Probabilistic Fusion of Neural Networks that Incorporates Global Information

Peng Xiao[†] XIAOPENG24@GMAIL.COM
Department of Computer Science and Technology, Tongji University, Shanghai, China

Biao Zhang[†] ZHANGB20@FUDAN.EDU.CN
School of Mathematical Sciences, Fudan University, Shanghai, China

Samuel Cheng[*] SAMUEL.CHENG@OU.EDU
School of Electrical and Computer Engineering, University of Oklahoma, Oklahoma City, US

Ke Wei KEWEI@FUDAN.EDU.CN
School of Data Sciences, Fudan University, Shanghai, China

Shuqin Zhang ZHANGS@FUDAN.EDU.CN
School of Mathematical Sciences, Fudan University, Shanghai, China

Editors: Emtiyaz Khan and Mehmet Gönen

Abstract

As one of the approaches in Federated Learning, model fusion distills models trained on local clients into a global model. The previous method, Probabilistic Federated Neural Matching (PFNM), can match and fuse local neural networks with varying global model sizes and data heterogeneity using the Bayesian nonparametric framework. However, the alternating optimization process applied by PFNM causes absence of global neuron information. In this paper, we propose a new method that extends PFNM by introducing a Kullback-Leibler (KL) divergence penalty, so that it can exploit information in both local and global neurons. We show theoretically that the extended PFNM with a penalty derived from KL divergence can fix the drawback of PFNM by making a balance between Euclidean distance and the prior probability of neurons. Experiments on deep fully-connected as well as deep convolutional neural networks demonstrate that our new method outperforms popular state-of-the-art federated learning methods in both image classification and semantic segmentation tasks.

Keywords: Federated learning;Model fusion;Neural networks.

1. Introduction

The rapid development of artificial intelligence (especially the subdomain, machine learning) in the last decade is mainly due to the availability of huge quantities of data. However, the growing privacy concerns are an obstacle in the accessibility to some datasets in several applications, especially in medical and financial fields. Hence, federated learning (FL), in which a model is learned from siloed data, is proposed to tackle this challenge.

Federated learning provides a learning paradigm where local trained models are fused into a shared global model. The initial aggregation method in FL is FedAvg (McMahan et al., 2017) in

†. These authors contributed equally to this work
∗. Corresponding author

which parameters of local models are averaged coordinate-wisely. FedAvg was shown to be not optimal (Karimireddy et al., 2020; Deng et al., 2021; Xiao et al., 2020) and the performance of FedAvg can also deteriorate significantly when data is non-i.i.d (Independent and Identically Distributed). Subsequent improved optimizing methods (Zhang et al., 2020; Li et al., 2018; Karimireddy et al., 2020; Mohri et al., 2019; Deng et al., 2021; Smith et al., 2017) in FL are either constrained by the Lipschitz continuous assumption (Li et al., 2018; Karimireddy et al., 2020; Deng et al., 2021) or not generalized to non-convex problem (Mohri et al., 2019; Zhang et al., 2020; Smith et al., 2017), which means they are not suitable for neural network (NN) applications. Different from those methods, *model fusion*, the problem of learning a unified global model from a collection of pre-trained local models, provides an alternative approach to FL. Model fusion based approaches are widely used in FL applications currently (Yurochkin et al., 2019b; Wang et al., 2020; Singh and Jaggi, 2020; Claici et al., 2020; Lin et al., 2020; Zhu et al., 2021; Chen and Chao, 2020). These works can further be broadly divided into two categories. One category is *knowledge distillation* (Hinton et al., 2015; Buciluǎ et al., 2006; Schmidhuber, 1992), where the key idea is to employ the knowledge of pre-trained teacher neural networks (local models) to learn a student neural network (global model) (Lin et al., 2020; Chen and Chao, 2020; Zhu et al., 2021). The main disadvantage of distillation is its high computational complexity: in a global model, the distillation process requires collecting extra datasets or generating data with a Generative Adversarial Network (GAN), which is essentially as expensive as training a local model from scratch. Another category is *parameter matching*, where the key idea is matching the parameters with inherent permutation invariance from different local models before aggregating them together. In Singh and Jaggi (2020), the authors utilize optimal transport to minimize a transportation cost matrix to align neurons across different NNs, and it fixes the number of neurons in global model which make it infeasible when data is highly heterogeneous across clients. Some work (Claici et al., 2020) optimizes the assignments between global and local components under a KL divergence through variational inference. But the optimization process is not straightforward, and the calculation of variational inference is complicated.

