Feature Aggregation with Latent Generative Replay for Federated Continual Learning of Socially Appropriate Robot Behaviours

Nikhil Churamani[1,*], Saksham Checker[2], Hao-Tien Lewis Chiang[3], and Hatice Gunes[1]

***Abstract*— For widespread real-world applications, it is beneficial for robots to explore Federated Learning (FL) settings where several robots, deployed in parallel, can learn independently while also sharing their learning with each other. This work explores a simulated living room environment where robots need to learn the social appropriateness of their actions. We propose Federated Root (FedRoot), a novel weight aggregation strategy which disentangles feature learning across clients from individual task-based learning. Adapting popular FL strategies to use FedRoot instead, we present a novel FL benchmark for learning the social appropriateness of different robot actions in diverse social configurations. FedRoot-based methods offer competitive performance compared to others while offering sizeable (up to 86% for CPU usage and up to 72% for GPU usage) reduction in resource consumption. Furthermore, real-world interactions require social robots to dynamically adapt to changing environmental and task settings. To facilitate this, we propose Federated Latent Generative Replay (FedLGR), a novel Federated Continual Learning (FCL) strategy that uses FedRoot-based weight aggregation and embeds each client with a *generator* model for pseudo-rehearsal of learnt feature embeddings to mitigate forgetting in a resource-efficient manner. Our benchmark results demonstrate that FedRoot-based FCL methods outperform other methods while also offering sizeable (up to 84% for CPU usage and up to 92% for GPU usage) reduction in resource consumption, with FedLGR providing the best results across evaluations.**

I. Introduction

As advances in Artificial Intelligence (AI) gear social robots towards a ubiquitous application, they are expected to be deployed across environmental and contextual settings, interacting with several users at a time and learning different tasks [1]. Each robot, operating in its unique application settings, should be able to adapt to the dynamics of its ever-changing environment while also being sensitive to the changing preferences of the users it interacts with [2]. Such an understanding of social dynamics and norms [3] can help robots effectively navigate complex social settings while also offering enriching interaction experiences to their users. Moreover, given the vast array of potential applications [4], robots can benefit from sharing knowledge and individual experiences with one another. Under such complex and diverse application settings, there is a need to move beyond centralised platforms towards more *distributed learning paradigms*, enabling robots to keep learning *continually* while also sharing their learning with others.

Moving beyond *centralised* learning paradigms, where individual robots only gather data and send it to a central server to be aggregated and used to train a single global model for application, Federated Learning (FL) [5] (see Fig. 1; left) offers an efficient *distributed learning* paradigm where individual robots can learn *independently* from their own *unique experiences*, updating their learning models using only the data collected by them *locally*. Over time, these local updates can be *aggregated* across the centralised server, in the form of model updates that can inform training the unified global model. FL allows for a *privacy-preserving learning paradigm* where local data is never shared with a centralised server. FL solutions have been used popularly in embedded or EdgeAI devices [6] that benefit from *distributed* learning settings [7] gathering and processing their own data but sharing their learning towards training a global aggregated model [8]. More recently, FL has been explored for robotic and autonomous systems [1], [9] allowing for collaborative learning across robots, learning from each others' experience while maintaining end-user privacy.

Another challenge faced by social robots operating in dynamic human-centred environments is to effectively discern novel information from past experiences and adapt their learning models to accommodate this new knowledge [10]. As the real-world is not static and changes dynamically [11], each individual robot may learn with incremental and/or sequential streams of data where data is not *independent and identically distributed* (i.i.d). This may result in the robot forgetting past knowledge, overfitting to the current task or data. Even though FL allows for model aggregation across client robots, such forgetting locally will cause the globally aggregated model to also forget past knowledge. Continual Learning (CL) [11] can help address this problem by enabling individual robots to adapt their learning with incrementally acquired data from non-stationary or changing environments [2]. This allows robots to accumulate new information locally while preserving previously seen knowledge. Combining the principles of FL and CL, Federated Continual Learning (FCL) [1], [12] (see Fig. 1; right) allows for individual robots, learning incrementally in their unique settings, to also benefit from other robots' learning. Each robot periodically sends only their model parameters to the centralised server where the knowledge from all agents is aggregated into a unified model which is sent back.

[1]N. Churamani and H. Gunes are with the Department of Computer Science and Technology, University of Cambridge, UK

[2]S. Checker is with Delhi Technological University, India. S. Checker contributed to this work while undertaking a remote visiting studentship at Department of Computer Science and Technology, University of Cambridge.

[3]H.T.L. Chiang is with Google DeepMind.

This work is supported by Google under the GIG funding scheme. The authors also thank Jie Tan and Carolina Parada for their valuable feedback.

*Corresponding Author: `Nikhil.Churamani@cl.cam.ac.uk`

Code: `https://github.com/nchuramani/FedLGR_SocRob`

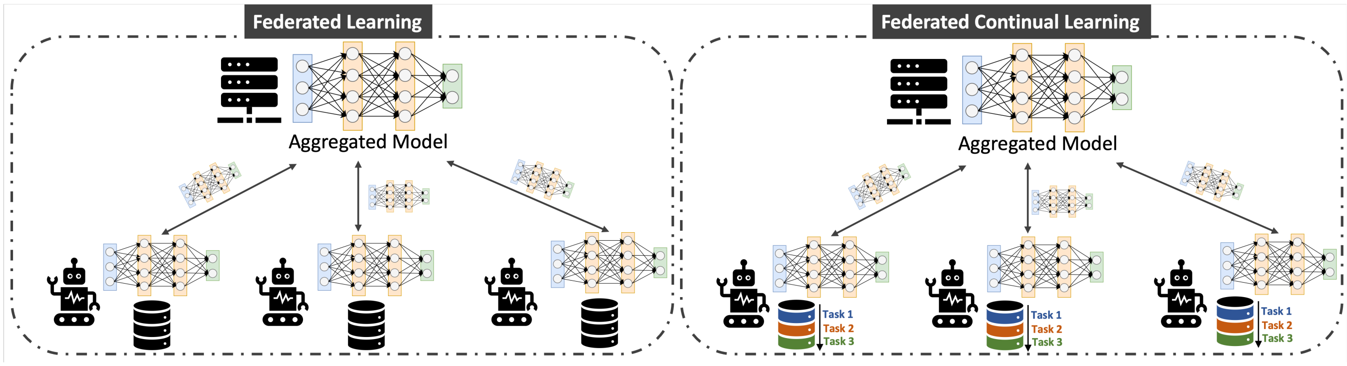

Fig. 1: **Federated Learning (FL)** (left): Local models are aggregated on the server without sharing data. **Federated Continual Learning (FCL)** (right): Individual robots incrementally learn tasks, periodically sharing model updates with each other.

