by Backprop for Edge Inference

Though multiple graph aggregation layers, we aim to infer a full predictive distribution over the unknown query labels relying on distributional Bayesian Decision Theory (Finn, Xu, and Levine 2018; Gordon et al. 2019). Notably, we amortize the posterior distribution of the classification weights $p(\psi)$ as q_ψ to enable quick prediction at the meta-test stage and learn parameters by minimizing the average expected loss over tasks *i.e.,* $\psi^* = \arg\min_\psi \mathcal{L}_B$, where

$$\begin{aligned} \mathcal{L}_B &= \mathbb{E}[D_{KL}(p(\tilde{y}|\tilde{x}, \mathcal{S}) \parallel q_\psi(\tilde{y}|\tilde{x}, \mathcal{S}))], \\ \psi^* &= \arg\max_\psi \mathbb{E}[\log \int p(\tilde{y}|\tilde{x}, \psi) q_\psi(\psi|\tilde{x}, \mathcal{S}) \, \mathrm{d}\psi]. \end{aligned} \tag{8}$$

To ensure likelihood tractable, we use a factorized Gaussian distribution for $q_\psi(\psi|\tilde{x}, \mathcal{S}; \Theta)$ with means and variances set by the amortization network,

$$\begin{aligned} q_\psi(\psi|\tilde{x}, \mathcal{S}; \Theta) = \prod_{t=1}^{|\mathcal{T}|} q_\psi(\psi_t|\{h_i\}_{i=1}^{|\mathcal{T}|}) = \prod_{t=1}^{|\mathcal{T}|} \mathcal{N}(\mu_t, \delta_t^2), \\ \mu_t = f_\mu(\{h_i\}_{i=1}^{|\mathcal{T}|}; \theta_\mu), \, \delta_t^2 = f_\delta(\{h_i\}_{i=1}^{|\mathcal{T}|}; \theta_\delta). \end{aligned} \tag{9}$$

Algorithm 1 Meta-training of the Proposed CML-BGNN.

Inputs:
Task distribution of $\mathcal{D}_{train}$: $p(\mathcal{T})$;
Outputs:
Model Weights: $\Theta = \{\theta_n, \theta_h, \theta_e, \theta_\mu, \theta_\delta\}$;
Initialize:
Hyper-parameters: M, K, γ;
Adjacency matrix A; Hidden states h;
Visual features $\mathcal{X}$ for all tasks;
Minibatch size m and learning rate η;
for M iterations **do**
  Sample batch of tasks $\mathcal{T} \sim p(\mathcal{T})$;
  Message passing and update node representations in Equation (5) and Equation (6);
  Compute adjacency matrix in Equation (7);
  Generate task-specific parameter μ_t, δ_t, sample $\psi_t \sim \mathcal{N}(\mu_t, \delta_t^2)$;
  Compute prediction $p(\tilde{y}|\tilde{x}, \psi_t)$ of query samples;
  Update parameters Θ by descending stochastic gradients according to Equation (11).
end for

With the generated posterior distribution, we can adaptively transform the predictive logits from the k-th layer based on the learned adjacency matrix and sampled classifier weights $\psi_t = \{W_t, b_t\}$ for each specific task,

$$p(\tilde{y}^{(k)}|\tilde{x}, h^{(k)}, \psi_t) = M_q \odot \sigma(W_t A^{(k)} + b_t) \odot M_s,$$
$$W_t,\ b_t \sim \mathcal{N}(\mu_t, \delta_t^2), \tag{10}$$

where σ and $\odot$ indicate the sigmoid function and Hadamard product, respectively. M_q indicates the mask matrix to select query predictions, where the value is assigned 1 when the row index belongs to query set and 0 otherwise. Similarly, M_s selects columns when the column index belongs to support set. During the meta-training time, the proposed model is optimized by minimizing the binary cross-entropy loss of query edges and Bayes by Backprop loss, *i.e.,*