Probabilistic Federated Neural Matching (PFNM) (Yurochkin et al., 2019b) is one of the representative parameters matching methods in FL. It develops a Bayesian nonparametric model to match and combine NNs across data sources. By applying Bayesian framework, PFNM can easily accommodate unobserved global model in the presence of local modelsDunson (2001). PFNM extends well in modern architectures such as convolutional neural networks (CNNs) and long short-term memory (LSTMs) (Wang et al., 2020), and its variants are also utilized in aggregating various statistical models such as Gaussian topic models, hierarchical Dirichlet process based hidden Markov models (Yurochkin et al., 2019a, 2018), etc. PFNM treats the local neurons, i.e., neurons of local NNs, as noisy realizations of latent global neurons, i.e., neurons of the fused global NN, and formally characterize the generative process through Beta-Bernoulli process (BBP) (Thibaux and Jordan, 2007). The matching is then governed by maximum a posterior (MAP) of the BBP via alternating optimization. In each step, PFNM allows neurons of current local NN to either match existing global neurons or create new global neurons if existing matches are poor. Due to the complexity of the objective function, alternative optimization generally can not achieve a global optima in PFNM (Beck, 2017). Specifically, for one local neuron, PFNM will select a global neuron which has smaller Euclidean distance to this local neuron. However, this ignores the differences in prior probabilities among global neurons. Therefore, PFNM may not pick those global neurons with higher prior probability and probably acquires a biased solution.

Kullback-Leibler (KL) divergence between the distribution of the global and local neural weights is a natural measure of model fusion performance. Thus in this paper we extend the MAP inference in PFNM through an additional penalty derived via KL divergence that incorporates global neuron information under BBP. In summary, our contributions include:

- **Theory**: We theoretically analyze that PFNM only uses Euclidean distance to measure the importance of neurons and ignores the prior probability of neurons in the fused model, thus leading to a biased solution. We further prove that KL divergence can fix the drawback of PFNM by making a balance between Euclidean distance and the prior probability of neurons. The theoretical results we demonstrate about KL may have implications in other fields as well.
- **Algorithms**: We propose a novel algorithm that substantially improves the performance of the fused model. Compared to other typical FL methods, it requires significantly fewer neurons, which indicates that not every neuron in the neural network can play a positive role. This further justifies the rationality of model fusion. Moreover, our method is robust to its hyperparameter.
- **Experiments**: Through a series of experiments over different client amounts, network architecture and tasks, we demonstrate the superiority of our method. In addition, to the best of our knowledge, it is the first work to extend the parameter matching method to batch normalization layer and then apply it in the U-net architecture.

2. Preliminaries and Problem Formulation

This section introduces some notations of federated learning in neural networks and the permutation invariance property of NN architectures. Then, we formulate the matched aggregating problem of local neurons.

Preliminaries For simplicity and without loss of generality, we suppose there are S fully connected (FC) NNs with one hidden layer trained from different datasets: $f_s(\boldsymbol{x}) = \sigma(\boldsymbol{x}^{\mathrm{T}}\boldsymbol{W}_s^{(0)})\boldsymbol{W}_s^{(1)}$,for $s = 1,\cdots,S$ (biases are omitted to simplify notation), where $\sigma(\cdot)$ is the nonlinear activation function, $\boldsymbol{W}_s^{(0)} \in \mathbb{R}^{D\times J_s}$ and $\boldsymbol{W}_s^{(1)} \in \mathbb{R}^{J_s\times K}$ are the weights; with D being the input dimension, J_s being the number of neurons on the hidden layer of sth NN, and K being the output dimension (i.e., number of classes). In federated learning, given the collection of weights $\{\boldsymbol{W}_s^{(0)}, \boldsymbol{W}_s^{(1)}\}_{s=1}^{S}$, we want to learn a global neural network with weights $\Theta^{(0)} \in \mathbb{R}^{D\times J}, \Theta^{(1)} \in \mathbb{R}^{J\times K}$, where $J \ll \sum_{s=1}^{S} J_s$ is an inferred variable denoting the number of hidden neurons of the global network.