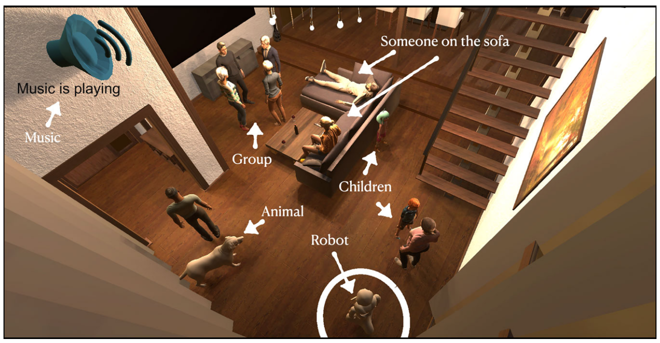

Fig. 2: MANNERS-DB: A Living Room scenario with a robot [13].

In this paper, we explore a simulated home environment with humans and animals where the robot is required to learn the social appropriateness of different tasks. We explore the MANNERS-DB dataset [13] as it has been used previously for CL-based [13] and zero-shot learning [14]. MANNERS-DB provides social-appropriateness ratings for different high-level robot actions in simulated living room settings (see Fig. 2), where the robot learns these actions in different social configurations. To simulate several robots deployed in parallel under similar living room settings, the dataset (training data) is split evenly across these robots (clients), with a shared test-set. We extend our preliminary work [15] benchmarking FL and FCL on the MANNERS-DB using the predefined $29-$d scene descriptor vectors to learning end-to-end, directly from scene renders. We propose Federated Root (FedRoot) averaging, a novel FL strategy which disentangles feature-based learning across clients from individual task-based learning. Under FedRoot, only the feature extraction layers are aggregated between *clients* however task-based learning is kept *strictly local* enhancing the resource-efficiency and privacy for FL. Adapting popular FL strategies to use FedRoot instead, we present a novel FL benchmark for learning the social appropriateness of different robot actions in diverse social configurations. Extending FedRoot to FCL settings, we propose Federated Latent Generative Replay (FedLGR), a novel FCL strategy that combines FedRoot with LGR [16] by adding a local generator model to the client for efficient pseudo-rehearsal of learnt features in order to mitigate forgetting. For successive tasks, pseudo-samples of previously seen tasks are generated and mixed with the current task's data to simulate *i.i.d* data settings. We benchmark FedLGR against popular CL adaptations of FL strategies, with and without FedRoot-based aggregation, on their ability to learn socially appropriate robot behaviours in a task-incremental manner. Both FedRoot and FedLGR reduce the memory and resource footprint of their FL and FCL counterparts, without compromising model performance.

II. Related Work

A. Socially Appropriate Robot Behaviours

Human interactions are governed by different social norms laying out expected behaviours from individuals which may be considered socially appropriate by others [17]. For social robots operating in human-centred environments, where interaction contexts and individual user preferences can vary widely, these robots must be able to comprehend and respond to the unique dynamics of each contextual setting [18]. This not only enhances their ability to navigate complex social situations but also improves the users' impressions of the robots and their acceptance in social Human-Robot Interaction (HRI) [19]. Furthermore, social robots need to *continually* learn from their interactions and adjust their behaviours accordingly [20]. Whether it is effectively navigating complex social environments [21], learning approach and positioning behaviours [22], [23] or learning task-specific behaviours [13], it is essential for robots to consider the social-appropriateness of their behaviours in order to comply with social norms [3]. With distributed deployment increasingly becoming a reality, there is a need to investigate *federated* application frameworks that allow robots to learn in their own unique environments while also informing a global learning of *generalisable* social norms and preferences [1]. This works aims to move in this direction in a more resource-efficient and privacy preserving manner.

B. Federated Learning

Distributed learning settings can be particularly desirable for social robots, to understand and learn socially appropriate behaviours, depending upon the context of the interaction, environmental factors as well as individual user preferences [1]. Federated Learning (FL) [5] enables a network of distributed client devices (such as robots) to learn individually with locally gathered data while updating their model parameters towards aggregating a global learning model (server) that combines these updates from each individual device, over successive update rounds, and distributes this model back to individual clients. This way, each client shares their knowledge with the other clients, without sharing any local data, and gets updated with global model which combines the knowledge from other clients. Strategies such as Federated Averaging (FedAvg) [8] offer a straightforward approach for weight aggregation by collating individual client model weights and computing a weighted average in the form of the global model. However, FedAvg is sensitive to data imbalances and concepts drift challenges, especially in non-i.i.d data settings. Several improvements have been proposed on FedAvg, for instance, FedBN [24] that adapts FedAvg by keeping the parameters for all the *BatchNorm* layers 'strictly local', that is, all other model weights are aggregated across clients apart from the *BatchNorm* parameters. Similar to FedBN, FedProx [25] also proposes improvements over FedAvg by allowing for only partial aggregation of weights by adding and tracking a *proximal term* to FedAvg. FedOpt [26] offers a 'general optimisation framework' where each client uses a *client optimiser* to optimise on local data while the server updates apply a gradient-based *server optimiser* to the aggregated model weights. FedDistill [27], on the other hand, aims to improve the ability of the clients to deal with *heterogeneous* data conditions by using *knowledge distillation* [28]. Each client maintains two models: (i) a local copy of the global model and (ii) a personalised model that acts as a teacher to the student global model. The updated student model is then aggregated across clients. In this work, we adapt these methods for a regression-based task, learning to predict the social appropriateness of different robot actions.

