Distributed Multi-Agent Lifelong Learning

Anonymous authors
Paper under double-blind review

Abstract

Lifelong learning (LL) machines are designed to operate safely in dynamic environments by continually updating their knowledge. Conventional LL paradigms often assume that new data come labeled and that each LL machine has to learn independently from its environment. However, human labeling is expensive and impractical in remote conditions where automation is most desired. We introduce the Peer Parallel Lifelong Learning (PEEPLL) framework for distributed Multi-Agent Lifelong Learning, where agents continually learn online by actively requesting assistance from other agents instead of relying on the expensive environment to teach them. Unlike classical distributed AI, where communication scales poorly, lifelong learners need to communicate only on information they have not yet learned. Additionally, agents reply only if they are highly confident: Our TRUE confidence score represents an application of a Variational Classifier to quantify confidence in prediction. TRUE outperforms traditional Entropy-based confidence scores, reducing communication overhead by 18.05% on CIFAR-100 and 5.8% on MiniImageNet. To improve system resilience to low-quality or adversarial responses, our agents selectively accept a subset of received responses using the REFINE algorithm, which results in a 51.99% increase in the percentage of correct accepted responses on CIFAR-100 and 25.79% on MiniImageNet. Like traditional LL agents, PEEPLL agents store a subset of previously acquired knowledge as memory to learn alongside new information to prevent forgetting. We propose a Dynamic Memory-Update mechanism for PEEPLL agents that improves QA's classification performance by 44.17% on CIFAR-100 and 26.8% on MiniImageNet compared to the baseline Memory-Update mechanism. Our findings demonstrate that a PEEPLL agent can outperform an LL agent even if the latter has environmental supervision available, thus significantly reducing the need for labeling. PEEPLL provides a framework to facilitate research in distributed multi-agent LL, marking a substantial step towards practical, scalable lifelong learning technologies at the edge. Code is available at https://anonymous.4open.science/r/peepll-tmlr.

Code Availability

The source code for reproducing our results is available at [anonymous link].

1 Introduction

Lifelong Learning (LL) is an emerging field of AI that aims to enable continual learning while preventing catastrophic forgetting of previous experiences (Aljundi et al., 2019b;a; Mai et al., 2020; Chaudhry et al., 2018b; 2019). It aims to allow AI systems to improve with experience and makes no assumptions about how data appears in the real environment. To be context-aware, LL must include online lifelong learning (OLL), where models learn online from each data point only once. This is in addition to an offline global consolidation when time permits. OLL is particularly relevant for edge conditions where agents must be highly autonomous and adaptive.

State-of-the-art LL paradigms often assume that all incoming new data are labeled through environmental supervision and that LL agents must learn individually from their environments. This assumption is limiting the applicability of AI. In realistic environments, agents may have varied skills that they learned through their

particular experiences. We propose a paradigm in which agents can learn from their peer agents in addition to learning from the environment. This improves the system's adaptability and reduces reliance on expensive environmental supervision. This study focuses on presenting and solving distributed *multi-agent* OLL, which has numerous applications from self-driving vehicles to drones and more. Unlike classical distributed AI, our agents continually learn, and hence, they communicate only on unprecedented experiences; this limits communication among agents. Reducing reliance on environmental supervision is important to make LL applications more seamless for realistic and edge environments.

Recent literature motivates the urgency to incorporate a multi-agent perspective into LL paradigms. The authors, who participated in DARPA's Shared Experience Lifelong Learning (ShELL) program, argue that "One vision of a future artificial intelligence (AI) is where many separate units can learn independently over a lifetime and share their knowledge with each other. ... **The result is a network of agents that can quickly respond to and learn new tasks, that collectively hold more knowledge than a single agent and that can extend current knowledge in more diverse ways than a single agent.** ... We propose that the convergence of such scientific and technological advances will lead to the emergence of new types of scalable, resilient and sustainable AI systems." (Andrea Soltoggio, 2024). They suggest, among possible applications, Multi-Agent Active Sensing, Space Exploration, Responsive and Personalized Medicine, and Distributed Cybersecurity Systems.

Multi-Agent Machine Learning (MAML) facilitates agent collaboration either by solving problems jointly during deployment (Foerster et al., 2016; Gupta et al., 2017; Hüttenrauch et al., 2017) or by distributing tasks among agents during training and integrating their efforts during testing (Raja et al., 2022; Kim et al., 2022; Brito et al., 2021). However, the integration of MAML into LL introduces new challenges and necessitates a reevaluation of traditional MAML and LL methods. A primary challenge in MAML settings is the communication overhead among agents scales poorly as the number of agents increases. Furthermore, a unique limitation arises when integrating MAML into LL: communication protocols in MAML are often based on past interactions, reputations, or a fixed, known task distribution among the agents (Das et al., 2018; Jiang & Lu, 2018; Hoshen, 2017; Das et al., 2017). This limiting assumption does not hold true for LL agents who continuously update their knowledge. Finally, agents must navigate learning from potentially incorrect peer responses. To address these issues, we introduce a framework to facilitate multi-agent coordination for LL agents: **Pee**r, **P**arallel **L**ifelong **L**earning (PEEPLL, pronounced 'People').

1.1 Peer, Parallel Lifelong Learning (PEEPLL)

PEEPLL's mechanism and framework are designed to navigate the dynamic, uncertain environments typical of multi-agent LL settings; it focuses on judicious peer communication without overwhelming the system with excessive communication. Multi-agent cooperation across domains of fields helps reduce reliance on annotated data (environmental supervision). Regardless of the domain, in any decentralized multi-agent LL system, agents must be self-aware—able to autonomously identify novel tasks, actively seek help, responsibly assist others, and carefully evaluate received help.

While this study focuses on supervised image classification, Lifelong Learning is also increasingly being applied in domains like language modeling (Gururangan et al., 2020; Qin et al., 2023), robotics (Lesort et al., 2019; Logacjov Aleksej, 2021), and medical AI (Lee & Lee, 2020; Vokinger et al., 2021), where agents develop sub-domain-expertise in their respective environments through continual learning. The principles underlying PEEPLL are applicable across various domains. In language modeling, for instance, PEEPLL can allow LLM agents to communicate with peers when handling uncertain prompts, allowing them to continuously learn to generate accurate and contextually relevant responses. The received peer responses can also be used for future fine-tuning. In high-stakes medical AI, PEEPLL can allow agents to seek assistance when uncertain and take only confident and well-validated actions. In robotics, PEEPLL can facilitate real-time collaboration, enabling robots to responsibly exchange sub-domain-specific expertise and adapt to new tasks (either online or offline, depending on the application). The **Framework** section below provides domain-agnostic guidelines for developing dedicated algorithms to the components of PEEPLL.

Our solution to PEEPLL demonstrates that a PEEPLL agent with no environmental supervision can perform better on novel tasks than an LL agent with complete environmental supervision. This advancement helps minimize disruptions to the user experience and progresses toward realizing seamless ShELL technologies.

Mechanism: When an agent, designated Query Agent (QA), receives a query that it cannot confidently answer, it seeks assistance from other peer agents in the network, designated Response Agents (RAs). The RAs respond if they are confident in their answers. The QA then accepts these responses, or a subset of them, for lifelong learning. For a complete understanding, please refer to Figure 1, which illustrates the PEEPLL mechanism.

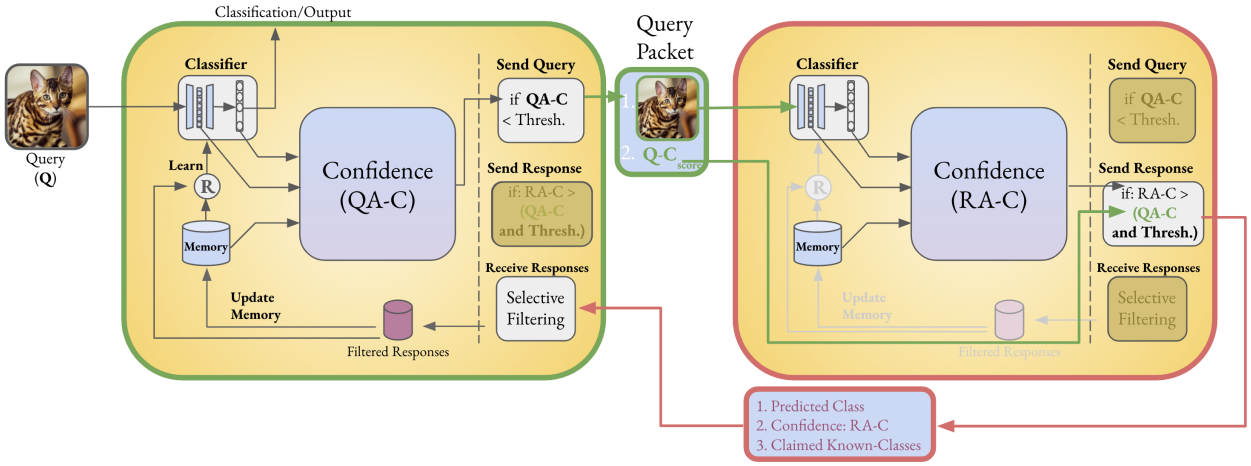


Figure 1: **PEEPLL's Query-Reponse Mechanism:** The agent receiving the original query is designated as the Query Agent (QA), while the agents it seeks assistance from are termed Response Agents (RAs). All agents are identical and can function interchangeably as either a QA or an RA, depending on the recipient of the query. Upon receiving a query, the QA assesses its confidence in providing an accurate response. If underconfident, the QA queries other agents in the network (RAs). RAs then evaluate their confidence in responding to the query. If confident, they transmit their responses back to the QA. Finally, the QA applies Selective Filtering to accept only a subset of the received responses. The QA then learns from these accepted responses in conjunction with replay samples from its memory (circle with R) while updating its memory with the accepted responses.

Framework: We identify three major components of the PEEPLL mechanism that are essential for effective multi-agent coordination in lifelong learning technologies - (1) Confidence-Evaluation, (2) Selective Filtering, (3) Lifelong Learning in PEEPLL. In the rest of this section, we detail the components and the initial benchmarks in those components.

1. **Confidence-Evaluation:** This component enables (a) the QA to autonomously recognize novel tasks and request assistance when needed and (b) the RAs to evaluate their ability to assist the QA reliably, transmitting only likely correct and relevant responses. The primary aim is to (1) restrict communication to essential queries and responses, and (2) promote stable learning at the QA.

   Given the continual evolution of LL agents' knowledge, Confidence Evaluation must operate beyond reliance on historical performance or preexisting knowledge bases, ensuring assessments are agent-agnostic and contextually relevant to each specific query. Moreover, the confidence score must retain its efficacy even after the agent has lifelong learned.

   Our proposed TRUE confidence score reduces communication overhead by 18.05% on CIFAR-100 and 5.8% on Mini-ImageNet compared to the widely accepted and effective 'Entropy' based methods (Yona et al., 2022; Chen et al., 2021) (see Section 4.1). TRUE serves as a general contribution to the ML community for uncertainty quantification.

2. **Selective Response Filter:** After receiving responses from all the confident RAs, the QA further accepts only the most reliable responses. This component is important to security in multi-agent

systems, assuring that an agent does not simply choose the first or the most confident response, which might have been adversarial.
Our most effective method increases the proportion of correct responses in the selected responses by 51.99% on CIFAR100 and 25.79% on MiniImageNet (Section 4.2).