Permutation invariance of one-layer FCNNs Expanding the preceding expression of a FCNN: $f_s(\boldsymbol{x}) = \sum_{j=1}^{J_s} \boldsymbol{W}_{s,j\cdot}^{(1)}\sigma(\langle \boldsymbol{x}, \boldsymbol{W}_{s,\cdot j}^{(0)}\rangle)$, where $j\cdot$ and $\cdot j$ denote the jth row and column correspondingly. Summation is a permutation invariant operation, thus for any $\{\boldsymbol{W}_s^{(0)}, \boldsymbol{W}_s^{(1)}\}$ there are $J_s!$ practically equivalent parametrizations, which can be expressed as

$$f_s(\boldsymbol{x}) = \sigma(\boldsymbol{x}^{\mathrm{T}}\boldsymbol{W}_s^{(0)}\mathbb{A}_s)\mathbb{A}_s^T\boldsymbol{W}_s^{(1)}, \tag{1}$$

where $\mathbb{A}_s$ is any $J_s \times J_s$ permutation matrix. Supposing $\{\boldsymbol{W}_s^{(0)}, \boldsymbol{W}_s^{(1)}\}$ are optimal weights, then weights acquired from two datasets X_{s_1}, X_{s_2} are $\{\boldsymbol{W}_s^{(0)}\mathbb{A}_{s_1}, \mathbb{A}_{s_1}^T\boldsymbol{W}_s^{(1)}\}$ and $\{\boldsymbol{W}_s^{(0)}\mathbb{A}_{s_2}, \mathbb{A}_{s_2}^T\boldsymbol{W}_s^{(1)}\}$. With a high probability $\mathbb{A}_{s_1} \neq \mathbb{A}_{s_2}$ and $(\boldsymbol{W}_s^{(0)}\mathbb{A}_{s_1} + \boldsymbol{W}_s^{(0)}\mathbb{A}_{s_2})/2 \neq \boldsymbol{W}_s^{(0)}\mathbb{A}_{s_1}$ for any permutation matrix. Thus to meaningfully aggregate neural networks in the weight space we should first align the weights $(\boldsymbol{W}_s^{(0)}\mathbb{A}_{s_1}\mathbb{A}_{s_1}^T + \boldsymbol{W}_s^{(0)}\mathbb{A}_{s_2}\mathbb{A}_{s_2}^T)/2 = \boldsymbol{W}_s^{(0)}$. Permutation invariance of deep neural architectures demonstrated in Supplement Sect. 1.

Matched Aggregation Formulation Here we present federated local neurons aggregation in one hidden layer NN under the permutation invariance. Let $\boldsymbol{w}_{sj}$ be the jth neuron learned on dataset s, $\boldsymbol{\theta}_i$ denote the ith neuron in the global model ($\boldsymbol{w}_{sj}, \boldsymbol{\theta}_i \in \mathbb{R}^{D+K}$ can be viewed as a concatenated vector of weight matrices). The solution of the following optimization problem is the required matching assignments:

$$\min_{\{\boldsymbol{\theta}_i\},\{\boldsymbol{A}^s\}} \sum_{s=1}^{S}\sum_{i=1}^{J}\sum_{j=1}^{J_s} \boldsymbol{A}_{i,j}^s \boldsymbol{C}_{i,j}^s$$
$$s.t. \sum_i \boldsymbol{A}_{i,j}^s = 1; \sum_j \boldsymbol{A}_{i,j}^s \leq 1; \boldsymbol{A}_{i,j}^s \in \{0,1\}, \tag{2}$$

where $\boldsymbol{C}_{i,j}^s = c(\boldsymbol{\theta}_i, \boldsymbol{w}_{sj})$ is an appropriate cost specification of assignment between local neuron $\boldsymbol{w}_{sj}$ and global neuron θ_i. And $\{\boldsymbol{A}^s\}_{s=1}^S$ is the collection of assignment variables of each dataset s, $\mathbb{A}_{s,i,j}^T = \boldsymbol{A}_{i,j}^s$, where $\boldsymbol{A}_{i,j}^s = 1$ implies the local neuron $\boldsymbol{w}_{sj}$ matched with global neuron $\boldsymbol{\theta}_i$ and vice versa. After assignments are inferred, the global model $\{\boldsymbol{\theta}_i = \arg\min_{\boldsymbol{\theta}_i} \sum_{s,j} \boldsymbol{A}_{i,j}^s \boldsymbol{C}_{i,j}^s\}_{i=1}^J$ can typically be solved in a closed form. The equality constraint implies that neurons of one client are a subset of aggregated global neurons. The inequality constraint indicates that neurons across clients may overlap only partially because of data heterogeneity in Federated Learning. Thus the size of constructed global model J satisfying $\max_s J_s \leq J \leq \sum_s J_s$. The overall matched aggregation procedure is illustrated in Fig. 1.