$$\mathcal{L}_E = -\sum_{k=1}^{K}\sum_{i=1}^{|\mathcal{Q}|} \tilde{y}_i \log(p(\tilde{y}_i^{(k)})) + (1-\tilde{y}_i)\log(1-p(\tilde{y}_i^{(k)})),$$
$$\Theta^* = \arg\min_{\Theta} \mathcal{L}_E + \gamma \mathcal{L}_B, \tag{11}$$

where shared parameters $\Theta = \{\theta_n, \theta_h, \theta_e, \theta_\mu, \theta_\delta\}$ are jointly optimized. the loss aggregates all predictions from different layers. The overall algorithm for meta-training is shown in Algorithm 1.

Experiments

Datasets

For fair comparisons with state-of-the-art baselines, we conduct extensive experiments on two benchmark few-shot classification datasets:

***mini*ImageNet** is the subset of the ILSVRC-12 dataset, where 600 images for each of 100 classes are randomly chosen to be the part to the dataset. We follow the class split used by (Ravi and Larochelle 2017), where 64 classes are used for training, 16 for validation, and 20 for testing. All the input images have the size of $84 \times 84 \times 3$.

***tiered*ImageNet** is a larger subset of ILSVRC-2012, which contains 608 classes in 34 higher-level categories sampled from the high-level nodes in the ImageNet. The standard split includes 351 classes for training, 97 classes for validation, and 160 classes for testing. The average number of images in each class is $1,281$.

Baselines

We compare our approach with the following baseline methods to justify its effectiveness:

Optimization-based: Meta-learner LSTM (Ravi and Larochelle 2017), MAML (Finn, Abbeel, and Levine 2017), REPTILE (Nichol, Achiam, and Schulman 2018), Meta-SGD (Li et al. 2017), SNAIL (Mishra et al. 2018), LEO (Rusu et al. 2019).

Generation-based: PLATIPUS (Finn, Xu, and Levine 2018), VERSA (Gordon et al. 2019), LwoF (Rusu et al. 2016), Param_Predict (Qiao et al. 2018), wDAE (Gidaris and Komodakis 2019).

Metric-based: Matching Net (Vinyals et al. 2016), Prototypical Net (Snell, Swersky, and Zemel 2017), Relation Net (Sung et al. 2018), TADAM (Oreshkin, López, and Lacoste 2018), CTM (Li et al. 2019a).

Graph-based: GNN (Garcia and Bruna 2018), CovaMNet (Li et al. 2019b), TPN (Liu et al. 2019), EGNN (Kim et al. 2019).

Implementation Details

Our source code[1] is implemented based on Pytorch. All experiments are conducted on a server with two GeForce GTX 1080 Ti and two GTX 2080 Ti GPUs.

Module Architecture. Despite the generality of backbone embedding module, we adopt the same architecture used in some recent work (Sung et al. 2018; Kim et al. 2019). Specifically, the network consists of four convolutional blocks including a 2D covolutional layer with a 3×3 kernal, a batch normalization, a 2×2 max-pooling and a LeakyReLU activation. Regarding to specification of recurrent units, the dimension of hidden states and all embedding size are fixed to 96. We fix the number of hidden states to 8.

Parameter Settings. The mini-batch size for all graph-based models is 80 and 64 for 1-shot and 5-shot experiments, respectively. The proposed model was trained by Adam optimizer with an initial learning rate η of 1×10^{-3} and weight decay of 1×10^{-6}. The dropout rate is set to 0.3 and the loss coefficient γ is set to 1. We report the final results of the proposed model trained with 70K and 160K iterations on *mini*ImageNet and *tiered*ImageNet.

Comparisons with State-of-The-Art

To verify the effectiveness of our proposed continual meta-learning model, we compare it with state-of-the-art meta-learning methods on the *mini*ImageNet and *tiered*ImageNet

[1] https://github.com/Luoyadan/BGNN-AAAI

Table 1: The 5-way 1-shot and 5-shot classification accuracies (%) on the test split of the *mini*ImageNet dataset, with 95% confidence interval. † indicates our re-implementation. "w/o" indicates without.