3. **Lifelong Learning in PEEPLL:** Integrating MAML into LL requires reevaluating traditional LL concepts. Replay-based LL agents allocate memory to store samples from previous experiences and learn alongside new information to prevent forgetting. In PEEPLL, incorporating incorrect responses from RAs into the QA's learning process and memory introduces new challenges:

   (1) *Memory-Update:* Algorithms must handle incorrect responses from RAs when updating the QA's memory. These incorrect responses can fill the QA's memory budget, but we show that they may also have a beneficial regularization effect on learning (see Section 4.3).

   (2) *Memory-Retrieval:* Algorithms must dynamically retrieve appropriate memory samples by considering (a) relevant current queries, (b) the QA's confidence in those queries, and (c) the confidence of the responding RAs. Algorithms can also consider the regularization effect of noisy responses in selecting the most pertinent memory samples to improve performance.

   (3) *Adaptive-Learning:* Algorithms must modulate the importance of potentially incorrect responses based on their confidence. These algorithms may also consider the regularization effect of noisy responses to improve performance, balancing stability and regularization as a tradeoff.

   Our proposed *Memory-Update* method improves QA's local performance on all lifelong learned tasks by 44.17% on CIFAR-100 and 26.8% on MiniImageNet (see Section 4.3).

We present the conceptual (1) and technical contributions (2, 3) of this paper below:

1. We introduce PEEPLL, a framework for solving distributed Multi-Agent Lifelong Learning, addressing the unique challenges of integrating multi-agent coordination with lifelong learning.

2. We set initial benchmarks for the components of PEEPLL. Within our solution to PEEPLL, an agent with no environmental supervision achieves higher lifelong performance than an LL agent with complete environmental supervision while minimizing communication overhead.

3. We introduce an application of a Variational Classifier for superior uncertainty quantification compared to Entropy-based methods.

2 Related Work

This section discusses (1) Single-Agent Lifelong Learning, given the literature's gap in Multi-Agent Online Lifelong Learning, and (2) Multi-Agent Lifelong Learning, which focuses on non-online implementations. Finally, we include the rationale behind the choice of a fully distributed setup in our study.

Single-Agent Lifelong Learning: We modify the data introduction regime of Single-Agent Lifelong Learning for Multi-Agent Online Lifelong Learning, as outlined in Section 4. Thus, for fair benchmarking, we implement ER (Chaudhry et al., 2019) (ER in Table 2) on our end.
We examine Memory-Based Lifelong Learning strategies within the Class-Incremental setting, where data of unknown classes is presented sequentially in distinct, non-overlapping 'tasks' to the lifelong learning agent. Architecture-based methods are incompatible (Mai et al., 2021), and Regularization-based methods struggle with this setting (Lesort et al., 2021). Elastic Weight Consolidation (EWC) (Kirkpatrick et al., 2016) is a Regularization-based strategy that prevents forgetting by penalizing changes to parameters important for previously learned tasks, using a Fisher information matrix to measure importance. Memory-based lifelong learning methods mitigate forgetting by storing and replaying (relearning) data from previously seen classes. This typically involves *Memory-Update* selecting what to store and *Memory-Retrieval* selecting what to replay. Mai et al. (2020), use Shapley Values for informed retrieval; Aljundi et al. (2019a) retrieve samples that maximize learning interference; Aljundi et al. (2019b) focuses on gradient diversity while storing in the memory; Chaudhry et al. (2018b) constraints the model's loss on its memory samples to mitigate forgetting;

Prabhu et al. (2020) (GDumb) adopts a simple, balanced sampling approach, retraining the network from scratch at test time. Despite its simplicity, GDumb's effectiveness has prompted a reevaluation of the progress in lifelong learning. Similarly, Chaudhry et al. (2019) (ER) uses random sampling for memory management (update and retrieval), and often surpasses other complex methods. ER is considered state-of-the-art van de Ven et al. (2022). *Naive* strategies fine-tune the model on incoming data without replaying any previously learned data. *IID* strategies sample independently and identically from all of the training data.

Multi-Agent Lifelong Learning: In Rostami et al. (2017), each agent knows specific distributions of the dataset. They assume that this distribution is fixed and known. Their lifelong learning agent is designed to be aware of all classes and learns only from out-of-distribution data, known as the Domain-Incremental Scenario. Our research differs in two key aspects: Firstly, we don't predefine or know how tasks are distributed among agents, as this is not a fair assumption for lifelong learning agents. Secondly, our agents are strictly not aware of certain classes, focusing instead on Class-Incremental Lifelong Learning. Babakniya et al. (2023) tackle catastrophic forgetting in Federated Learning (FL) using a global generative model for data-free knowledge distillation, focusing on privacy but increasing communication overhead and presenting a risk of single point of failure due to server centrality (centralized topology of communication). To mitigate these issues, we focus on a Peer-to-Peer topology.

We propose using GDumb as a Multi-Agent LL baseline. Assuming memory is shared among all agents, the query agent can retrain using relevant samples when encountering new classes. Although this assumes 100% accurate communication and isn't a fully distributed setup, it provides a strong baseline to beat.

Multi-Agent Architecture : After thoroughly evaluating communication topologies (Verbraeken et al., 2019), we selected a fully distributed peer-to-peer architecture for the PEEPLL system. While centralized systems offer quick synchronization, they struggle with scalability and single-node failure risks; decentralized systems improve scalability but still have communication overhead and node failure risks. Peer-to-peer networks avoid these issues but face challenges in synchronizing agent knowledge. PEEPLL addresses this through Confidence-Evaluation methods for knowledge alignment and Lifelong Learning strategies for continuous knowledge synchronization.

3 PEEPLL: Our Approach

This section presents our solutions to the components of the PEEPLL mechanism discussed in Section 1.1 - (1) Confidence-Evaluation Strategy, (2) Selective Response Filtering, (3) Lifelong Learning in PEEPLL.

3.1 Confidence-Evaluation Strategy

We introduce a modified compute-efficient version of Variational Autoencoders for image classification without input reconstruction to quantify uncertainty in model prediction. This architecture is similar to the recently proposed Variational Classifier (VC) by Dhuliawala et al. (2024). Since our study focuses on lifelong learning, where subsets of experiences are preserved as memory by the model, we focus on memory-based models with our uncertainty quantification mechanism.

We first detail our Variational Classifier's *Architecture and Training*. Then, detail how the model's uncertainty in prediction (inversely, its confidence in prediction) is discerned.

Architecture and Training: VC's architecture comprises two primary components:
1. **Encoder:** Takes the image as input ('x') and outputs $z_{\text{mean}}(x)$ and $z_{\text{log_var}}(x)$. $z_{\text{mean}}(x)$ and $z_{\text{log_var}}(x)$ are used to define the probability distribution from which a latent code z for the image is sampled.

The Encoder is optimized using a Kullback–Leibler divergence loss to facilitate a continuous latent space representation. The Kullback–Leibler divergence loss can be mathematically expressed as:

$$\text{KL}(q_\phi(z|x)||p(z)) = -\frac{1}{2}\sum_{k=1}^{K}\left(1 + z^k_{\text{log_var}}(x) - (z^k_{\text{mean}}(x))^2 - e^{z^k_{\text{log_var}}(x)}\right)$$

where $q_\phi(\mathbf{z}|\mathbf{x})$ is the approximate posterior distribution of z given x, parameterized by ϕ (encoder), $\mathbf{p(z)}$ is the prior distribution over z (Gaussian), $\mathbf{K}$ is the dimension of z_{mean} and z_{log_var}, $\mathbf{z}^k_{\text{mean}}$ is k-th entry of the z_{mean} vector, and $\mathbf{z}^k_{\text{log_var}}$ is k-th entry of the z_{log_var} vector.

2. **Decoder:** Takes in the latent code z as input and outputs a classification layer. This component is trained using a Cross-Entropy loss to predict the class of the input image. Note here that we forgo input reconstruction to conserve computational resources, and hence, no reconstruction loss is used for training.

Confidence-Evaluation mechanism: Given a query q, the encoder evaluates z^q_{mean} and $z^q_{\text{log_var}}$, and a latent code z is sampled. The decoder then takes in z and yields a prediction p_q for q. If associated memory samples exists for the label p_q, the corresponding $z^{\text{memory}}_{\text{mean}}$ and $z^{\text{memory}}_{\text{log_var}}$ for each memory samples is retrieved. Using these, we evaluate (1) *Semantic Disparity*, (2) *Dispersion Disparity*, and (3) Entropy, which are then averaged to produce the confidence score.

(1) *Semantic Disparity* captures the disparity in the property of the mean - query's position in the latent space versus the mean position of the model's memory of the predicted label for the query. We define *Semantic Disparity* using Euclidean distance:

$$d_{\text{semantic}}(q, \text{memory}) = ||z^q_{\text{mean}} - \overline{z^{\text{memory}}_{\text{mean}}}||$$

where $\overline{z^{\text{memory}}_{\text{mean}}}$ represents $\frac{1}{N}\sum_i z^{\text{memory}_i}_{\text{memory}}$ where N is the total number of retrieved memory samples. z_{mean} represents the mean of the latent variable's probability distribution for a given input; it is where the model predicts the center in the latent space is for the input.

(2) *Dispersion Disparity* captures the disparity in the property of variance - query's variance versus the variance of the model's memory of the predicted label for the query. We define *Dispersion Disparity* using Manhattan distance:

$$d_{\text{dispersion}}(q, \text{memory}) = ||\exp(z^q_{\text{log_var}}) - \overline{\exp(z^{\text{memory}}_{\text{log_var}})}||_1$$

where $\overline{z^{\text{memory}}_{\text{log_var}}}$ represents $\frac{1}{N}\sum_i z^{\text{memory}_i}_{\text{log_var}}$ where N is the number of retrieved memory samples. $z_{\text{log_var}}$ is the logarithm of the variance of the latent variable's distribution. Intuitively, the variance represents the spread or dispersion of the latent representations around z_{mean}. A larger variance (higher $\exp(z_{\text{log_var}})$) indicates the model's higher uncertainty in pinpointing the class for similar images.

The choice of Euclidean distance for the *Semantic Distance* is to measure the spatial distance between the query's *z_mean* and the memory samples' *z_mean* in the latent space, as *z_mean* represents a meaningful 'physical' location in this space. In contrast, *z_var* does not correspond to a physical spatial location but rather indicates the variance across dimensions. Therefore, we use the Manhattan Distance to evaluate the sum of disparities between the query's and memory samples' variance across all dimensions. After experimenting with various configurations, we find this approach to be the most effective.

Semantic & Dispersion Disparity are transformed into confidence scores via a negative exponential function, $\exp(-\alpha d_{\text{semantic}})$ (denoted C_{semantic}), $\exp(-\beta d_{\text{dispersion}})$ (denoted $C_{\text{dispersion}}$). This ensures the scores are bounded between [0, 1] and inversely proportional to the assessed distances, where α and β are scaling factors. The transformed distances are subsequently normalized (denoted $\hat{C}_{\text{semantics}}$, and $\hat{C}_{\text{dispersion}}$ respectively).

(3) Entropy (Shannon, 1948) is defined as $H(\mathbf{p}) = -\sum p_i \cdot \log_2(p_i + \epsilon)$, where $\mathbf{p}$ is the probability distribution of the agent's predictions and ϵ (a small number) prevents undefined logarithmic operations.