C. *Federated Continual Learning*

As robots in real-world settings continually encounter novel information, under non-i.i.d conditions, their ability to remember previously learnt tasks may progressively deteriorate, resulting in forgetting [29]. Continual Learning (CL) [11], [30] strategies may enable robots to learn and adapt throughout their 'lifetime', balancing *incremental* learning of novel information with the retention of past knowledge. Federated Continual Learning (FCL) [12] combines FL and CL principles, enabling individual robots to *incrementally* learn *without forgetting* on streams of gathered data. Individual robots learn a series of local tasks while periodically updating the parameters of a global *aggregated* model that combines updates from individual clients, over successive update rounds. Several recent works [12] explore FCL for vision [31] and natural language processing [32] applications, however very little work has been done to explore its application for social robots [1]. Adapting FL approaches by adding CL-based objectives can offer straightforward solutions for FCL. For instance, regularisation-based methods such as Elastic Weight Consolidation (EWC) [33], EWCOnline [34], Synaptic Intelligence (SI) [35] or Memory

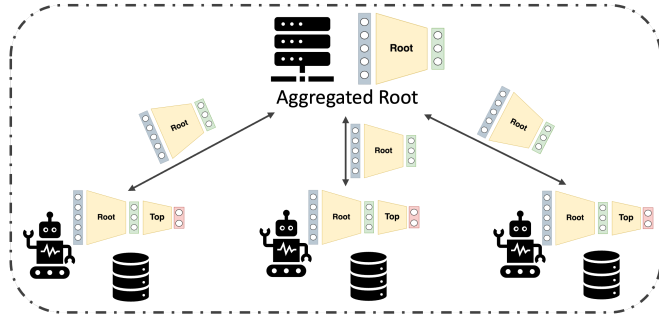

Fig. 3: **Federated Root (FedRoot)**: Local model split into (i) *Root* for feature extraction and (ii) *Top* for task-based learning. Only model *Root* is aggregated across clients while *Top* remains local.

Aware Synapses (MAS) [36] can be used to apply penalties on weight updates between old and new tasks to help mitigate forgetting. Rehearsal strategies such as Naive Rehearsal (NR) [37] can be used to maintain local memory buffers for each robot to store and rehearse previously seen data to preserve knowledge. Alternatively, an efficient pseudo-rehearsal of data [38] or features [16] can help mitigate forgetting by maintaining generators that model local data distributions for previously seen tasks. In this work, we augment FL with regularisation-based CL objectives for multi-label regression, predicting social appropriateness of the different robot actions under different contextual settings.

III. METHODOLOGY

A. *FedRoot: Federated Root for Feature Aggregation*

With several robotic clients, deployed in parallel, FL ensures end-user privacy as data remains *strictly local* for each client and is never shared with others. Yet, as the learning models, both in terms of learnt features as well as task-based learning layers are shared with the server, there might still be 'information/data leak' [39] that may reveal how end-users interact with individual robots. Furthermore, as the robots may be equipped with Deep Neural Network (DNN)-based learning models that process high-dimensional input data, aggregating individual model weights into a global model can be computationally expensive [40]. To address the above-mentioned challenges, we propose Federated Root (FedRoot) (see Fig. 3) as a novel model aggregation FL strategy that splits client learning models into two modules:

1) Root (R): The *Root* constitutes the feature extraction layers of the learning model. For example, for Convolutional Neural Network (CNN)-based learning models, as implemented in this work, the *Conv* layers would constitute the *root* of the model that learn meaningful and relevant features from input images. For robots operating in home scenarios, as discussed here, the *root* learns efficient and descriptive scene embeddings that can help summarise each scene.

2) Top (T): The *Top* of the model, on the other hand, constitutes the task-based learning (classification or regression) Fully Connected (FC) layers. Using the *Root*-extracted features, the *top* of the model is used for task-based predictions, for example, learning to predict the social appropriateness of different robot actions.

Splitting model learning into two parts allows for robots to effectively share their learning with other clients while also protecting end-user privacy. The model is trained in an end-to-end manner, however, only the *root*, that is, the feature extraction layers, from individual client models gets aggregated across clients, over repeated aggregation rounds. This allows for the clients to improve their feature-embeddings, making them more robust, benefiting from the diversity of data settings experienced by each individual client. Yet, client-specific data as well as the task-based learning from the data, for instance end-user preferences on social appropriateness of individual robot behaviours, is kept 'strictly local'. Since FedRoot is proposed as a weight aggregation strategy, we adapt popular FL approaches (see Section II-B) to use FedRoot instead.

B. FedLGR: Federated Latent Generative Replay

Real-world application adds another complexity for robotic clients with respect to *incremental learning* where they are exposed to information in an incremental manner and have to constantly learn new tasks while preserving past knowledge [2], [29]. While replay-based CL may provide a straightforward approach for remembering past information by storing and periodically *replaying* past data [37], this may not always be possible due to memory constraints or privacy and compliance reasons. *Pseudo-rehearsal* approaches such as Deep Generative Replay (DGR) [41], on the other hand, use probabilistic or generative models that learn the inherent data statistics of previously seen data and draw *pseudo-samples* as and when needed, to be replayed to the model. These generative models, however, become hard to train as the number of tasks increases as the generator needs to learn to reconstruct high-quality discriminative pseudo-samples, adding a significant computational expense to the model. Generative feature rehearsal approaches such as Latent Generative Replay (LGR) [16] offer resource-efficient pseudo-rehearsal strategies that combine the benefits of DGR [41], with using low-dimensional latent features [42].

Under FCL learning settings clients not only need to incrementally learn on the acquired 'local data' ensuring they do not forget past knowledge, they also need to aggregate this learning in a centralised global model. To achieve this, we propose Federated Latent Generative Replay (FedLGR) (see Fig. 4) as a novel FCL strategy that applies LGR-based pseudo-rehearsal [16] under *federated learning* settings making use of FedRoot-based weight aggregation.