Models	Backbone	1-shot	5-shot
Optimization-based			
Meta-learner LSTM	Conv4	43.44 ± 0.77	60.60 ± 0.71
MAML	Conv4	48.70 ± 1.84	63.10 ± 0.92
REPTILE	Conv4	49.97 ± 0.32	65.99 ± 0.58
Meta-SGD	Conv4	50.47 ± 1.87	64.03 ± 0.94
SNAIL	ResNet-12	55.71 ± 0.99	68.88 ± 0.92
LEO	WRN-28	61.76 ± 0.08	77.59 ± 0.12
Generation-based			
PLATIPUS	Conv4	50.13 ± 1.86	-
VERSA	Conv4	53.40 ± 1.82	67.37 ± 0.86
LwoF	Conv4	56.20 ± 0.86	72.81 ± 0.62
Param_Predict	WRN-28	59.60 ± 0.41	73.74 ± 0.19
wDAE	WRN-28	61.07 ± 0.15	76.75 ± 0.11
Metric-based			
Matching Net	Conv4	43.56 ± 0.84	55.31 ± 0.73
Prototypical Net	Conv4	49.42 ± 0.78	68.20 ± 0.66
Relation Net	Conv4	50.40 ± 0.80	65.30 ± 0.70
TADAM	ResNet-12	58.50 ± 0.30	76.70 ± 0.30
CTM	Conv4	62.05 ± 0.55	78.63 ± 0.06
Graph-based			
GNN	Conv4	50.33 ± 0.36	66.41 ± 0.63
CovaMNet	Conv4	51.19 ± 0.76	67.65 ± 0.63
TPN	Conv4	53.75 ± 0.86	69.43 ± 0.67
EGNN	Conv4	-	76.37 ± 0.30
EGNN†	Conv4	58.65 ± 0.55	75.25 ± 0.49
Ours			
CML-BGNN w/o C	Conv4	63.74 ± 0.63	79.36 ± 0.57
CML-BGNN w/o B	Conv4	87.15 ± 0.24	91.21 ± 0.19
CML-BGNN	Conv4	**88.62** ± 0.43	**92.69** ± 0.31

Table 2: The 5-way 1-shot and 5-shot classification accuracies (%) on the test split of the *tiered*ImageNet dataset, with 95% confidence interval. † indicates the re-implementation. "w/o" indicates without.

Models	Backbone	1-shot	5-shot
Optimization-based			
MAML	Conv4	51.67 ± 1.81	70.30 ± 0.08
REPTILE	Conv4	52.36 ± 0.23	71.03 ± 0.22
Meta-SGD	Conv4	62.95 ± 0.03	79.34 ± 0.06
LEO	WRN-28	66.33 ± 0.05	81.44 ± 0.09
Generation-based			
LwoF	Conv4	50.90 ± 0.46	66.69 ± 0.36
wDAE	WRN-28	68.18 ± 0.16	83.09 ± 0.12
Metric-based			
Matching Net†	Conv4	54.02 ± 0.00	70.11 ± 0.00
Prototypical Net	Conv4	53.31 ± 0.89	72.69 ± 0.74
Relation Net	Conv4	54.48 ± 0.93	71.32 ± 0.78
CTM	Conv4	64.78 ± 0.11	81.05 ± 0.13
Graph-based			
GNN	Conv4	43.56 ± 0.84	55.31 ± 0.73
TPN	Conv4	57.53 ± 0.96	72.85 ± 0.74
EGNN-3	Conv4	-	80.15 ± 0.30
EGNN-1†	Conv4	61.04 ± 0.45	73.91 ± 0.40
EGNN-2†	Conv4	64.64 ± 0.50	79.80 ± 0.43
EGNN-3†	Conv4	65.30 ± 0.53	82.39 ± 0.43
Ours			
CML-BGNN-1	Conv4	85.91 ± 0.51	89.58 ± 0.28
CML-BGNN-2	Conv4	88.02 ± 0.50	91.50 ± 0.25
CML-BGNN-3	Conv4	**88.87** ± 0.51	**92.77** ± 0.28