Finally, the mean of the three is calculated to ensure that the resultant Triplet Uncertainty Evaluation (TRUE) confidence score falls within the range [0, 1]:

$$\text{TRUE} = \frac{\hat{C}_{\text{semantics}} + \hat{C}_{\text{dispersion}} + \text{Entropy}}{3}$$

Refer to Figure 12 and Algorithm 2 in the Appendix for a visualization and a pseudocode to evaluate TRUE.

The QA utilizes TRUE to decide whether to initiate communication with the RAs, and the RAs utilize TRUE to determine whether their responses should be transmitted back to the QA. We measure TRUE's

effectiveness by the 'Sharing Accuracy' metric: the proportion of responses sent by RAs to the QA that were correct. We demonstrate that TRUE offers a higher Sharing Accuracy than Entropy-based measures. This implies that it is possible to generate meaningful latent representations despite forgoing reconstruction.

3.2 Selective Response Filtering

We propose three approaches to filter out pertinent responses at the QA once the RAs have responded.

1. Majority Voting: Response agents are grouped by the label they are predicting. The label with the highest number of agent endorsements is selected.
2. Most-Confident-Group (MCG): Response agents are grouped by their predicted labels. The group with the highest average confidence is selected.
3. Intelligent Comparative Filtering (ICF): Responses from RAs predicting label l are rejected if any other RA that has memory samples of label l does not predict class l.

The effectiveness of a filter lies in accurately identifying and accepting the correct responses (increasing the proportion of correct responses in the accepted subset).

3.3 Lifelong Learning & *Memory-Update*

As the QA receives responses from RAs, it stores these responses and their corresponding confidence levels in a memory organized by class. We allocate a budget of 50 samples per class. To dynamize our *Memory-Update* mechanism, once the memory reaches 50 samples for a class, it dynamically replaces the least confident samples with new, higher-confidence responses.

4 Results & Analysis

This section outlines our modified traditional LL experimental setup used to fairly evaluate solutions to PEEPLL's components and discusses the results in Sections 4.1-4.3

Data Distribution and Agent Roles: The data in the training set is divided into two parts: (1.1) pretraining data for all agents and (1.2) lifelong learning data for the QA. The pretraining data (1.1) is further divided into (1.1.Q-Pre) and (1.1.R-Pre). (1.1.Q-Pre) contains data from 5 classes assigned for the QA's pretraining. (1.1.R-Pre) contains data from classes assigned for the pretraining of the RAs. These classes are distributed among RAs, ensuring that 2-3 RAs are assigned to each class to foster diverse responses. Each RA is trained on data from its assigned classes in (1.1.R-Pre). From the lifelong learning data (1.2), the data for classes not in (1.1.Q-Pre) are extracted and referred to as (1.2.LL). (1.2.LL) is segmented into distinct 'tasks,' each comprising a unique set of classes to facilitate the Class-Incremental Scenario as defined in Aljundi et al. (2019b;a); Mai et al. (2020). Task_t has classes, $\mathcal{C}_t$, absent in previously seen tasks, $\text{Task}_{1-(t-1)}$, such that $\mathcal{C}_t \subset \mathcal{C} \setminus \mathcal{C}_{1-(t-1)}$. The QA is incrementally introduced to a Task_t, which it learns in communication with the other agents (RAs) in the network. Note here that since we differentiated the data into the pretraining data (1.1) and the lifelong learning data (1.2), we can fairly evaluate the RA's responses and the confidence-evaluation strategy that discerns the RA's confidence while responding. In evaluation, we follow the single-head evaluation setup Chaudhry et al. (2018a) where the QA does not know the task it is introduced, so it has to choose between all possible classes for classification.

Our experimental setup considers 20 agents - 1 QA and 19 RAs - that are pre-trained on subsets of 5-10 classes each, achieving 75-85% accuracy on their specific tasks. This simulates scenarios with imperfect, non-teacher-like agents. We evaluate our solutions on vision datasets CIFAR-100 and MiniImageNet.

4.1 Confidence-Evaluation

Benchmarks: First, we establish benchmarks for comparing confidence-evaluation strategies.
(1) *Baseline:* Our baseline strategy for assessing agent confidence employs softmax probabilities defined as

$\sigma(\mathbf{p})_i = \frac{e^{p_i}}{\sum_{j=1}^{K} e^{p_j}}$. This transforms the output of the model - vector $\mathbf{p}$ - into a probability distribution. This distribution provides insight into the agent's confidence level regarding its response. We include results for this in our Appendix (Figure 13), given its documented inferior performance relative to entropy-based measures Yona et al. (2022), which constitute the primary focus of our comparative analysis and discussion. (2) *Entropy-Based Measure:* To enhance our confidence assessment, we use Entropy, introduced by Shannon (1948) and a widely effective and accepted metric (Yona et al., 2022; Chen et al., 2021) for uncertainty quantification. It quantifies 'chaos' in a probability distribution. Since Entropy ≥ 0, we transform this entropy into a confidence score using a negative exponential mapping, $e^{-H(\mathbf{p})}$. This transformation maps higher entropy (chaos) to lower confidence and vice versa. Note that this transformation clips the confidence score from 0 to 1.

(a) TRUE vs Entropy: This plot displays the percentage of correct responses in the responses sent by RAs to the QA using TRUE and Entropy as confidence scores (high is better). TRUE outperforms Entropy consistently.

(b) A PEEPLL system using TRUE needs 5-20% fewer total RA responses (vs. Entropy) for the QA to receive the same number of correct responses. This contributes to a reduction in system communication overhead.

Figure 2: These plots correspond to evaluation on CIFAR-100. See Figure 10 for MiniImageNet in Appendix.

TRUE Results: The RAs compare their confidence against a threshold, and if their confidence is greater than the threshold, they send their responses back to the QA. We measure *Sharing Accuracy*: the percentage of responses sent by RAs to the QAs that match the ground truth. For standardized analysis, in this section, we report numbers that refer to the thresholds for a 1:1 Sharing Ratio - the number of responses the QA receives is close to the number of queries it sends. See Figure 2 for a comparison of Entropy and TRUE across thresholds. TRUE shows a 19.29% improvement in Sharing Accuracy (from 46.3% to 55.2%) on CIFAR-100 and a 7.03% improvement (from 68.4% to 73.2%) on MiniImageNet over using Entropy. An ablation experiment demonstrates that incorporating Semantic Distance (d_{semantic}) yields an 11.7% improvement (47% to 52%) and Dispersion Distance ($d_{\text{dispersion}}$) yields a 16.75% improvement (46.5% to 54.3%) in Sharing Accuracy on CIFAR-100. The efficacy of TRUE also demonstrates that it is possible to derive a meaningful latent space in Variational Autoencoders without reconstruction.

Since the TRUE confidence score aligns more closely to response correctness than Entropy, the RAs are better equipped to judiciously transmit responses. As a result, it helps restrict communication. To get 'x' correct responses at the QA, the RAs need to send 5-20% fewer responses on CIFAR100 (Figure 2b) and 5-15% on MiniImageNet (Figure 10b in Appendix) while using TRUE vs Entropy. This is a critical analysis to assess the reduction in communication overhead.

We find that knowledge sharing among agents scales inversely with the number of agents in the environment (see Figure 7, Appendix). This trend is expected as the likelihood of randomly choosing the correct agent to listen to decreases with an increasing number of agents, making the task progressively harder. However, our results indicate that TRUE consistently outperforms Entropy across varying numbers of agents.

The computation complexity for an agent in PEEPLL, in terms of the scalable symbols - Q total queries, N(., out) the number of RAs a QA contacts for a query, and K total classes, is $\mathbf{O(Q \times N(., out) \times K)}$. Refer to Section E of the Appendix for a complete derivation of the complexity.

While TRUE has demonstrated clear advantages over entropy-based methods for uncertainty quantification, we note that its reliance on a variational classifier limits its direct application to pre-trained models and purely discriminative architectures. This presents interesting opportunities to advance and expand TRUE's applicability to such models by modifying the architecture and fine-tuning.

4.2 Selective Response Filter:

The QA accepts a subset of responses from all received RA responses. We measure the percentage of correct responses in the accepted subset of responses. We provide a structured display of results in Table 1.

Table 1: Efficacy of Filters applied at the QA

Method	Sharing Accuracy (%)	% Improvement (↑)	Sharing Accuracy (%)	% Improvement (↑)
	CIFAR-100		MiniImageNet	
TRUE	55.2	-	73.17	-
TRUE + Majority	66.2	19.92	83.56	14.02
TRUE + MCG	65.8	19.13	82.04	12.14
TRUE + ICF	82.8	50	91.22	24.67
TRUE + Majority + ICF	83.3	50.9	91.91	25.6
TRUE + REFINE	**83.9**	**51.99**	**92.04**	**25.79**

The QA employs our proposed filters to select a subset of responses received from all RAs. This table displays the percentage of accepted responses that match the ground truth (Sharing Accuracy). REFINE (ICF with MCG) is the most effective filter. These values correspond to a 1:1 Sharing Ratio (thresholds where the number of accepted responses by the QA is close to the number of queries sent out by the QA). For an equivalent analysis across thresholds, see Figure 3.

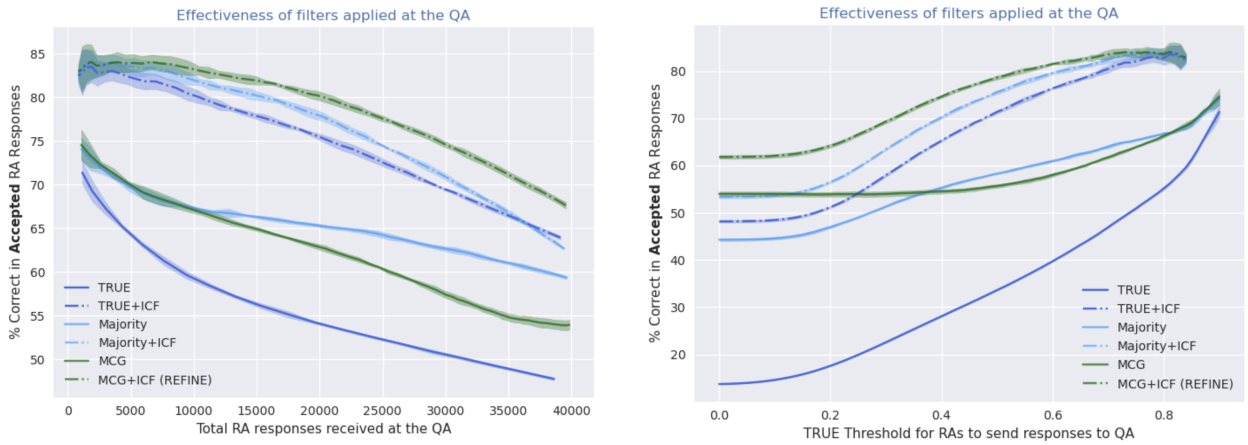
As seen in Figure 3b (and 11b in Appendix), the Most-Confident-Group (MCG) approach exhibits enhanced effectiveness at lower thresholds (more RAs respond), suggesting that in scenarios where response criteria are less strict, prioritizing collective agent confidence yields better results than majority voting. The rationale is that when agents with low confidence are allowed to respond, they cannot be taken at face value, and a further focus on their confidence is required. As the threshold increases, the Majority-Voting method parallels and occasionally surpasses the MCG approach. This trend indicates that with higher confidence thresholds, agents become more reliable, allowing responses with the most agent endorsements to be selected with greater assurance of accuracy. Implementing Intelligent Comparative Filtering (ICF) with the MCG filter retains a consistent edge over Majority Voting with ICF.