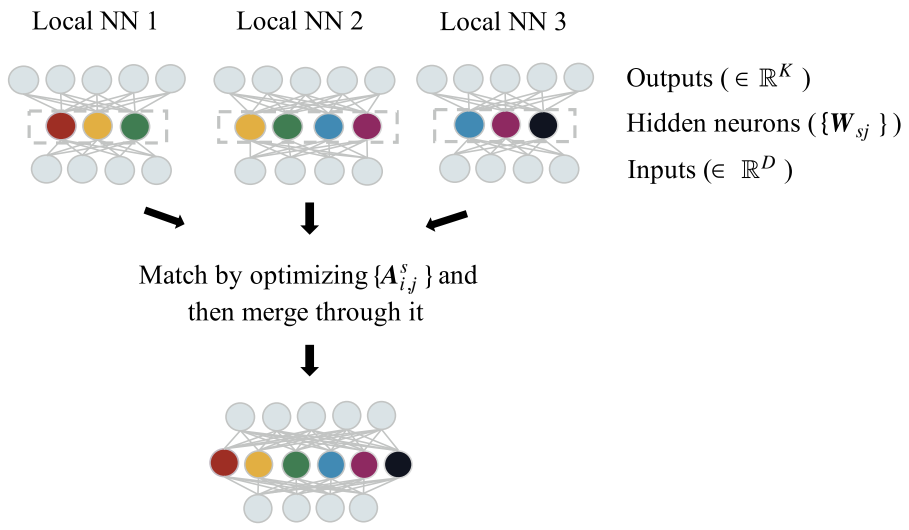
Local NN 1
Local NN 2
Local NN 3
Outputs ($\in \mathbb{R}^K$)
Hidden neurons ($\{\boldsymbol{W}_{sj}\}$)
Inputs ($\in \mathbb{R}^D$)
Match by optimizing $\{\boldsymbol{A}_{i,j}^s\}$ and then merge through it

Figure 1: The matching progress of three local NNs via Federated Neural matched aggregation. Colored nodes here indicate hidden neurons; the same colored hidden neurons have been matched.

3. Probabilistic Federated Neural Matching

In this section, we show how to deduce the cost specifications in Eq. (2). Specifically, PFNM models the generative process of neurons through BBP and acquires the cost specifications via maximizing a posterior estimation of the model. We firstly introduce some basic mathematical tools required by PFNM and our framework. Then we briefly review the PFNM model (Yurochkin et al., 2019b).

PFNM models the generative process of observed local neurons via a Beta-Bernoulli process (Yurochkin et al., 2019b) which described in Supplement Sect. 2. In PFNM, global atoms (hidden layer neurons) are drawn from a Beta process prior with a base measure H indicating the distribution of neurons and mass parameter γ_0, $M = \sum_i m_i \delta_{\boldsymbol{\theta}_i}$. Base measure H is chosen as multivariate Gaussian distribution $H = \mathcal{N}(\boldsymbol{\mu}_0, \boldsymbol{\Sigma_0})$ with $\boldsymbol{\mu}_0 \in \mathbb{R}^{D+K}$ and diagonal $\boldsymbol{\Sigma_0}$.

Local atoms are observed as noisy measurements of subsets of global atoms $\boldsymbol{w}_{sj} \mid \mathcal{T}_s \sim \mathcal{N}(\mathcal{T}_{sj}, \boldsymbol{\Sigma_s})$ for $s = 1, \cdots, S$; $j = 1, \cdots, J_s$; $J_s :=$ cardinality$(\mathcal{T}_s)$. The subset $\mathcal{T}_s$ is selected via the Bernoulli process: $\mathcal{T}_s := \sum_i a_{si}\delta_{\boldsymbol{\theta}_i}$ where $a_{si}|m_i \sim \text{Bern}(m_i)$ for any i. $\mathcal{T}_s$ is supported by atoms $\{\boldsymbol{\theta}_i : a_{si} = 1, i = 1, 2, \cdots\}$ and represent the identities of the atoms (neurons) used by subset s. Under this model, there is a one-to-one correspondence between $\{a_{si}\}_{i=1}^{\infty}$ and assignment variables $\{\boldsymbol{A}^s\}_{s=1}^{S}$ to be inferred, where $\boldsymbol{A}^s_{i,j} = 1$ implies $\mathcal{T}_{sj} = \boldsymbol{\theta}_i$.