datasets. Here we report the best performance for every model in Table 1 and Table 2, along with the specifications of the backbone embedding models for feature extraction. **Conv4** refers to a 4-layer convolutional network, **ResNet-12** (He et al. 2016) denotes 4 layer blocks of depth 3 with $3\times$ 3 kernels and short connections, and **WRN-28** is a 28-layer wide residual network. Generally, a deeper embedding network will lead to a better classification performance yet with a risk of overfitting. From Table 1, we can observe that our **CML-BGNN** equipped with three graph layers surpasses all compared meta-learning methods with a large margin, especially in the challenging scenario of 1-shot learning. More concretely, the proposed model with the basic conv4 embedding structure gains 43.5%, 45.1%, 42.8%, 51.1% relative improvements over the previous best optimization-based **LEO** (Rusu et al. 2019), generation-based **wDAE** (Gidaris and Komodakis 2019), metric-based **CTM** (Li et al. 2019a) and graph-based methods **EGNN** (Kim et al. 2019) in a 5-way 1-shot *mini*ImageNet experiment, respectively. This is mainly owing to the learned history transition, which reinforces the memory of rare samples and correlations between classes. Furthermore, we re-implemented the most powerful graph-based baseline EGNN with mini-batch size of 80 for fairness and present a detailed comparison in boxplots. All parameters are randomly initialized in three trials with fixed seeds 111, 222, 333 for reproducibility. As depicted in Figure 3, the absolute value of validation accuracy either in 1-shot or 5-shot setting tends to go up as training iterations increase. The proposed method reaches the peaks at an early stage and achieves a much higher performance, yet showing the sensitivity to seed selection in the case of 5-way 5-shot classification. We infer this variance is mainly introduced by edge inference sampling, which can be alleviated by averaging predictions from multiple sampling. From Table 2, we observe that our re-implemented EGNN obtains better performance (as indicated with †) by enlarging the batch size from 40 to 80. This phenomenon consistently verifies that task correlations are more likely to contribute positively to few-shot learning.

Ablation Study

Effect of Components. The major ablation results regarding to CML-BGNN with different components on *mini*ImageNet dataset are shown in gray blocks of Table 1. All variants are trained with three graph layers, mini-batch size of 80. Removing the history transition module, the variant **CML-BGNN w/o C** can only mine the pattern from local neighborhood without maintaining related prior information for reference, thus inevitably leading to a inferior performance, e.g., averagely decreasing 5-way 1-shot performance from 88.62% to 63.74%. **CML-BGNN w/o B** indicates the variant of our proposed model that directly utilizes the adjacency matrix to predict query labels without inferring task-specific parameters. Accordingly, the classification accuracy suffers a slight drop on both datasets, e.g., from 92.69% to 91.21% in a 5-way 5-shot setting, which demon-

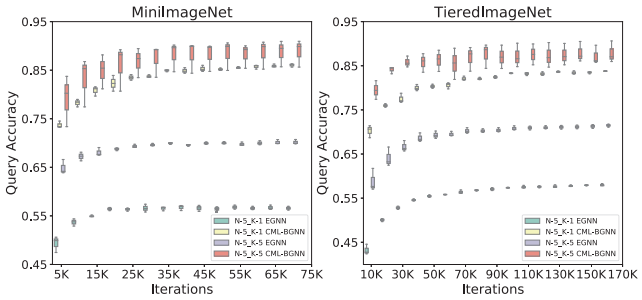

Figure 3: The 5-way 1-shot and 5-shot accuracies of query classification on the validation split of *mini*ImageNet (*left*) and *tiered*ImageNet dataset (*right*).