ICF consistently outperforms non-ICF methods. ICF with MCG constitutes our most effective filter, which we term "**REFINE**" filter; see Table 1 and Figure 3.

4.3 Lifelong Learning

For the sake of initial comparison, we designate one agent as the QA, while the remaining agents act as RAs. After the QA learns all Tasks_{1-T} incrementally in communication with its imperfect peers, we measure its local performance on the complete test set. For standardized testing, we evaluate the QA's performance with the communication protocol's thresholds set to achieve a 1:1 Sharing Ratio, where the number of peer responses the QA accepts and learns from is close to the number of queries it seeks assistance on.

We compare our solution to PEEPLL against a Single-Agent that learns with complete supervision - the QA learns new tasks using ground truth labels, same as the Experience Replay (ER) (Chaudhry et al., 2019)



(a) (b)

Figure 3: The QA applies our proposed filters to select a subset of responses received from RAs. These plots display the percentage of correct responses in the accepted subset (filter efficacy) across different confidence thresholds that the RAs must meet before sending their responses to the QA. Both plots are equivalent: the left plot's x-axis (total responses the QA receives) is controlled by the confidence threshold parameter shown on the right plot's x-axis. In the left figure, we cut off at a threshold of a 1:2 ratio of Queries asked to Responses received for clarity. Our REFINE filter consistently outperforms all other filters. This plot corresponds to the evaluation on CIFAR-100 (for MiniImageNet, see Appendix Figure 11).

strategy. This refers to 'ER' as seen in Table 2. ER has consistently been shown to outperform other LL strategies and is state-of-the-art (van de Ven et al., 2022). Matching ER necessitates advanced strategies in Confidence-Evaluation, Selective Response Filter, and Lifelong Learning in PEEPLL, particularly those that judiciously utilize potentially incorrect responses.

As shown in Table 2, despite learning from noisy peer responses, our solutions to PEEPLL outperform ER, the Supervised Single-Agent, which relies on ground-truth responses (complete environmental supervision), by 2.1% on CIFAR-100 and 2.1% on MiniImageNet.

We find that the noisy responses that the QA learns from, in fact, generate a beneficial regularization effect in its learning process. To analyze this effect and the QA's ability to effectively learn from potentially incorrect peer responses, we monitor its performance on future yet-to-be-introduced tasks throughout its lifelong learning journey (see Figure 5a and 8a in Appendix). Observe that a PEEPLL system using a communication protocol with lower Sharing Accuracy shows better QA performance on future tasks. This suggests that responses deemed 'incorrect' may still hold relevance for queries of another class with similar characteristics. This is also supported by Figure 6a, where the QA's initial confidence at each subsequent task progressively rises throughout its learning journey. This is because the QA has previously, albeit briefly, learned about these classes from 'incorrect' responses. We hypothesize that briefly learning about future classes while learning similar classes beforehand produces a regularization effect that aids in more effective learning of those future classes when they become relevant. This is supported by the trends in Figure 5b (and 8b, Appendix), where initially, the Single-Agent (Supervised ER) and the QA learning from peer responses perform comparably, but the QA outperforms in later stages. This phenomenon is akin to, and further supported by, the less challenging Domain-Incremental Learning (Domain-IL) scenario in lifelong learning, where an agent incrementally learns the same classes from different data distributions. Research has shown that LL agents find learning easier under Domain-IL than Class-IL (van de Ven et al., 2022), explaining the QA's effective learning despite potentially incorrect responses coming from peers. However, note that the QA's improved performance on future tasks comes at the expense of current task performance, highlighting the need for advanced communication protocols. This tradeoff holds implications for LL algorithms that will be developed for PEEPLL agents.

Note that the communication protocol influences the subset of responses chosen for learning and the percentage of correct responses within that subset, affecting the regularization effect and the QA's learning for current and future tasks. While we demonstrate that a regularization effect exists, future work will examine in detail the impact of our proposed communication protocols on the regularization effect and the QA's performance on current and future tasks.

Another important observation from Figures 5a (and 8a in Appendix) is that the QA performs better on future tasks within the PEEPLL system that uses TRUE as the communication protocol compared to Entropy, despite TRUE having higher Sharing Accuracy. This indicates that the TRUE score not only aligns more closely with response correctness but also more effectively captures query similarity. In other words, a high TRUE score for an incorrect response suggests that the predicted class is likely similar to the query's original class. Conversely, a high Entropy score for an incorrect response indicates that the predicted class is less likely, compared to TRUE, to be similar to the query's original class.

Table 2: The QA's local performance on the complete test set after lifelong learning of all tasks.

Communication Protocol	**Accuracy (%)**	**Quality (%)**	**Memory Samples**	**Accuracy (%)**	**Quality (%)**	**Memory Samples**
	CIFAR-100			**MiniImageNet**		
Offline Single-Agent (10 epochs)						
IID	58.2 ± 1.1	100	-	52.4 ± 1.2	100	-
Naive (fine-tune)	4.9 ± 0.2	100	-	4.6 ± 0.2	100	-
EWC	3.9 ± 0.3	100	-	3.1 ± 0.2	100	-
GDumb	**39.5 ± 0.8**	**100**	**5k**	**47 ± 1.3**	**100**	**5k**
Offline PEEPLL (10 epochs)						
PEEPLL+REFINE - **Naive**	4.2 ± 0.3	83.9	-	4.1 ± 0.3	92.04	-
Online Single-Agent						
IID	36.7 ± 0.9	100	-	34.5 ± 0.8	100	-
Naive	2.4 ± 0.1	100	-	2.2 ± 0.2	100	-
GDumb	3.6 ± 0.5	100	5k	9.2 ± 0.7	100	5k
ER	29.2 ± 0.5	100	5k	24.8 ± 0.47	100	5k
Online PEEPLL						
PEEPLL+REFINE - **Naive**	2.1 ± 0.1	83.9	-	1.9 ± 0.1	92.04	-
Online PEEPLL w. Confidence						
Entropy	26.5 ± 0.4	46.3	5k	22.35 ± 0.3	68.4	5k
TRUE	29.7 ± 0.15	55.2	4.5k	25.6 ± 0.58	73.17	4.3k
Online PEEPLL w. Confidence + Filter						
TRUE + Majority	30.8 ± 0.57	66.2	4.7k	**26.9 ± 0.76**	**83.56**	**4.6k**
TRUE + MCG	28.9 ± 0.39	65.8	4.9k	26.7 ± 0.76	82.05	5k
TRUE + ICF	31.1 ± 0.31	82.8	4.5k	26.9 ± 0.92	91.22	4.3k
TRUE + Majority + ICF	30.6 ± 0.27	83.3	4.5k	25.6 ± 0.49	91.91	4.7k
TRUE + REFINE	**31.3 ± 0.72**	**83.9**	**4.5k**	26.1 ± 1.10	92.04	4.7k

This table presents the QA's local performance on the complete test set following lifelong learning of all tasks - Accuracy (%). Other metrics shown include Quality (%), and Memory Samples. *Quality (%)* refers to the percentage of correct responses the QA learns from, determined by the communication protocol's Sharing Accuracy. Single-Agents learn from the environment with 100% Quality. PEEPLL agents may not fill their memory buffers if they do not receive enough RA responses for certain classes, while Naive strategies do not utilize Replay/Memory.Note that although GDumb is not specifically designed for Lifelong Learning, it is a strong baseline for comparison. Our results demonstrate that our PEEPLL agent outperforms ER (the state-of-the-art for Online Single-Agent LL) despite learning from noisy peer responses.

Figure 4 illustrates how the PEEPLL mechanism reduces system communication overhead over time. As the QA learns a task, its confidence in answering the task's queries increases. And since the QA only requests assistance when it is underconfident in answering a query, the increase in confidence leads to fewer requests for assistance. In Figure 4, the change in QA's confidence in each task is shown in green, while the change in

the likelihood of initiating a communication call is shown in red. We find the correlation between the change in QA's confidence and the change in communication likelihood is strong, 0.81, with a p-value of 2.98e-05, meaning it is statistically significant. Figure 6 in the supplementary material shows the unprocessed data of growing confidence and a decrease in communication. This reduction in communication overhead is crucial for multi-agent systems.

Note that even after the QA completes its subsequent lifelong learning tasks, the TRUE score continues to exhibit expected trends with each new task—initially low confidence that increases over time (see Figures 4 and 6). This demonstrates TRUE's context-aware adaptability to the QA's continual learning capability, which is crucial for lifelong learning applications.

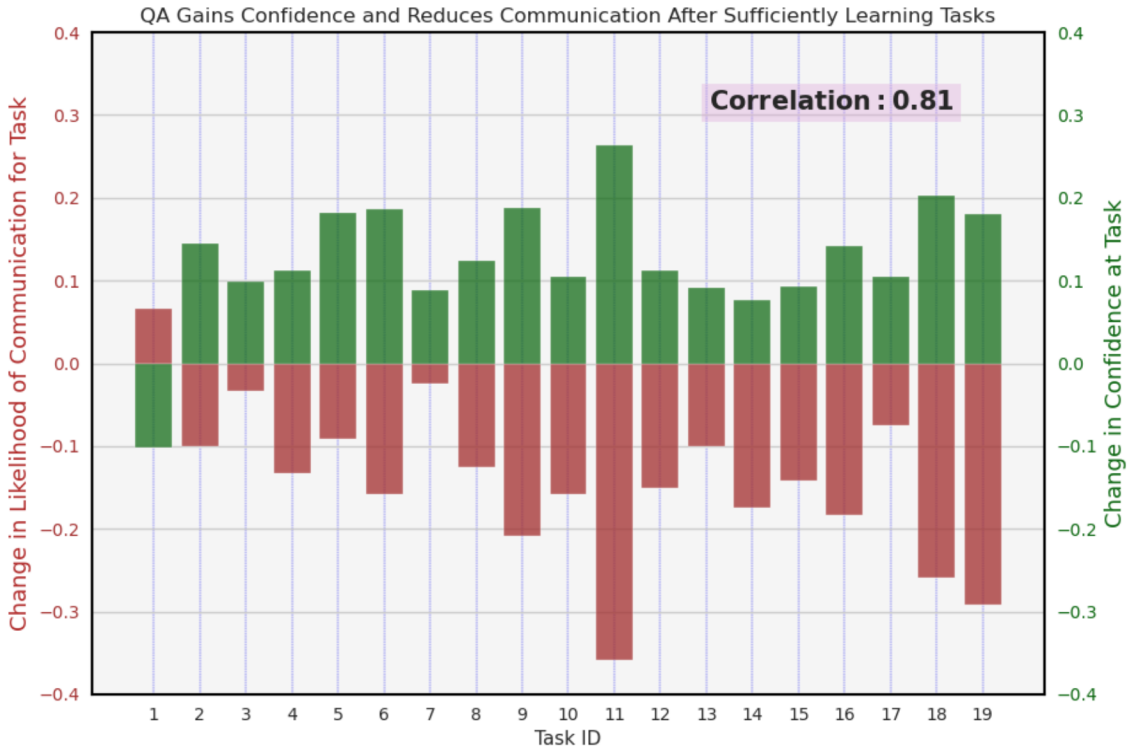
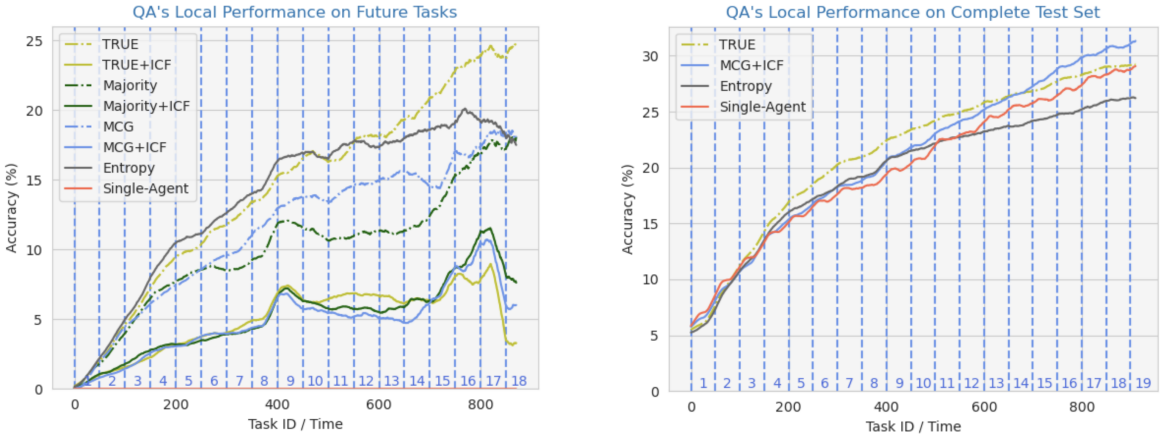