To formulate the objective function in Eq. (2), PFNM maximizes a posterior estimation of global neurons for the above model:

$$\max_{\{\boldsymbol{\theta}_i\},\{\boldsymbol{A}^s\}} P(\{\boldsymbol{\theta}_i\}, \{\boldsymbol{A}^s\}|\{\boldsymbol{w}_{sj}\}) \propto P(\{\boldsymbol{w}_{sj}\}|\{\boldsymbol{\theta}_i\}, \{\boldsymbol{A}^s\})P(\{\boldsymbol{A}^s\})P(\{\boldsymbol{\theta}_i\}), \tag{3}$$

and by taking negative natural logarithm it can obtain:

$$\min_{\{\boldsymbol{\theta}_i\},\{\boldsymbol{A}^s\}} -\sum_i \left(\sum_{s,j} \boldsymbol{A}^s_{i,j} \log(p(\boldsymbol{w}_{sj}|\boldsymbol{\theta}_i)) + \log(q(\boldsymbol{\theta}_i)) \right) - \log(P(\{\boldsymbol{A}^s\})), \tag{4}$$

where $p(\boldsymbol{w}_{sj}|\boldsymbol{\theta}_i)$ denotes the probability density function of $\boldsymbol{w}_{sj}$ parameterized by mean $\boldsymbol{\theta}_i$, $q(\boldsymbol{\theta}_i)$ denotes the probability density function of $\boldsymbol{\theta}_i$, $P(\{\boldsymbol{A}^s\})$ is interpreted by IBP and demonstrated in Supplement Sect. 2. Given $\{\boldsymbol{A}^s\}_{s=1}^{S}$, the closed form of $\{\boldsymbol{\theta}_i\}$ can be estimated according to the Gaussian-Gaussian conjugacy:

$$\hat{\boldsymbol{\theta}}_i = \frac{\boldsymbol{\mu}_0/\sigma_0^2 + \sum_{s,j} \boldsymbol{A}^s_{i,j} \boldsymbol{w}_{sj}/\sigma_s^2}{1/\sigma_0^2 + \sum_{s,j} \boldsymbol{A}^s_{i,j}/\sigma_s^2} \text{ for } i = 1, \cdots, J, \tag{5}$$

where for simplicity we assume $\boldsymbol{\Sigma}_0 = \boldsymbol{I}\sigma_0^2$ and $\boldsymbol{\Sigma}_s = \boldsymbol{I}\sigma_s^2$.

PFNM solves the assignments and constructs the global model in an iterative approach. Let $-s'$ be "all but s'", fixing assignments $\{\boldsymbol{A}^s_{i,j}\}_{i,j,s\in-s'}$, it can find the optimal assignments $\{\boldsymbol{A}^{s'}_{i,j}\}_{i,j}$ corresponding to dataset s' at current iteration via Hungarian algorithm (Kuhn, 1955), and then it iterates over remaining datasets. At each iteration, given current estimates of $\{\boldsymbol{A}^s_{i,j}\}_{i,j,s\in-s'}$, it find a corresponding matched aggregating global model $\{\boldsymbol{\theta}_i = \arg\min_{\boldsymbol{\theta}_i} \sum_{s\in-s',j} \boldsymbol{A}^s_{i,j}\boldsymbol{C}^s_{i,j}\}_{i=1}^{J_{-s'}}$ via the closed form expression, i.e. Eq. (5), where $J_{-s'} = \max\{i : \boldsymbol{A}^s_{ij} = 1, \text{ for } s \in -s', j = 1, \ldots, J_s\}$ denote the number of active global neurons outside of s'.

The following proposition describes the assignment cost $\{\boldsymbol{C}^{s'}_{i,j}\}_{i,j}$ obtained by PFNM corresponding in Eq. (2):

Proposition 1 *The assignment cost specification $\boldsymbol{C}^{s'}_{i,j}$ for finding $\{\boldsymbol{A}^{s'}\}$ corresponding to objective Eq. (4) is*

$$\begin{cases} \dfrac{\|\tilde{\boldsymbol{\theta}}_i - \boldsymbol{w}_{s'j}\|_2^2 - L_1\|\boldsymbol{w}_{s'j}\|_2^2}{L_2/L_3} - 2\log\dfrac{n_i^{-s'}}{S-n_i^{-s'}}, & i \le J_{-s'}, \\ \dfrac{\|\boldsymbol{\mu}_0 - \boldsymbol{w}_{s'j}\|_2^2 - L_4\|\boldsymbol{w}_{s'j}\|_2^2}{\sigma_0^2+\sigma_{s'}^2} + 2\log\dfrac{i-J_{-s'}}{\gamma_0/S}, & J_{-s'} < i \le J_{-s'} + J_{s'}, \end{cases} \tag{6}$$