FedLGR uses a *scholar*-based architecture [16] that consists of three modules: (i) a *Root* to extract latent feature representations, (ii) a *Top* to learn task-discriminative information, and (iii) a *Generator* to reconstruct and rehearse *latent features*. Under FedLGR, each client locally uses LGR to *continually* and *incrementally* learn the social appropriateness of robot actions under different contextual settings. This learning is then shared with other clients by aggregating *Root* weights using FedRoot. At each time-step, the learning for each client *locally* consists of the following steps:

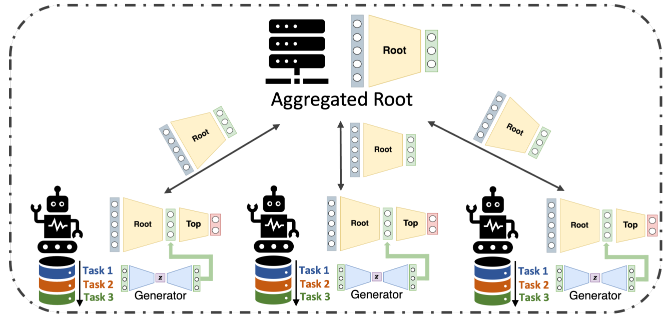

Fig. 4: **Federated Latent Generative Replay (FedLGR)**: Local model split into (i) *Root* for feature extraction, (ii) *Top* for task-based learning and (iii) *Generator* for *pseudo-rehearsal* of features. Only *Root* is aggregated across clients while *Top* and *Generator* remain local.

1) The *root* and *top* for the *scholar* are updated with the current task's data, however at different learning rates. While the *top* is trained rapidly to learn task-discriminative information, the *root* is updated *slowly* ensuring the new data does not completely *overwrite* previously seen information. This is an important step as LGR [16] is based on the assumption that if the latent-space distribution of *root*-extracted features remains relatively static between model updates, the extracted features can be effectively used to rehearse past knowledge [42].
2) For training the *generator*, the *Root*-extracted feature representations $(R(x))$ for the current task's data (x) are interleaved with *generated* features (R'_x) for all previously seen tasks. Training the *generator* on both $R(x)$ and R'_x (only $R(x)$ is used for Task 1) ensures that the updated *generator* encodes both new and old tasks. R'_x is passed to the *top* to obtain labels $T(R'_x)$.
3) Once the *generator* is updated, the *top* of the *solver* is updated using the current task data $\langle R(x), y\rangle$, interleaved with labelled *latent pseudo-samples* $\langle R'_x, T(R'_x)\rangle$ generated for previously seen tasks. The *root* is then updated *slowly* using only current task's data.

Training on each task consists of several rounds of weight aggregations, where FedRoot is used to aggregate *root* module weights across clients. The *top* and the *generator* for each client are kept 'strictly local' ensuring task-discriminative information is never shared.

IV. Experiments

A. MANNERS-DB Dataset

For our proof-of-concept evaluations on FL and FCL for learning socially appropriate robot behaviours, we explore the MANNERS-DB dataset [13] that consists of Unity-generated scenes of the Pepper robot co-inhabiting a living room (see Figure 2) with other humans and animals under different social settings. MANNERS-DB has been explored for CL-based [13] and zero-shot [14] learning, however, not for FCL. For each scene, the robot can perform 8 different actions, that is, *vacuuming, mopping, carrying warm food, carrying cold food, carrying big objects, carrying small objects, carrying drinks and cleaning/starting conversations* either *within a circle of influence* or in the *direction of operation* (see Figure 2). Crowd-sourced annotations are provided for the social appropriateness of each of these actions for every scene ($\approx$ 1000 scenes), labelled on a 5-point Likert scale, ranging from *very inappropriate* to *very appropriate*. For both FL and FCL evaluations, the data is split into training and test splits in the ratio of $3:1$. The training data is further split amongst the different clients (2 or 10) with a shared test-set used for evaluation. For FCL evaluations, the training set of individual clients is further split into two tasks, that is, samples depicting the robot operating with an circle (**Task 1**) and in the direction of the arrow (**Task 2**). For training the models, *normalised* RGB images are used, resized to $(128\times128\times3)$ due computational restrictions of training multiple clients in parallel on GPU. Since MANNERS-DB is a relatively small dataset consisting of $\approx$ 1000 images, we evaluate the different approaches both *without* and *with* data-augmentation. We use random ($p=0.5$) horizontal flipping and random ($p=0.5$) rotation (up to $10°$) in either direction to augment the dataset. For a fair comparison, results are presented individually for *without* and *with* augmentation comparisons.

B. Implementation Details

For each approach, a Convolutional Neural Network (CNN)-based model is used consisting of two parts: (i) a *Conv* module that uses the popularly used MobileNet-V2 [43] backbone to extract scene features, followed by an *AdaptiveAvgPool* layer for dimensionality reduction, and (ii) an *FC* module where the $1280-$d *flattened* scene features are passed through a FC layer consisting of 32 units followed by the $8-$unit output layer. Each layer in the *FC* module is followed by *BatchNorm* layer and uses a *linear* activation with the output layer predicting the *social appropriateness* of all 8 robot actions. All models are trained using the *Adam* optimiser (β_1=0.9, β_2=0.999) with *learning rate* of lr=$1e^{-}3$. All methods are implemented using the *Flower*[1] and PyTorch Python libraries. The FL and FCL experiments are run for $2-10$ clients, attuned to a potential real-world evaluation to be conducted in the future using physical robots. For brevity, results for only 2 and 10 clients are presented. Model hyper-parameters such as the regularisation coefficients for FCL approaches are optimised using a grid-search. For without augmentation experiments, the *BatchSize* is empirically set to 16 for *two clients* and 8 for *ten clients* due to the overall low number of samples per client. When using with augmentation, *BatchSize* is set to 16. All models are trained using an Mean Squared Error (MSE) loss, computed as an average across the 8 actions. Each client agent trains on its own split of the training data but a shared test-set is used for model evaluation. We report the average MSE loss averaged across the 8 actions and all clients.

C. Evaluation Metrics

Predicting the social-appropriateness for each robot action requires transforming FL and FCL methods to use regression-based objectives, using the following metrics:

- *Root Mean Squared Error (RMSE):* We report the RMSE values which are calculated for the test-set, averaged across the 8 actions across all clients. RMSE scores are used to compare our results with the baseline provided with the MANNERS-DB dataset [13].
- *Pearson's Correlation Coefficient (PCC):* In addition to an absolute metric such as the RMSE, we also evaluate the predicted social appropriateness values relative to the ground-truth using the PCC [44] scores. PCC scores are calculated on the test-set individually for each of the 8 actions, for each client. Average scores across all actions and all clients are reported.
- *CPU Usage (s):* To evaluate the resource efficiency of FL and FCL approaches, CPU usage is reported as the CPU time (in seconds) allocated to training each client, on average. This is calculated as the CPU time utilised by each client, across all rounds of weight aggregation.
- *GPU Usage (%):* Similar to CPU usage, GPU usage is also important to evaluate the resource efficiency of the proposed methods. Since we have a single GPU simulation set-up, we report the percentage (%) of the GPU allocated per client, on average. GPU usage is also calculated using Nvidia's `nvidia-smi` tool, logged at different intervals during the training process, and *averaged* over the entire training time for each client.