Figure 4: As the QA learns each task incrementally (x-axis), it gains confidence in answering queries related to that task (green). Consequently, it reduces communication calls (red) to the RAs for assistance. The green line indicates the change in the QA's confidence in answering queries from the start to the end of each task. The red line shows the change in the number of times the QA initiates communication with RAs per query from the start to the end of each task. The correlation between the change in QA's TRUE confidence and the reduction in communication calls is 0.81, with a p-value of 2.98e-05. This plot corresponds to a PEEPLL agent learning MiniImageNet with TRUE: Table 2, Row 12, and performs 25.6% on the test set.

The dynamism in our proposed Memory-Update mechanism improves QA's performance by 44.17% on CIFAR-100's test set (21.71% to 31.3% on CIFAR-100 with PEEPLL using TRUE + REFINE) and 26.8% on MiniImageNet's test set (21.21% to 26.9% on CIFAR-100 with PEEPLL using TRUE + Majority) compared to a static memory-update system that does not replace old, less-confident responses. This mechanism mitigates the adverse effects of previously received incorrect responses. More importantly, this mechanism is crucial for handling failures of critical agents. If a QA needs assistance from an agent who is hard to reach (e.g., offline or distant) but skilled in a certain task, it must be able to receive new responses once that agent becomes available and replace previous low-quality responses in memory. This mechanism stabilizes learning and enhances system reliability.



(a) QA's performance on yet-to-be-introduced tasks.

(b) QA's performance on all tasks.

Figure 5: Each vertical blue line marks the introduction of a new task to learn for the QA. In (a), observe that QA learning with communication protocol with lower sharing accuracy exhibits higher performance on untrained tasks. In (b), observe that QA learning from peer responses performs comparably as the single-agent initially, but outperforms in the later stages. This plot corresponds to evaluation on CIFAR-100. See Figure 8 in Appendix for MiniImageNet.

5 Conclusion

This paper introduces Peer Parallel Lifelong Learning (PEEPLL), the first distributed multi-agent lifelong learning framework. PEEPLL brings forth essential directions for realizing distributed lifelong learning, such as autonomously identifying novel tasks, navigating potentially incorrect responses from peers, and managing communication overhead. This paper lays the foundation for further research by setting initial benchmarks: TRUE for Confidence-Evaluation, REFINE for Selective Filtering, and Dynamic Memory-Update for Lifelong Learning in PEEPLL. Our results demonstrate that PEEPLL agents can outperform traditional LL agents with complete supervision, even when learning from potentially incorrect peer responses. By reducing reliance on environmental supervision, PEEPLL marks a step toward realizing seamless lifelong learning technologies at the edge. Moreover, since PEEPLL's self-aware agents respond only when confident, seek peer assistance, and learn only when underconfident, this reduces the risks of encountering unprecedented events as well as forgetting previously learned knowledge. Thus, our proposed PEEPLL mechanism significantly improves the safety and adaptability of LL systems in dynamic learning conditions. Despite its controlled scope, this study establishes a foundation for research in distributed multi-agent lifelong learning. Future research will delve into mixed dependence on environmental supervision and peer assistance and explore variations in agent quantity, expertise, and memory budgets. We will also explore communication strategies with centralized support to develop more advanced communication protocols. These investigations will further refine PEEPLL's capabilities, paving the way for more seamless, resilient, and adaptive Lifelong Learning systems.

References

Alexander A. Alemi, Ian Fischer, Joshua V. Dillon, and Kevin Murphy. Deep variational information bottleneck. *CoRR*, abs/1612.00410, 2016. URL http://arxiv.org/abs/1612.00410.

Rahaf Aljundi, Lucas Caccia, Eugene Belilovsky, Massimo Caccia, Min Lin, Laurent Charlin, and Tinne Tuytelaars. Online continual learning with maximally interfered retrieval. *CoRR*, abs/1908.04742, 2019a. URL http://arxiv.org/abs/1908.04742.

Rahaf Aljundi, Min Lin, Baptiste Goujaud, and Yoshua Bengio. Online continual learning with no task boundaries. *CoRR*, abs/1903.08671, 2019b. URL http://arxiv.org/abs/1903.08671.

Vladimir Braverman et al. Andrea Soltoggio, Eseoghene Ben-Iwhiwhu. A collective AI via lifelong learning and sharing at the edge. 3 2024. URL https://repository.lboro.ac.uk/articles/journal_contribution/A_collective_AI_via_lifelong_learning_and_sharing_at_the_edge/25470358.

Sara Babakniya, Zalan Fabian, and et al. Chaoyang He. A data-free approach to mitigate catastrophic forgetting in federated class incremental learning for vision tasks, 2023.

Bruno Brito, Michael Everett, Jonathan P. How, and Javier Alonso-Mora. Where to go next: Learning a subgoal recommendation policy for navigation in dynamic environments. *IEEE Robotics and Automation Letters*, 6:4616–4623, 2021.

Arslan Chaudhry, Puneet Kumar Dokania, Thalaiyasingam Ajanthan, and Philip H. S. Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. *ArXiv*, abs/1801.10112, 2018a.

Arslan Chaudhry, Marc'Aurelio Ranzato, and et al. Marcus Rohrbach. Efficient lifelong learning with A-GEM. *CoRR*, abs/1812.00420, 2018b. URL http://arxiv.org/abs/1812.00420.

Arslan Chaudhry, Marcus Rohrbach, and et al. Mohamed Elhoseiny. Continual learning with tiny episodic memories. *CoRR*, abs/1902.10486, 2019. URL http://arxiv.org/abs/1902.10486.

Zhi Chen, Zi Huang, Jingjing Li, and Zheng Zhang. Entropy-based uncertainty calibration for generalized zero-shot learning, 2021.

Abhishek Das, Satwik Kottur, José M. F. Moura, Stefan Lee, and Dhruv Batra. Learning cooperative visual dialog agents with deep reinforcement learning. *CoRR*, abs/1703.06585, 2017. URL http://arxiv.org/abs/1703.06585.

Abhishek Das, Théophile Gervet, and et al. Joshua Romoff. Tarmac: Targeted multi-agent communication. *CoRR*, abs/1810.11187, 2018. URL http://arxiv.org/abs/1810.11187.

Shehzaad Dhuliawala, Mrinmaya Sachan, and Carl Allen. Variational classification, 2024. URL https://arxiv.org/abs/2305.10406.

Jakob N. Foerster, Yannis M. Assael, Nando de Freitas, and Shimon Whiteson. Learning to communicate with deep multi-agent reinforcement learning. *CoRR*, abs/1605.06676, 2016. URL http://arxiv.org/abs/1605.06676.

Jayesh K. Gupta, Maxim Egorov, and Mykel J. Kochenderfer. Cooperative multi-agent control using deep reinforcement learning. In *AAMAS Workshops*, 2017.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. Don't stop pretraining: Adapt language models to domains and tasks. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault (eds.), *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pp. 8342–8360, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.740. URL https://aclanthology.org/2020.acl-main.740.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

Yedid Hoshen. VAIN: attentional multi-agent predictive modeling. *CoRR*, abs/1706.06122, 2017. URL http://arxiv.org/abs/1706.06122.

Maximilian Hüttenrauch, Adrian Šošić, and Gerhard Neumann. Local communication protocols for learning complex swarm behaviors with deep reinforcement learning. In *ANTS Conference*, 2017.

Jiechuan Jiang and Zongqing Lu. Learning attentional communication for multi-agent cooperation. *CoRR*, abs/1805.07733, 2018. URL http://arxiv.org/abs/1805.07733.

Benjamin Keel, Aaron Quyn, David Jayne, and Samuel D. Relton. Variational autoencoders for feature exploration and malignancy prediction of lung lesions. In *British Machine Vision Conference*, 2023. URL https://api.semanticscholar.org/CorpusID:265456413.

Dong-Ki Kim, Matthew Riemer, and et al. Miao Liu. Influencing long-term behavior in multiagent reinforcement learning. *ArXiv*, abs/2203.03535, 2022.

James Kirkpatrick, Razvan Pascanu, Neil C. Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *CoRR*, abs/1612.00796, 2016. URL http://arxiv.org/abs/1612.00796.

Cecilia S Lee and Aaron Y Lee. Clinical applications of continual learning machine learning. *The Lancet Digital Health*, 2(6):e279–e281, 2020.

Timothée Lesort, Vincenzo Lomonaco, Andrei Stoian, Davide Maltoni, David Filliat, and Natalia Díaz Rodríguez. Continual learning for robotics. *CoRR*, abs/1907.00182, 2019. URL http://arxiv.org/abs/1907.00182.

Timothée Lesort, Andrei Stoian, and David Filliat. Regularization shortcomings for continual learning, 2021.

Wermter Stefan Logacjov Aleksej, Kerzel Matthias. Learning then, learning now, and every second in between: Lifelong learning with a simulated humanoid robot. *Frontiers in Neurorobotics*, 2021.

Zheda Mai, Dongsub Shim, Jihwan Jeong, Scott Sanner, Hyunwoo Kim, and Jongseong Jang. Adversarial shapley value experience replay for task-free continual learning. *CoRR*, abs/2009.00093, 2020. URL https://arxiv.org/abs/2009.00093.

Zheda Mai, Ruiwen Li, Jihwan Jeong, David Quispe, Hyunwoo Kim, and Scott Sanner. Online continual learning in image classification: An empirical survey. *CoRR*, abs/2101.10423, 2021. URL https://arxiv.org/abs/2101.10423.