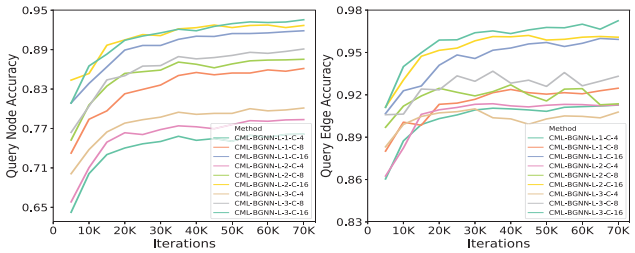

Figure 4: The 5-way 1-shot accuracies (%) of query classification on the validation split of *mini*ImageNet with different number of hidden states. Best viewed in color.

strates the necessity of the full CML-BGNN formulations.

Effect of GNN's Depth. In addition to the evaluation for investigating the impact of GNN's depth, we test our model in both 5-way 1-shot and 5-shot with different depth of graph neural networks on both miniImageNet (shown in Table 3) and *mini*ImageNet (shown in Table 2) dataset. Generally, larger depth enables node to learn from a global perspective and thus enhances the expressive power of graph neural networks. For instance, the proposed **CML-BGNN**, **EGNN** and **GNN** equipped with a 3-layer structure respectively improve the classification accuracy by 3.4%, 6.4% and 4.3% *w.r.t* 5-way 1-shot classification, compared with the one with one-layer structure.

Effect of Number of Hidden States. In order to study the impact of number of hidden states in history transition module, we compare nine variants of our model on both datasets and show validation curves in terms of node classification accuracy and edge binary classification results in Figure 4. The **CML-BGNN-L-{1,2,3}-C-16** indicates the variants that leverage unrolled gated recurrent units with 16 time steps to transfer history messages with different number of graph layers, which significantly outperform other variants with fewer hidden states **CML-BGNN-L-{1,2,3}-C-{4,8}**. This confirms that the augmented memory module effectively enhances node representation learning by bridging the prior task learning regardless of the model depth.

Table 3: 5-Way 5-shot and 1-shot classification accuracies (%) on *mini*ImageNet dataset with different depths of graph neural networks. † indicates our re-implementation.

	Methods	Layer-1	Layer-2	Layer-3
1-shot	GNN†	48.25 ± 0.65	49.17 ± 0.35	50.32 ± 0.41
	EGNN†	55.13 ± 0.44	57.47 ± 0.53	58.65 ± 0.55
	CML-BGNN	**85.73** ± 0.47	**87.67** ± 0.47	**88.62** ± 0.43
5-shot	GNN†	65.58 ± 0.34	67.21 ± 0.49	66.99 ± 0.43
	EGNN†	67.76 ± 0.42	74.70 ± 0.46	75.25 ± 0.49
	CML-BGNN	**90.85** ± 0.27	**91.63** ± 0.26	**92.69** ± 0.31

Table 4: Semi-supervised few-shot classification accuracies (%) on *mini*ImageNet with 95% confidence intervals.

Methods	5-way 5-shot		
	20%-labeled	40%-labeled	100%-labeled
GNN-LabeledOnly	50.33 ± 0.36	$56.91\% \pm 0.42$	66.41 ± 0.63
GNN-Semi	52.45 ± 0.88	$58.76\% \pm 0.86$	66.41 ± 0.63
EGNN-LabeledOnly†	58.65 ± 0.55	$56.91\% \pm 0.00$	75.25 ± 0.49
EGNN-Semi†	63.62 ± 0.00	$64.32\% \pm 0.00$	75.25 ± 0.49
CML-BGNN-LabeledOnly	84.37 ± 0.54	$88.62\% \pm 0.29$	**92.69** ± 0.31
CML-BGNN-Semi	**88.95** ± 0.32	**89.70**% ± 0.32	**92.69** ± 0.31

Robustness Evaluation by Semi-supervised Learning

To quantitatively analyze the model capacity of handling uncertainty, we conduct 5-way 5-shot semi-supervised experiments on *mini*ImageNet dataset and showcase major results in Table 4. In this semi-supervised regime, support data is partially labeled while balanced across all classes, which poses a greater challenge of modeling uncertain relationships between labeled and unlabeled samples. In particular, the 20%**-labeled** column indicates that each episode contains 4 labeled support instances and 1 unlabeled instance. Here we use **LabeledOnly** to denote the strategy with only labeled support samples, and **Semi** presents training with both labeled and unlabeled data. By comparing the results with all graph-based counterparts, the proposed method greatly outperforms with a large margin (88.95% vs 63.62% and 52.45%, when 20% are labeled). The superior performance results from our uncertainty modeling, which effectively adapts the noise and misguidance from adjacency initialization with task-specific parameters.