[1] https://flower.dev

TABLE I: Federated Learning Results for the MANNERS-DB Dataset for Two (left) and Ten (Right) Clients. **Bold** values denote best while [bracketed] denote second-best values.

	Two Clients			*Ten Clients*		
Method	Loss ▼	RMSE ▼	PCC ▲	Loss ▼	RMSE ▼	PCC ▲
	W/O Augmentation					
FedAvg	0.264	0.510	[0.508]	0.220	0.467	0.487
FedBN	0.310	0.553	0.486	0.226	0.471	0.486
FedProx	[0.262]	[0.507]	**0.513**	[0.211]	[0.459]	**0.517**
FedOpt	**0.245**	**0.492**	0.501	**0.210**	**0.458**	[0.492]
FedDistill	0.276	0.522	0.475	0.233	0.481	0.470
FedRootAvg	0.276	0.522	0.482	0.231	0.477	0.459
FedRootBN	0.272	0.518	0.498	0.263	0.510	0.420
FedRootProx	0.264	0.511	0.481	0.240	0.486	0.447
FedRootOpt	0.285	0.530	0.473	0.240	0.486	0.460
FedRootDistill	0.294	0.539	0.489	0.305	0.550	0.363
	W/ Augmentation					
FedAvg	0.192	0.435	0.547	0.178	[0.420]	**0.586**
FedBN	0.205	0.447	[0.554]	0.188	0.429	0.567
FedProx	0.208	0.451	0.536	**0.173**	**0.412**	[0.581]
FedOpt	0.204	0.450	0.506	[0.177]	[0.420]	0.568
FedDistill	0.221	0.468	0.538	0.205	0.449	0.560
FedRootAvg	0.192	0.433	0.547	0.198	0.440	0.563
FedRootBN	0.197	0.442	**0.558**	0.194	0.439	0.539
FedRootProx	**0.186**	**0.427**	0.541	0.196	0.439	0.552
FedRootOpt	[0.190]	[0.432]	0.528	0.206	0.450	0.549
FedRootDistill	0.225	0.469	0.525	0.222	0.465	0.539

V. Results

A. Federated Learning Benchmark

Table I presents the FL benchmark results on the MANNERS-DB dataset comparing popular FL strategies and their FedRoot-based adaptations for *two* and *ten* clients,

TABLE II: Average CPU Usage (s) and GPU Usage (%) per client, per aggregation round for Federated Learning experiments.

Metric	FedAvg	FedRootAvg	FedBN	FedRootBN	FedProx	FedRootProx	FedOpt	FedRootOpt	FedDistill	FedRootDistill
CPU Usage (s) ▼	19.06	**2.55** (▼ 86.6%)	18.4	**2.50** (▼ 86.4%)	18.98	**2.54** (▼ 86.6%)	18.61	**2.46** (▼ 86.7%)	19.46	**2.61** (▼ 86.4%)
GPU Usage (%) ▼	0.63	**0.24** (▼ 61.9%)	0.93	**0.27** (▼ 70.9%)	0.25	**0.15** (▼ 40.0%)	0.69	**0.19** (▼ 72.4%)	0.10	**0.07** (▼ 30.0%)

TABLE III: Average CPU Usage (s) and GPU Usage (%) per client, per aggregation round for Federated Continual Learning experiments.

Metric	FedAvg EWC	FedRoot EWC	FedAvg EWCOnline	FedRoot EWCOnline	FedAvg MAS	FedRoot MAS	FedAvg SI	FedRoot SI	FedAvg NR	FedRoot NR	FedLGR
CPU Usage (s) ▼	24.17	**3.87** (▼ 83.9%)	25.05	**4.97** (▼ 80.2%)	22.41	**5.36** (▼ 76.1%)	20.32	**3.91** (▼ 80.7%)	25.97	**4.05** (▼ 84.4%)	**4.06**
GPU Usage (%) ▼	0.57	**0.04** (▼ 92.9%)	0.13	**0.03** (▼ 76.9%)	0.04	**0.01** (▼ 75.0%)	0.10	**0.05** (▼ 50.0%)	0.06	**0.04** (▼ 33.3%)	**0.02**

both without and with data augmentation, respectively. For without data augmentation experiments with *two* and *ten clients* FedOpt and FedProx emerge as the best performing approaches, on average. The 'general optimisation framework' of FedOpt, with *Adam*-based client and server optimisers are able to efficiently aggregate learning across clients, resulting in the robust performance of the model. FedProx, on the other hand, adds a proximal term $\mu = 0.1$ (similar to [25]) to FedAvg and updates the objective for each client to minimise $F_k(\omega) + \frac{\mu}{2}||\omega - \omega^t||^2$ where F_k is the loss, ω are the local model weights to optimise and ω^t are the global parameters at time-step t. For the proposed FedRoot variants of the compared FL approaches, as task-discriminative *top* weights are kept *strictly local*, we see an overall drop in model performance. This is primarily due to the small size of the MANNERS-DB dataset which does have enough data samples for the FedRoot to optimise model performance, given only feature extracting *root* is aggregated. However, FedRoot offers *sizeable reduction* in CPU and GPU Usage per client, per aggregation round, compared to the original methods, as can be seen in Table II.

Data augmentation has a net positive impact on all models with all approaches reporting better metrics, across evaluations. With the larger amount of data available per client, we see that FedRoot-based methods improve upon their counterparts across all metrics, for the experiments with *two clients*. Similar to without augmentation experiments, using *Adam*-based client and server optimisation as well as adding a proximal term to the model learning objective results in the best performance, however here for FedRootOpt and FedRootProx, respectively. Data available per client still remains relatively low when split across *ten clients*, even when using data augmentation. Thus, despite an overall improvement compared to without augmentation results FedRoot-based approaches are still not able to match the performance of FedProx and FedOpt for *ten clients*, despite offering high resource-efficiency.