Ameya Prabhu, Philip HS Torr, and Puneet K Dokania. Gdumb: A simple approach that questions our progress in continual learning. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, pp. 524–540. Springer, 2020.

Vignesh Prasad, Dipanjan Das, and Brojeshwar Bhowmick. Variational clustering: Leveraging variational autoencoders for image clustering. *CoRR*, abs/2005.04613, 2020. URL https://arxiv.org/abs/2005.04613.

Yujia Qin, Cheng Qian, Xu Han, Yankai Lin, Huadong Wang, Ruobing Xie, Zhiyuan Liu, Maosong Sun, and Jie Zhou. Recyclable tuning for continual pre-training. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Findings of the Association for Computational Linguistics: ACL 2023*, pp. 11403–11426, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.findings-acl.723. URL https://aclanthology.org/2023.findings-acl.723.

Sharan Raja, Golnaz Habibi, and Jonathan P. How. Communication-aware consensus-based decentralized task allocation in communication constrained environments. *IEEE Access*, 10:19753–19767, 2022. doi: 10.1109/ACCESS.2021.3138857.

Mohammad Rostami, Soheil Kolouri, Kyungnam Kim, and Eric Eaton. Multi-agent distributed lifelong learning for collective knowledge acquisition. *CoRR*, abs/1709.05412, 2017. URL http://arxiv.org/abs/1709.05412.

C. E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27(3): 379–423, 1948. doi: 10.1002/j.1538-7305.1948.tb01338.x.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2015.

Gido M. van de Ven, Tinne Tuytelaars, and Andreas S. Tolias. Three types of incremental learning. *Nature Machine Intelligence*, 4(12):1185–1197, 2022. ISSN 2522-5839. doi: 10.1038/s42256-022-00568-3. URL https://doi.org/10.1038/s42256-022-00568-3.

Joost Verbraeken, Matthijs Wolting, Jonathan Katzy, Jeroen Kloppenburg, Tim Verbelen, and Jan S. Rellermeyer. A survey on distributed machine learning. *CoRR*, abs/1912.09789, 2019. URL http://arxiv.org/abs/1912.09789.

Kerstin N Vokinger, Stefan Feuerriegel, and Aaron S Kesselheim. Continual learning in medical devices: Fda's action plan and beyond. *The Lancet Digital Health*, 3(6):e337–e338, 2021.

Gal Yona, Amir Feder, and Itay Laish. Useful confidence measures: Beyond the max score, 2022.

A Appendix

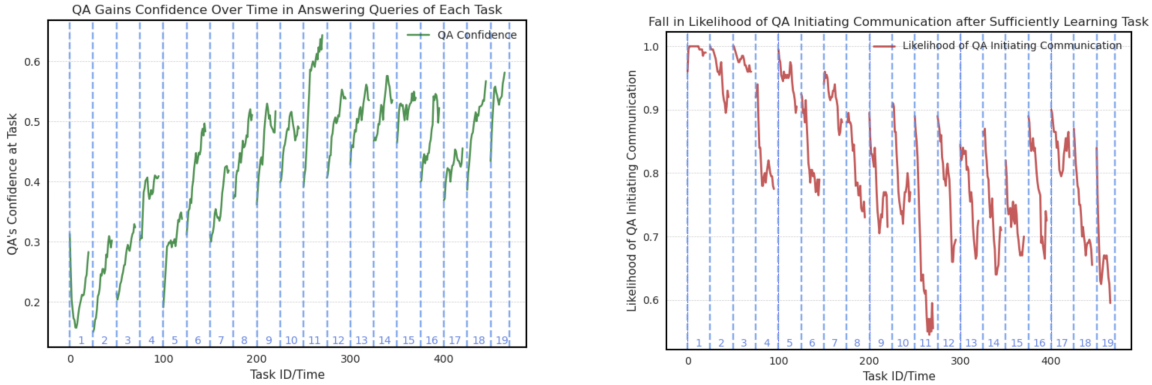


Figure 6: QA Gains Confidence & Reduces Communication: Each vertical blue line marks the introduction of a new task in the QA's lifelong learning journey on MiniImageNet. (a) Initially, the QA exhibits low confidence in queries related to the new task, but as it learns through communication with RAs, its confidence increases. (b) This increased confidence leads to self-reliance, reducing QA's number of calls to the network for responses and diminishing the system's communication overhead.

B Methodology

Here, we briefly discuss how we conduct experimental evaluations of our solutions to the different elements of the PEEPLL framework.

B.1 Confidence-Evaluation Strategies:

Setup: During lifelong learning, the QA is introduced to queries from 1.3. The QA queries the RAs for inputs when their confidence is low. The RAs (pre-trained on 1.2) answer and evaluate their confidence in the received queries. Responses with confidence higher than a certain confidence threshold are sent back to the QA.

Experimental Evaluation:

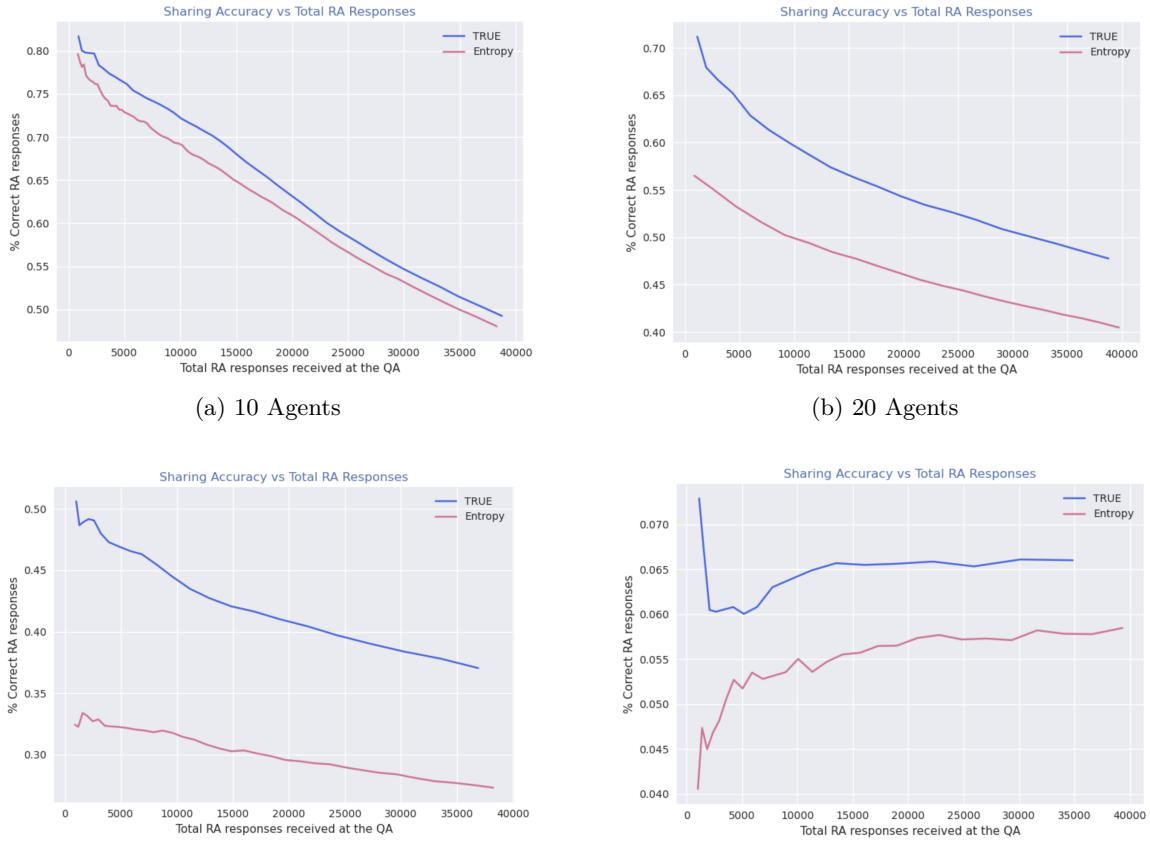
1. We measure the proportion of responses, that the RAs sent back to the QAs, that matched the ground truth at various confidence thresholds (see Figure 2a).
2. We assess the reduction in total data sent from the RAs to the QAs to achieve the same number of correct responses, compared to the entropy-based approach (see Figures 2b and 8). This metric is vital for reducing communication overhead in multi-agent settings.
3. We track the QA's growing confidence within specific tasks (Figure 4a). This helps us diminish communication needs, as the QA becomes more adept at new tasks (Figure 4b). These demonstrations will showcase the confidence metric's adaptability to the QA's continual learning capability – an essential property in lifelong learning settings.
4. We also conduct a specific analysis at a 1:1 sharing ratio, where the number of responses received by the QA from the RAs equals the total number of queries processed (corresponding to the 1.3 data segment). This controlled setting provides a standardized basis for comparison under balanced data exchange conditions.

B.2 Selective Response Filter:

Setup: Upon receipt of responses from RAs, the QA implements an additional selective response filter. This filtering mechanism leverages on the broader response pool from multiple RAs, thereby facilitating more informed decision-making.

Experimental Evaluation:

The efficacy of the Selective Response Filter is measured by its ability to increase the proportion of the



(a) 10 Agents

(b) 20 Agents

(c) 40 Agents

(d) 100 Agents

Figure 7: Sharing Accuracy Scaling with Number of Agents. As the number of agents increases, expectedly discerning the correct agent to listen to becomes increasingly challenging. Baseline probabilities for randomly selecting the correct agent are as follows: 0.1 for 10 agents, 0.05 for 20 agents, 0.025 for 40 agents, and 0.001 for 100 agents. For fair comparisons against these baselines, refer to when 20k RA responses are received at the QA, as the experiment considers 20k queries. While our current performance indicates progress, there remains significant room for improvement in developing more effective communication protocols for a higher number of agents. Notably, TRUE consistently outperforms Entropy, though their performance becomes nearly comparable with 100 agents.

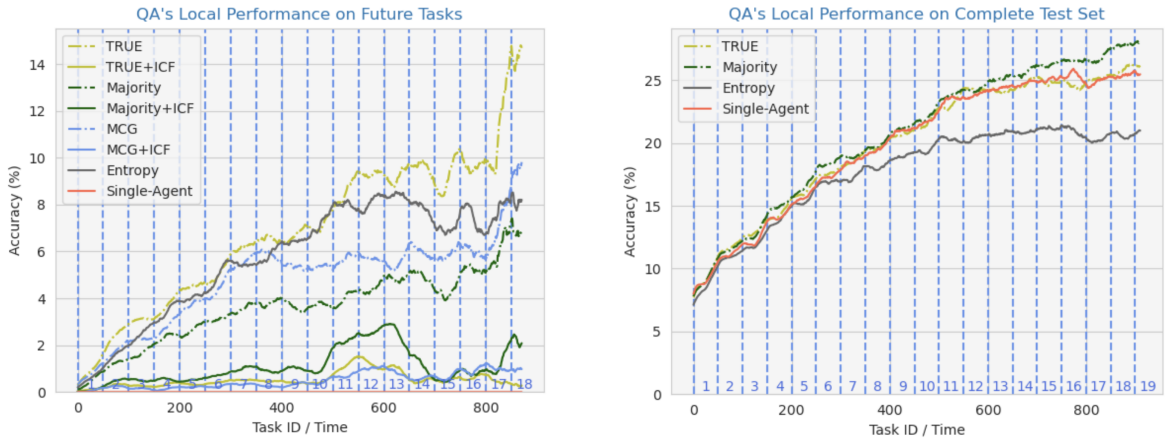
correct responses in the accepted set (Figures 3 and 10, Table 1). For instance, if the filter receives 10,000 samples with 3,000 correct responses (proportion: 30%) and subsequently filters out 5,000 samples, retaining 2,500 correct responses (proportion: 50%), this indicates an increase in the proportion of correct responses (by 20%). Such analysis underscores the filter's capacity to effectively identify and preserve the most relevant and accurate responses. Moreover, we conduct specific 1:1 Sharing Analysis for these processes as well.