Conclusion

In this work, we propose a continual meta-learning model with Bayesian graph neural networks for few-shot classification problem. Towards preserving more history messages associated with related tasks, the proposed CML-BGNN mines the prior knowledge patterns by updating a memory-augmented graph neural network and handles topological uncertainty with Bayes by Backprop. Distinguishing our work from conventional graph-based meta-learning methods, it naturally alleviates the catastrophic forgetting and insufficient robustness issues and thus encourages an efficient adaptation and generalization to novel tasks.

Acknowledgement. This work was partially supported by ARC DP 190102353.

References

Battaglia, P. W.; Hamrick, J. B.; Bapst, V.; Sanchez-Gonzalez, A.; Zambaldi, V. F.; Malinowski, M.; Tacchetti, A.; et al. 2018. Relational inductive biases, deep learning, and graph networks. *CoRR* abs/1806.01261.

Bertinetto, L.; Henriques, J. F.; Torr, P. H. S.; and Vedaldi, A. 2019. Meta-learning with differentiable closed-form solvers. In *ICLR*.

Bin, Y.; Yang, Y.; Tao, C.; Huang, Z.; Li, J.; and Shen, H. T. 2019. MR-NET: exploiting mutual relation for visual relationship detection. In *AAAI*.

Ditzler, G.; Roveri, M.; Alippi, C.; and Polikar, R. 2015. Learning in nonstationary environments: A survey. *IEEE Comp. Int. Mag.* 10(4):12–25.

Finn, C.; Abbeel, P.; and Levine, S. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *ICML*.

Finn, C.; Xu, K.; and Levine, S. 2018. Probabilistic model-agnostic meta-learning. In *NeurIPS*.

Garcia, V., and Bruna, J. 2018. Few-shot learning with graph neural networks. In *ICLR*.

Gidaris, S., and Komodakis, N. 2019. Generating classification weights with GNN denoising autoencoders for few-shot learning. In *CVPR*.

Gordon, J.; Bronskill, J.; Bauer, M.; Nowozin, S.; and Turner, R. E. 2019. Meta-learning probabilistic inference for prediction. In *ICLR*.

Grant, E.; Finn, C.; Levine, S.; Darrell, T.; and Griffiths, T. L. 2018. Recasting gradient-based meta-learning as hierarchical bayes. In *ICLR*.

Hariharan, B., and Girshick, R. B. 2017. Low-shot visual recognition by shrinking and hallucinating features. In *ICCV*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *CVPR*.

Kemker, R.; McClure, M.; Abitino, A.; Hayes, T. L.; and Kanan, C. 2018. Measuring catastrophic forgetting in neural networks. In *AAAI*.

Kim, J.; Kim, T.; Kim, S.; and Yoo, C. D. 2019. Edge-labeling graph neural network for few-shot learning. In *CVPR*.

Lee, Y., and Choi, S. 2018. Gradient-based meta-learning with learned layerwise metric and subspace. In *ICML*.

Li, Z., and Hoiem, D. 2016. Learning without forgetting. In *ECCV*.

Li, Y.; Tarlow, D.; Brockschmidt, M.; and Zemel, R. S. 2016. Gated graph sequence neural networks. In *ICLR*.

Li, Z.; Zhou, F.; Chen, F.; and Li, H. 2017. Meta-sgd: Learning to learn quickly for few shot learning. *CoRR*.

Li, H.; Eigen, D.; Dodge, S.; Zeiler, M.; and Wang, X. 2019a. Finding task-relevant features for few-shot learning by category traversal. In *CVPR*.