B. Federated Continual Learning Benchmark

Table IV presents the FCL results on the MANNERS-DB dataset adapting FedAvg and FedRoot-based weight aggregation strategies to use CL-based learning objectives to mitigate forgetting under incremental settings. Evaluation metrics are reported after training and testing on the data-split depicting the robot operating *within the circle of influence* **(Task 1)** and training on the data-split depicting the robot *in the direction of operation* **(Task 2)** and testing on both the splits. Similar to the FL benchmark, we compare the different FCL strategies for *two* and *ten* clients as well as without and with data augmentation. As *federated averaging* of weights struggles with learning under non-i.i.d data settings [24], extending these methods with CL-based objectives allows us to evaluate how they can objectively contribute towards maintaining model performance when learning *incrementally*.

Our results demonstrate that FedRoot-based approaches are able to outperform their FedAvg-based counterparts across most evaluations. However, similar to FL evaluations, FedRoot-based methods struggle in the scarcity of data, for instance the evaluations across *ten* clients without data augmentation, resulting in much worse loss and RMSE scores. Data augmentation has a net positive impact on model performances, especially for FedRoot-based methods. Furthermore, using FedRoot-based weight aggregation results in a sizeable (up to *84%* for CPU usage and *92%* for GPU usage) reduction in computational expense of running these methods, per client, per aggregation round (see Table III). This can be particularly beneficial for application in resource-constrained devices such as social robots. The proposed FedLGR approach performs the best across all evaluations. This is especially true when data is scarce, that is learning without using data augmentation across *two* and *ten* clients. FedLGR is able to efficiently use pseudo-rehearsal of features to maintain model performance after learning Task 2.

VI. Discussion and Conclusion

Social robots operating in dynamic real-world settings can benefit from federated learning mechanisms where, learning from and adapting towards their unique environmental and data conditions, they can also share their learning with other robots, benefiting from each others' experiences. FL-based approaches enable such a learning paradigm for robots to aggregate model updates across individual agents into a global model, while ensuring that end-user privacy is preserved. However, the resource constraints inhibit such learning settings resulting in most existing methods opting for *centralised* learning, where end-user devices only collect data for a centralised model to be trained, in isolation, for later application *in-the-wild*. The proposed Federated Root (FedRoot) weight aggregation strategy aims to address this very challenge by splitting each client's learning model into

TABLE IV: Federated Continual Learning Results for the MANNERS-DB Dataset for Two (left) and Ten (Right) Clients. **Bold** values denote best while [bracketed] denote second-best values.

	Two Clients						*Ten Clients*					
	After Task 1			After Task 2			After Task 1			After Task 2		
Method	Loss ▼	RMSE ▼	PCC ▲	Loss ▼	RMSE ▼	PCC ▲	Loss ▼	RMSE ▼	PCC ▲	Loss ▼	RMSE ▼	PCC ▲
	W/O Data-augmentation											
FedAvgEWC	0.705	0.826	0.488	0.911	0.944	0.347	0.364	0.591	0.325	0.393	0.617	0.278
FedAvgEWCOnline	0.488	0.685	0.469	0.381	0.608	0.384	[0.328]	[0.566]	0.373	[0.359]	[0.589]	0.321
FedAvgMAS	0.568	0.740	0.570	0.363	0.597	0.357	0.421	0.641	[0.494]	0.464	0.674	0.357
FedAvgSI	0.488	0.686	[0.571]	0.675	0.811	0.343	0.451	0.665	0.323	0.376	0.598	0.306
FedAvgNR	0.403	0.634	0.491	[0.296]	[0.538]	0.333	0.376	0.600	0.447	0.408	0.628	0.341
FedRootEWC	0.433	0.652	0.472	0.998	0.963	0.382	9.320	3.050	0.336	8.950	2.990	0.335
FedRootEWCOnline	0.614	0.778	0.436	0.447	0.659	0.400	9.170	3.030	0.330	9.560	3.090	[0.376]
FedRootMAS	[0.397]	[0.627]	0.429	0.569	0.740	0.422	8.860	2.980	0.363	8.750	2.960	0.369
FedRootSI	0.968	0.984	0.441	0.840	0.893	0.418	9.360	3.060	0.417	9.120	3.020	0.369
FedRootNR	0.762	0.870	0.325	**0.287**	**0.533**	[0.444]	0.446	0.656	0.462	0.482	0.677	0.339
FedLGR	**0.252**	**0.499**	**0.599**	0.465	0.677	**0.534**	**0.299**	**0.539**	**0.515**	**0.320**	**0.553**	**0.421**
	W/ Data-augmentation											
FedAvgEWC	0.669	0.807	0.420	0.348	0.584	0.395	0.397	0.624	0.472	0.433	0.653	0.456
FedAvgEWCOnline	0.495	0.702	0.438	0.629	0.788	0.376	[0.319]	[0.561]	0.365	[0.329]	[0.579]	0.363
FedAvgMAS	0.267	0.514	**0.601**	0.665	0.814	0.394	0.343	0.580	0.480	0.354	0.587	0.328
FedAvgSI	0.280	0.526	[0.546]	[0.328]	[0.568]	0.444	0.471	0.682	0.418	0.435	0.654	0.426
FedAvgNR	0.450	0.669	0.531	0.710	0.839	0.451	0.392	0.618	0.425	0.490	0.698	0.368
FedRootEWC	0.736	0.836	0.529	**0.327**	**0.569**	0.463	0.441	0.648	[0.543]	0.449	0.655	[0.467]
FedRootEWCOnline	0.517	0.712	0.398	0.446	0.661	0.363	0.688	0.805	0.446	0.612	0.761	0.405
FedRootMAS	[0.265]	[0.511]	0.492	0.616	0.782	0.436	0.797	0.851	0.542	1.030	0.930	0.439
FedRootSI	0.388	0.596	0.405	0.735	0.821	[0.478]	0.481	0.688	0.323	0.521	0.708	0.439
FedRootNR	0.781	0.875	0.491	0.368	0.604	0.354	0.539	0.728	0.268	0.409	0.635	0.447
FedLGR	**0.230**	**0.478**	0.531	0.493	0.699	**0.479**	**0.288**	**0.528**	**0.563**	**0.317**	**0.560**	**0.482**

aggregatable feature extraction layers, that is, model *root* and *private* task-relevant *top* layers that learn to predict the social appropriateness of different robot actions. The FL benchmark results (see Table I and Table II) highlight the competitive performance of FedRoot-based approaches with sizeable reductions in CPU and GPU usage for each client.