B.3 Lifelong Learning:

Setup: After receiving responses from RAs, the QA further employs a selective filter to accept only the most pertinent answers. The QA then learns from this accepted set of responses.

Experimental Evaluation:

1. QA's Local Performance on the Complete Test: We assess the QA's final test performance when learning responses received using different communication strategies (Table 2 and Figure 6).
2. QA's Local Performance on Untrained Tasks: The QA's effectiveness on tasks it is not explicitly trained on is evaluated to understand the regularization effect, as discussed in Section 4.3, and the quality of 'bad'



(a) QA's performance on yet-to-be-introduced tasks.

(b) QA's performance on all tasks.

Figure 8: Each vertical blue line marks the introduction of a new task to learn for the QA. In (a), observe that QA learning with communication protocol with lower sharing accuracy exhibits higher performance on untrained tasks. In (b), observe that QA learning from peer responses performs comparably as the single-agent initially, but outperforms in the later stages. This plot corresponds to evaluation on MiniImageNet.

Algorithm 1 The PEEPLL Algorithm

Input to QA: Sample x, Maximum confidence C_{max}, Confidence threshold $C_{\text{threshold}}$
At QA: Get QA's label on x, l_x^{QA}
At QA: Get QAs confidence on x, c_x^{QA}
At QA: If $c_x^{QA} < C_{\text{max}}$, send (x, c_x^{QA}) to RAs
At RA: Get RA's label on x, l_x^{RA}
At RA: Get RA confidence on x, c_x^{RA}
At RA: If $c_x^{RA} > C_{\text{threshold}}$ and $c_x^{RA} > c_x^{QA}$, send l_x^{RA} to QA
At QA: Collect all responses by RAs
At QA: Selectively filter which responses to accept
At QA: Update Memory with accepted responses
At QA: Learn Accepted Responses + Replay memory

responses (Figure 5).

C Implementation Details

We use VGG16 Simonyan & Zisserman (2015) as the backbone for our agents under PEEPLL. We deviate from the current lifelong learning research by doing so, that currently uses ResNet18 He et al. (2015). Our methodology utilizes VGG16 due to its lack of skip connections, enabling isolation of our lifelong learning strategies' impact. ResNet18's skip connections, known to mitigate vanishing gradients, could introduce confounding factors.

PEEPLL models employ VGG16 as the encoder with a small MLP decoder for task-specific outputs. The total parameters of our PEEPLL model were 15417124.

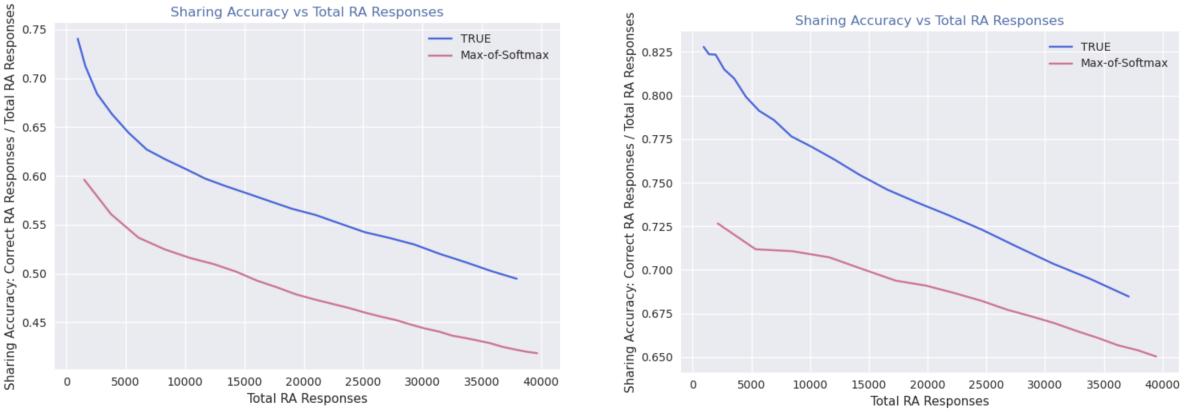
We use the same Optimizer parameters for pre-training and lifelong learning as it would not be practical to tune those parameters further for online learning.

Algorithm 2 Evaluating TRUE Confidence

1: **Input to Agent 'A':** Sample x
2: Get latent representations of x, z^x_{mean}, z^x_{logvar}
3: Prediction on x, p_x
4: Get label on x, $l_x = \text{argmax}(p_x)$
5: Get memory samples with label l_x, $\text{Memory}_{(x,l_x)}$
6: Get latent representation of retrieved memory samples, $\text{Memory}^{z_{\text{mean}}}_{(x,l_x)}$ and $\text{Memory}^{z_{\text{logvar}}}_{(x,l_x)}$
7: $d_{\text{semantic}} = ||z^x_{\text{mean}}$ - $\text{Mean}(\text{Memory}^{z_{\text{mean}}}_{(x,l_x)})||$
8: $d_{\text{dispersion}} = ||z^x_{\text{logvar}}$ - $\text{Mean}(\text{Memory}^{z_{\text{logvar}}}_{(x,l_x)})||_1$
9: $\text{entropy} = -\sum_i p_{x_i} \cdot \log_2(p_{x_i} + \epsilon)$
10: Transform into Confidence $C_{\text{semantic}} = e^{-d_{\text{semantic}}}$
11: Transform into Confidence $C_{\text{dispersion}} = e^{-d_{\text{dispersion}}}$
12: Transform into Confidence $C_{\text{entropy}} = e^{-d_{\text{entropy}}}$
13: Normalize scores
14: $\text{TRUE} = (C_{\text{semantic}} + C_{\text{dispersion}} + C_{\text{entropy}})/3$
15: **return** TRUE

D Additional Results

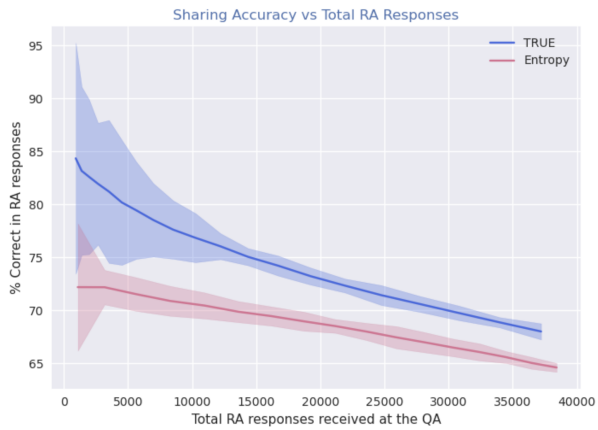
Results for Comparison of our TRUE Confidence with Max-of-Softmax are illustrated in Figure 13.



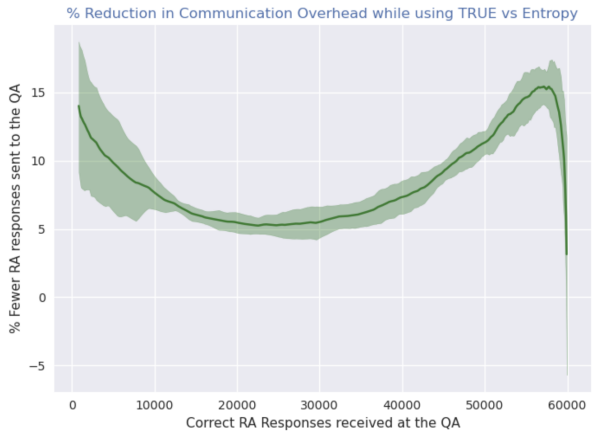
(a) CIFAR100 results

(b) MiniImageNet results

Figure 9: The TRUE score highly outperforms Max-of-Softmax as confidence. The plot shows the proportions between the total responses (from all RAs) to the responses that match the ground truth (high is better).

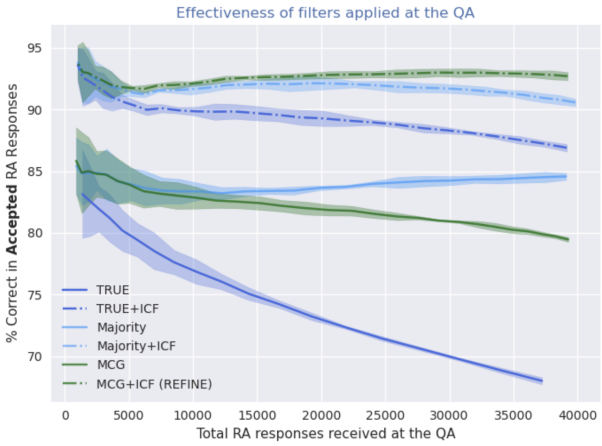


(a) TRUE vs Entropy: This plot displays the percentage of correct responses in the responses sent by RAs to the QA using TRUE and Entropy as confidence scores (high is better). TRUE outperforms Entropy consistently.



(b) A PEEPLL system using TRUE needs 5-20% fewer total RA responses (vs. Entropy) for the QA to receive the same number of correct responses. This contributes to a reduction in system communication overhead.

Figure 10: These plots correspond to evaluation on MiniImageNet.



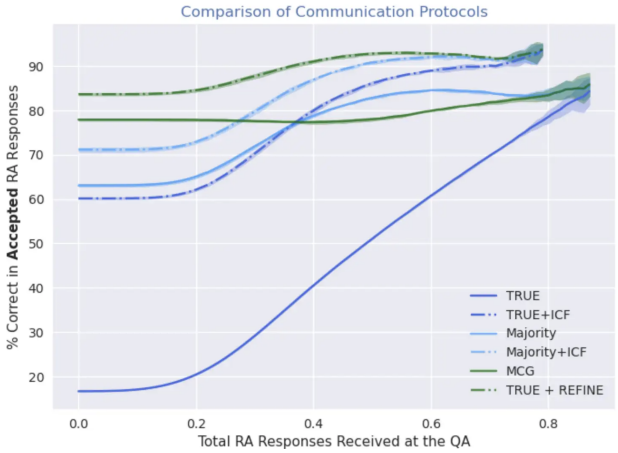


Figure 11: MiniImageNet: Each filter selectively accepts a subset of the responses received at the QA from the responses dispatched by the RAs utilizing TRUE. The figure depicts the ratio of correct responses within the selected subset. In the left figure, we cut off at a threshold of a 1:2 ratio of Queries asked to Responses received for clarity.

E Computation Complexity & Scalability

E.1 Symbols

M: Memory size, **K**: Total number of classes in the system, **K(i, t)**: Number of classes in agent i at time t, **N(q, out)**: Number of agents queried for query q, **N(q, back)**: Number of agents responding back for query q, **N**: Number of agents, **M(i, k, t)**: Number of samples in memory of class k at time t for agent i, **d**: Dimension of latent vector, **C**: Cost of communication, $\mathbf{F_N}$: Forward pass of the entire network (Encoder + Decoder), $\mathbf{F_E}$: Forward pass of the Encoder, $\mathbf{B_N}$: Backward pass of the entire network (Encoder + Decoder).