Li, W.; Xu, J.; Huo, J.; Wang, L.; Gao, Y.; and Luo, J. 2019b. Distribution consistency based covariance metric networks for few-shot learning. In *AAAI*.

Liu, Y.; Lee, J.; Park, M.; Kim, S.; Yang, E.; Hwang, S. J.; and Yang, Y. 2019. Learning to propagate labels: Transductive propagation network for few-shot learning. In *ICLR*.

Luo, Y.; Huang, Z.; Zhang, Z.; Wang, Z.; Li, J.; and Yang, Y. 2019. Curiosity-driven reinforcement learning for diverse visual paragraph generation. In *MM*.

Mishra, N.; Rohaninejad, M.; Chen, X.; and Abbeel, P. 2018. A simple neural attentive meta-learner. In *ICLR*.

Nichol, A.; Achiam, J.; and Schulman, J. 2018. On first-order meta-learning algorithms. *CoRR* abs/1803.02999.

Oreshkin, B. N.; López, P. R.; and Lacoste, A. 2018. TADAM: task dependent adaptive metric for improved few-shot learning. In *NeurIPS*.

Qiao, S.; Liu, C.; Shen, W.; and Yuille, A. L. 2018. Few-shot image recognition by predicting parameters from activations. In *CVPR*.

Ravi, S., and Larochelle, H. 2017. Optimization as a model for few-shot learning. In *ICLR*.

Ren, M.; Triantafillou, E.; Ravi, S.; Snell, J.; Swersky, K.; Tenenbaum, J. B.; Larochelle, H.; and Zemel, R. S. 2018. Meta-learning for semi-supervised few-shot classification. In *ICLR*.

Rusu, A. A.; Rabinowitz, N. C.; Desjardins, G.; Soyer, H.; Kirkpatrick, J.; Kavukcuoglu, K.; Pascanu, R.; and Hadsell, R. 2016. Progressive neural networks. *CoRR* abs/1606.04671.

Rusu, A. A.; Rao, D.; Sygnowski, J.; Vinyals, O.; Pascanu, R.; Osindero, S.; and Hadsell, R. 2019. Meta-learning with latent embedding optimization. In *ICLR*.

Shin, H.; Lee, J. K.; Kim, J.; and Kim, J. 2017. Continual learning with deep generative replay. In *NeurIPS*.

Snell, J.; Swersky, K.; and Zemel, R. S. 2017. Prototypical networks for few-shot learning. In *NeurIPS*.

Soltoggio, A.; Stanley, K. O.; and Risi, S. 2018. Born to learn: The inspiration, progress, and future of evolved plastic artificial neural networks. *Neural Networks* 108:48–67.

Sung, F.; Yang, Y.; Zhang, L.; Xiang, T.; Torr, P. H. S.; and Hospedales, T. M. 2018. Learning to compare: Relation network for few-shot learning. In *CVPR*.

Vinyals, O.; Blundell, C.; Lillicrap, T.; Kavukcuoglu, K.; and Wierstra, D. 2016. Matching networks for one shot learning. In *NeurIPS*.

Wang, Y.; Girshick, R. B.; Hebert, M.; and Hariharan, B. 2018. Low-shot learning from imaginary data. In *CVPR*.

Yan, S.; Zhang, S.; and He, X. 2019. A dual attention network with semantic embedding for few-shot learning. In *AAAI*.

Yang, Y.; Wu, Y.; Zhan, D.; Liu, Z.; and Jiang, Y. 2019. Deep robust unsupervised multi-modal network. In *AAAI*.

Yoon, J.; Kim, T.; Dia, O.; Kim, S.; Bengio, Y.; and Ahn, S. 2018. Bayesian model-agnostic meta-learning. In *NeurIPS*.

Zhang, Y.; Pal, S.; Coates, M.; and Üstebay, D. 2019. Bayesian graph convolutional neural networks for semi-supervised classification. In *AAAI*.