Furthermore, real-world applications may require robots to learn incrementally, for instance, in the form of developing novel capabilities or applying existing capabilities under novel contextual settings. In this work, we explore the latter where robots need to learn the social appropriateness of different actions depending upon the context in which they are operating, that is, operating *within the circle of influence* or *in a particular direction of operation*. We adapt the proposed FedRootAvg weight aggregation strategy and extend it by adapting popular CL-based learning objectives presenting a novel FCL benchmark on the MANNERS-DB dataset (see Table IV and Table III). In particular, the proposed Federated Latent Generative Replay (FedLGR) approach is seen to outperform other methods across all evaluations. It implements a local *generator* for efficient pseudo-rehearsal of latent features for mitigating forgetting, where the *top* learns task-relevant information. Our work contributes significantly to the fields of Federated Learning and robotics, offering promising avenues for the development of socially intelligent machines capable of learning and adapting in a decentralized and resource-efficient manner.

A. Limitations and Future Work

We present novel FL and FCL benchmarks for learning social appropriateness of high-level robot behaviours in simulated home settings. With our motivating results, we wish to conduct a user study with 2+ robots operating in different living room settings, sharing their knowledge with each other. However, MANNERS-DB is a relatively small dataset which limits our evaluations into the generalisability of proposed approaches across a large number of clients. In future, we wish to expand the MANNERS-DB dataset to include more scenes as well as different robot embodiments such as Pepper, Nao, PR2 robots, and points-of-view (PoVs) (robot-centric and scene-centric) to extend our evaluations (both FL and FCL), under simulation, to 100+ clients. This would result in a more generalisable evaluation for large-scale *federated* application for robots in diverse real-world settings. Additionally. it will be beneficial to explore further the scalability of FedRoot and FedLGR in more complex or dynamic environments, especially when data is scarce and data acquisition is complicated, as well as their applicability to a broader range of tasks beyond social appropriateness.

References

[1] L. Guerdan and H. Gunes, "Federated Continual Learning for Socially Aware Robotics," in *IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*, pp. 1522–1529, 2023.

[2] N. Churamani, S. Kalkan, and H. Gunes, "Continual Learning for Affective Robotics: Why, What and How?," in *IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*, pp. 425–431, IEEE, 2020.

[3] A. Ayub and A. R. Wagner, "What Am I Allowed to Do Here?: Online Learning of Context-Specific Norms by Pepper," in *Social Robotics*, pp. 220–231, Springer International Publishing, 2020.

[4] H. Mahdi, S. A. Akgun, S. Saleh, and K. Dautenhahn, "A survey on the design and evolution of social robots — past, present and future," *Robotics and Autonomous Systems*, vol. 156, p. 104193, 2022.

[5] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y. Arcas, "Communication-Efficient Learning of Deep Networks from Decentralized Data," in *International Conference on Artificial Intelligence and Statistics*, vol. 54 of *Proceedings of Machine Learning Research*, pp. 1273–1282, PMLR, 2017.

[6] A. Imteaj, U. Thakker, S. Wang, J. Li, and M. H. Amini, "A Survey on Federated Learning for Resource-Constrained IoT Devices," *IEEE Internet of Things Journal*, vol. 9, no. 1, pp. 1–24, 2022.

[7] C. Zhang, Y. Xie, H. Bai, B. Yu, W. Li, and Y. Gao, "A survey on Federated Learning," *Knowledge-Based Systems*, vol. 216, 2021.

[8] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the Convergence of FedAvg on Non-IID Data," in *International Conference on Learning Representations*, 2020.

[9] Y. Xianjia, J. P. Queralta, J. Heikkonen, and T. Westerlund, "Federated Learning in Robotic and Autonomous Systems," *Procedia Computer Science*, vol. 191, pp. 135–142, 2021.

[10] Z. Chen and B. Liu, *Lifelong Machine Learning*. Springer International Publishing, 2018.

[11] R. Hadsell, D. Rao, A. A. Rusu, and R. Pascanu, "Embracing Change: Continual Learning in Deep Neural Networks," *Trends in Cognitive Sciences*, vol. 24, no. 12, pp. 1028–1040, 2020.

[12] J. Yoon, W. Jeong, G. Lee, E. Yang, and S. J. Hwang, "Federated Continual Learning with Weighted Inter-client Transfer," in *International Conference on Machine Learning*, vol. 139, pp. 12073–12086, PMLR, 2021.

[13] J. Tjomsland, S. Kalkan, and H. Gunes, "Mind Your Manners! A Dataset and a Continual Learning Approach for Assessing Social Appropriateness of Robot Actions," *Frontiers in Robotics and AI*, vol. 9, 2022.

[14] B. Zhang and H. Soh, "Large language models as zero-shot human models for human-robot interaction," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 7961–7968, 2023.

[15] S. Checker, N. Churamani, and H. Gunes, "Federated Learning of Socially Appropriate Agent Behaviours in Simulated Home Environments," in *Workshop on Lifelong Learning and Personalization in Long-Term Human-Robot Interaction (LEAP-HRI), 19th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, 2024.

[16] S. Stoychev, N. Churamani, and H. Gunes, "Latent Generative Replay for Resource-Efficient Continual Learning of Facial Expressions," in *17th IEEE International Conference on Automatic Face and Gesture Recognition (FG)*, pp. 1–8, 2023.

[17] O. Roesler, E. Bagheri, A. Aly, S. Rossi, and R. Alami, "Socially acceptable robot behavior: Approaches for learning, adaptation and evaluation," *Interaction Studies*, vol. 23, no. Socially Acceptable Robot Behavior, pp. 355–359, 2022.