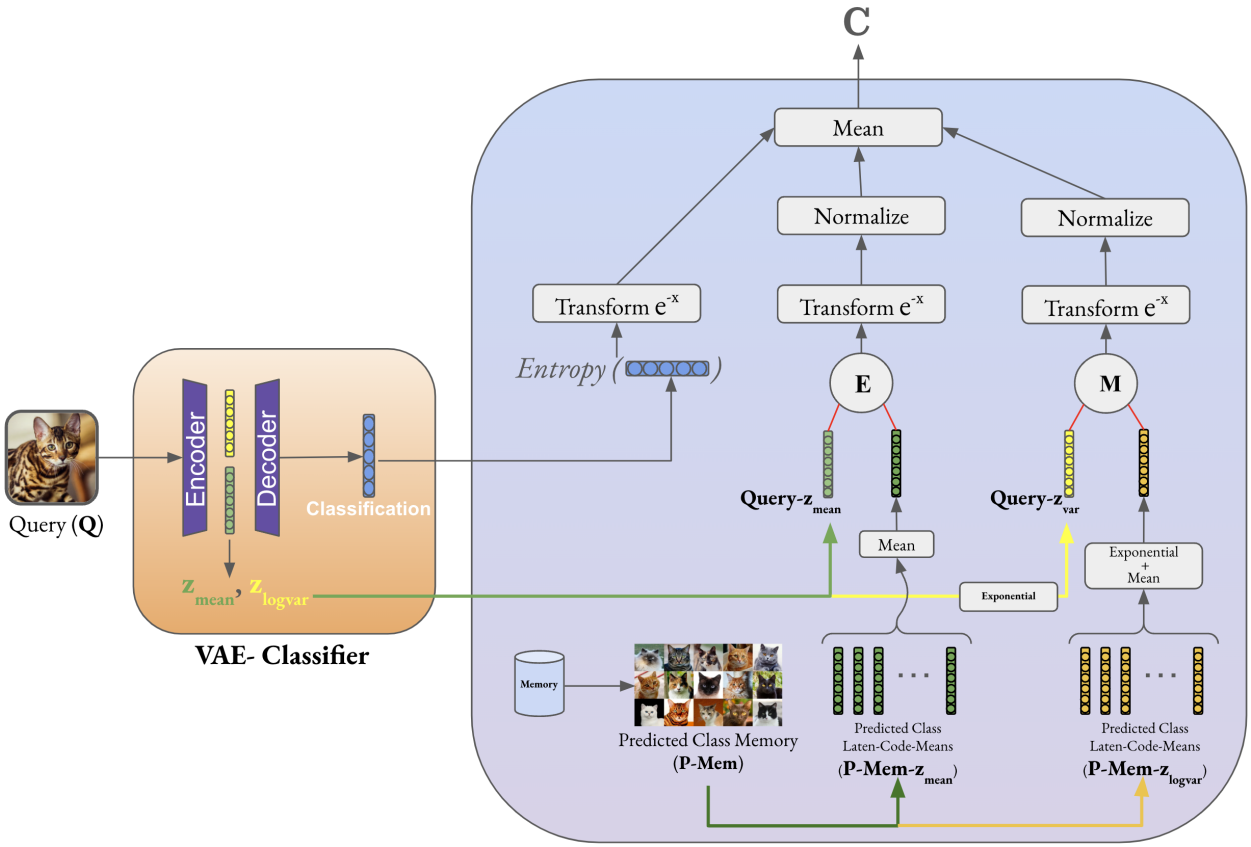


Figure 12: TRUE: Confidence Evaluation mechanism for an agent. The circle with E denotes the operation that returns the Euclidean distance between two vectors, and M the Manhattan Distance. 'Transform e^{-x}' takes in a value and maps it using the e^{-x} function.

Note $N(q, back) \leq N(q, out) \leq N$. The number of agents chosen for querying is an important research question. This selection should not be based on their previous interactions. Instead, it could be determined by factors such as proximity or other metadata.

E.2 Computational Complexity Analysis

E.3 Query Agent Complexity

Confidence Calculation

- *Get the prediction class k*: $O(\mathrm{F}_N)$
- *Find memory samples belonging to k*: $O(1)$
- *Evaluate latent codes for those memory samples*: $O(\mathrm{F}_E \times M(i, k, t))$
- *Calculate Dispersion and Semantic Distance*: $O(d \times M(i, k, t))$
- *Get the mean of Dispersion and Semantic Distance*: $O(1)$
- *Transform Mean Dispersion and Semantic Distance*: $O(1)$
- *Get Entropy of Prediction*: $O(d)$
- *Transform Entropy*: $O(1)$

- *Average three scores*: $O(1)$
- **Final Complexity:** O(F_N + $(F_E + d) \times M(i, k, t)$)

Seeking Assistance

- *Check confidence against threshold*: $O(1)$
- *Send query and confidence to other agent*: $O(N(q, out) \times C)$
- **Final Complexity:** O(N(q, out) × C)

Selective Filtering

- *Put agents into groups by their predicted class*: $O(N(q, back))$
- *Calculate each group's confidence (MCG)*: $O(1)$ (keep track while sorting)
- *Pick group with highest confidence (MCG)*: $O(1)$ (keep track while sorting)
- *Pick group with highest members (Majority)*: $O(1)$ (keep track while sorting)
- *Reject conflicting responses (ICF)*: $O(N(q, back) \times K)$* or $O(N(q, back) \times N(q, back))$**.
- **Final Complexity:** Minimum($O(N(q, back) \times K)$, $O(N(q, back) \times N(q, back))$ when ICF is used. Else, $O(N)$.

  *For a query, make a dictionary with keys as classes and values as the predictions of the agents who know that class. Maintain this dictionary as you receive the answers $O(N(q, back) \times K)$. Now, go through each response, and if the predicted class of that response has any value in the dictionary that is not the same as the key (predicted class), disregard that response O(N(q, back)). Hence, the complexity would be $O(N(q, back) \times K)$.

  **Go through each response and then check each agent if they know the response's predicted class, and if they do, compare their predictions with the prediction of the response. If they don't match, reject the response. The complexity is $O(N(q, back) \times \sum_i k(i, t)) = O(N(q, back) \times N(q, back) \times K)$, but if known classes in agents are indexed by the class id in a dictionary, we have $O(N(q, back) \times \sum_i 1)$ and the algorithm becomes $O(N(q, back) \times N(q, back))$.

Learning Update

- *Backward pass*: $O(B_N)$

Final QA Complexity:

- Minimum of the following

  O(F_N + $(F_E + d) \times M(i, k, t)$ + $N(q, out) \times C + N(q, back) \times K$ + B_N)

  O(F_N + $(F_E + d) \times M(i, k, t)$ + $N(q, out) \times C + N(q, back) \times N(q, back)$ + B_N)

E.4 Response Agent Complexity

- *Confidence Evaluation:* O(F_N + $(F_E + d) \times M(i, k, t)$)
- *Sending Response Back:* $O(C)$
- **Final RA complexity:** O(F_N + $(F_E + d) \times M(i, k, t) + C$)

E.5 Scalability Analysis

Total Complexity for Agents to exist in PEEPLL For Q queries: For an agent i, out of Q queries, Q1(i) number of times agent i serves as the QA, and Q2(i) number of times it serves as the RA. Note that Q1(i) + Q2(i) $\leq$ Q. Therefore, the total complexity is

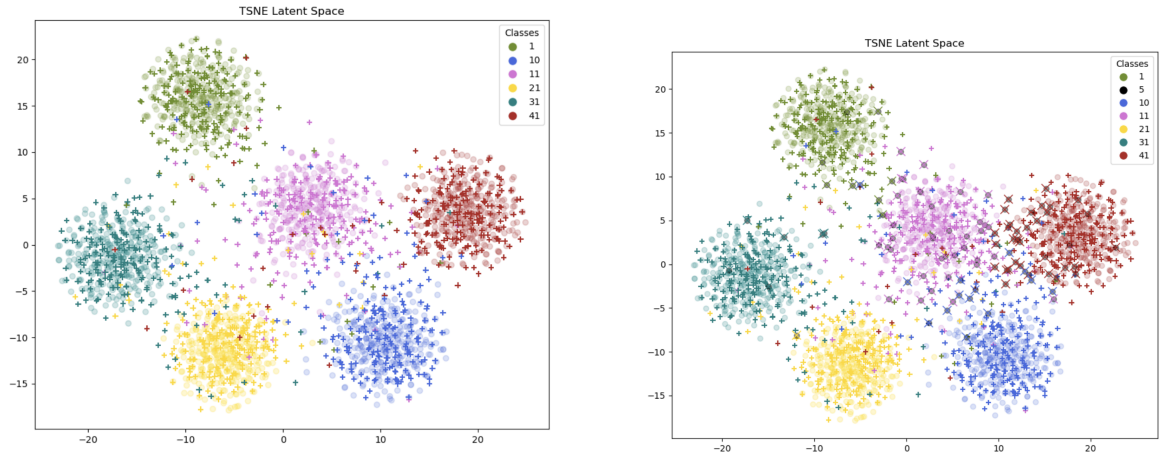
$$Q1(i)\cdot(F_N+(F_E+d)\times M(i,k,t)+N(q,out)\times C+N(q,back)\times K+B_N)+Q2(i)\cdot(F_N+(F_E+d)\times M(i,k,t)+C)$$

Generalizing the above by considering the three terms that scale - Q, N and K - we have a complexity of **O(Q×N(., out)×K)**, where N(., out) is the number of agents it reaches out for any query.

F Design Decisions

1. **Entropy in TRUE:** Confidence scores derived from Semantic and Dispersion distances alone outperform Entropy by 21.2% on CIFAR100 and 6.5% on MiniImageNet. This is comparable to TRUE (Semantic + Dispersion + Entropy) vs Entropy. Entropy provides an additional stream of information that may be beneficial for some applications.

2. **Semantic and Dispersion Distances:** Dhuliawala et al. (2024); Prasad et al. (2020); Alemi et al. (2016); Keel et al. (2023) show that the position of data in the latent space holds significance to the class it belongs to. To further support this, we plot the latent space and demonstrate the formation of clusters according to classes, as also recently shown by Prasad et al. (2020); see Figure 13. In TRUE, we measure the divergence in the mean and variance that define a datapoint's position in the latent space. Intuitively, TRUE is representative of how close a data point would be to the cluster predicted by the model in the latent space. Their inclusion is supported by ablation studies (Section 4.1, TRUE Results, para 1).



(a) Lighter dots represent the training data, and '+' denotes the test data. The colors of the dots and '+' represent its true labels. The agent has an average TRUE confidence of 0.75 for predicting the '+' data.

(b) 'x' represents data of class 5, which the model is not trained on. The color of 'x' indicates the model's prediction. The agent has an average TRUE confidence of 0.34 for predicting the 'x' data.

s

Figure 13: An agent's t-SNE visualization of its learned latent distribution of classes it is trained on, in CIFAR100. The agent is trained on classes displayed in Figure 1. Observe that the 'x' data is generally toward the edges of the clusters, and the 'x' latent vectors are not clustered themselves.