[18] H. Li, S. Milani, V. Krishnamoorthy, M. Lewis, and K. Sycara, "Perceptions of Domestic Robots' Normative Behavior Across Cultures," in *AAAI/ACM Conference on AI, Ethics, and Society*, AIES '19, p. 345–351, ACM, 2019.

[19] O. Roesler, E. Bagheri, and A. Aly, "Toward understanding the effects of socially aware robot behavior," *Interaction Studies*, vol. 23, no. Socially Acceptable Robot Behavior, pp. 513–552, 2022.

[20] K. Shiarlis, J. Messias, and S. Whiteson, "Acquiring social interaction behaviours for telepresence robots via deep learning from demonstration," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 37–42, 2017.

[21] A. Francis, C. Pérez-d'Arpino, C. Li, F. Xia, A. Alahi, R. Alami, A. Bera, A. Biswas, J. Biswas, R. Chandra, *et al.*, "Principles and guidelines for evaluating social robot navigation algorithms," *arXiv preprint arXiv:2306.16740*, 2023.

[22] Y. Gao, F. Yang, M. Frisk, D. Hemandez, C. Peters, and G. Castellano, "Learning socially appropriate robot approaching behavior toward groups using deep reinforcement learning," in *2019 28th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*, pp. 1–8, IEEE, 2019.

[23] E. McQuillin, N. Churamani, and H. Gunes, "Learning Socially Appropriate Robo-Waiter Behaviours through Real-Time User Feedback," in *ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, HRI '22, p. 541–550, IEEE Press, 2022.

[24] X. Li, M. Jiang, X. Zhang, M. Kamp, and Q. Dou, "FedBN: Federated Learning on Non-IID Features via Local Batch Normalization," in *International Conference on Learning Representations*, 2021.

[25] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated Optimization in Heterogeneous Networks," in *Proceedings of Machine Learning and Systems*, vol. 2, pp. 429–450, 2020.

[26] S. J. Reddi, Z. Charles, M. Zaheer, Z. Garrett, K. Rush, J. Konečný, S. Kumar, and H. B. McMahan, "Adaptive Federated Optimization," in *International Conference on Learning Representations*, 2021.

[27] D. Jiang, C. Shan, and Z. Zhang, "Federated Learning Algorithm Based on Knowledge Distillation," pp. 163–167, IEEE, 2020.

[28] G. Hinton, O. Vinyals, and J. Dean, "Distilling the Knowledge in a Neural Network," in *NIPS Deep Learning and Representation Learning Workshop*, 2015.

[29] T. Lesort, V. Lomonaco, A. Stoian, D. Maltoni, D. Filliat, and N. Díaz-Rodríguez, "Continual learning for robotics: Definition, framework, learning strategies, opportunities and challenges," *Information Fusion*, vol. 58, pp. 52–68, 2020.

[30] G. I. Parisi, R. Kemker, J. L. Part, C. Kanan, and S. Wermter, "Continual Lifelong Learning with Neural Networks: A review," *Neural Networks*, vol. 113, pp. 54–71, 2019.

[31] D. Shenaj, M. Toldo, A. Rigon, and P. Zanuttigh, "Asynchronous Federated Continual Learning," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pp. 5055–5063, 2023.

[32] Y. Chaudhary, P. Rai, M. Schubert, H. Schütze, and P. Gupta, "Federated Continual Learning for Text Classification via Selective Inter-client Transfer," in *Findings of the Association for Computational Linguistics: EMNLP 2022*, pp. 4789–4799, Association for Computational Linguistics, 2022.

[33] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, D. Hassabis, C. Clopath, D. Kumaran, and R. Hadsell, "Overcoming catastrophic forgetting in neural networks," *Proceedings of the National Academy of Sciences*, vol. 114, no. 13, pp. 3521–3526, 2017.

[34] J. Schwarz, W. Czarnecki, J. Luketina, A. Grabska-Barwinska, Y. W. Teh, R. Pascanu, and R. Hadsell, "Progress & Compress: A scalable framework for continual learning," in *International Conference on Machine Learning*, vol. 80, pp. 4528–4537, PMLR, 2018.

[35] F. Zenke, B. Poole, and S. Ganguli, "Continual Learning Through Synaptic Intelligence," in *International Conference on Machine Learning - Volume 70*, vol. 70 of *ICML'17*, pp. 3987–3995, JMLR.org, 2017.

[36] R. Aljundi, F. Babiloni, M. Elhoseiny, M. Rohrbach, and T. Tuytelaars, "Memory Aware Synapses: Learning What (not) to Forget," in *European Conference on Computer Vision (ECCV)*, pp. 139–154, 2018.

[37] Y.-C. Hsu, Y.-C. Liu, A. Ramasamy, and Z. Kira, "Re-evaluating Continual Learning Scenarios: A Categorization and Case for Strong Baselines," in *NeurIPS Continual learning Workshop*, 2018.

[38] H. Shin, J. K. Lee, J. Kim, and J. Kim, "Continual Learning with Deep Generative Replay," in *Advances in Neural Information Processing Systems 30*, pp. 2990–2999, Curran Associates, Inc., 2017.

[39] N. Carlini, J. Hayes, M. Nasr, M. Jagielski, V. Sehwag, F. Tramèr, B. Balle, D. Ippolito, and E. Wallace, "Extracting Training Data from Diffusion Models," in *32nd USENIX Security Symposium (USENIX Security 23)*, pp. 5253–5270, USENIX Association, 2023.

[40] E. Guerra, F. Wilhelmi, M. Miozzo, and P. Dini, "The Cost of Training Machine Learning Models Over Distributed Data Sources," *IEEE Open Journal of the Communications Society*, vol. 4, pp. 1111–1126, 2023.

[41] H. Shin, J. K. Lee, J. Kim, and J. Kim, "Continual Learning with Deep Generative Replay," in *Advances in Neural Information Processing Systems 30*, pp. 2990–2999, Curran Associates, Inc., 2017.

[42] L. Pellegrini, G. Graffieti, V. Lomonaco, and D. Maltoni, "Latent Replay for Real-Time Continual Learning," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 10203–10209, 2020.

[43] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4510–4520, 2018.

[44] J. Benesty, J. Chen, Y. Huang, and I. Cohen, "Pearson correlation coefficient," in *Noise reduction in speech processing*, pp. 37–40, Springer, 2009.