where the norm is l_2-norm, L_1, L_2, L_3, L_4 are constants whose expressions are given in the Supplement Sect. 5.1, and $\tilde{\boldsymbol{\theta}}_i = \frac{\boldsymbol{\mu}_0/\sigma_0^2 + \sum_{s\in-s',j}\boldsymbol{A}^s_{i,j}\boldsymbol{w}_{sj}/\sigma_s^2}{1/\sigma_0^2+\sum_{s\in-s',j}\boldsymbol{A}^s_{i,j}/\sigma_s^2}$ denotes the neuron in the estimated global model at current iteration, $n_i^{-s'} = \sum_{s\in-s',j}\boldsymbol{A}^s_{i,j}$ denotes the times that client neurons were assigned to global neuron i outside of s'.

The proof can be found in Supplement Sect. 5.1. Thus PFNM can be summarized as a alternative optimization process in Algorithm 1.

Algorithm 1 Alternating optimization in PFNM

```
Input:
    Local weights $\boldsymbol{w}_{sj}$ from $S$ clients;
Output:
    Global weights $\{\boldsymbol{\theta}_i\}$, matching assignments $\{\boldsymbol{A}^s\}_{s=1}^S$;
1: for $k = 1, 2, 3, \cdots$ do
2:     for $s' = 1, 2, \cdots, S$ do
3:         Fixing assignments $\{\boldsymbol{A}^s_{i,j}\}_{i,j,s\in-s'}$, construct corresponding global model via Eq. (5): $\tilde{\boldsymbol{\theta}}_i = \frac{\boldsymbol{\mu}_0/\sigma_0^2+\sum_{s\in-s',j}\boldsymbol{A}^s_{i,j}\boldsymbol{w}_{sj}/\sigma_s^2}{1/\sigma_0^2+\sum_{s\in-s',j}\boldsymbol{A}^s_{i,j}/\sigma_s^2}$;
4:         Obtain the assignment cost $\{\boldsymbol{C}^{s'}_{i,j}\}_{i,j}$ via Eq. (6);
5:         Solve the linear assignment problem via Hungarian algorithm to obtain $\{\boldsymbol{A}^{s'}_{i,j}\}_{i,j}$.
6:     end for
7:     Use $\{\boldsymbol{A}^s\}_{s=1}^S$ to update the global model.
8: end for
```

4. Extended Probabilistic Federated Neural Matching

In this section, we firstly point out that the alternating optimization method used in PFNM brings bias of solution, and can not optimize Eq. (4) globally in a supposed way. After that, to mitigate the bias, we introduce Kullback-Leibler (KL) divergence between the distribution of global and local neural components.

4.1. Analysis of PFNM

As we point out above, PFNM applies alternative optimizing to solve the assignments between global neurons and local neurons. However, alternative optimization generally can not achieve a global optima in PFNM (Beck, 2017). For simplicity, we only consider the prior probability of global neurons in Eq. (4), that is

$$\min_{\{\boldsymbol{A}^s\}} -\log q(\hat{\boldsymbol{\theta}}_i) \tag{7}$$

$$
\begin{aligned}
&= \min_{\{\boldsymbol{A}^s\}} \|\hat{\boldsymbol{\theta}}_i - \mu_0\|_2^2 + C \\
&= \min_{\{\boldsymbol{A}^s\}} \left\| \frac{\sum_{s,j} A_{ij}^s (\boldsymbol{w}_{sj} - \boldsymbol{\mu}_0)}{\sigma_s^2/\sigma_0^2 + \sum_{s,j} A_{ij}^s} \right\|_2^2 + C \\
&= \min_{\{\boldsymbol{A}^s\}} \left\| \frac{f_1(\boldsymbol{A}^{s'}) + f_2(\boldsymbol{A}^{-s'})}{g_1(\boldsymbol{A}^{s'}) + g_2(\boldsymbol{A}^{-s'})} \right\|_2^2 + C.
\end{aligned} \tag{8}
$$

where C is a constant and f_1,f_2,g_1 and g_2 are real functions. However, in the alternating minimization step of PFNM, the problem becomes

$$
\min_{\{\boldsymbol{A}^{s'}\}} \left\| \frac{f_1(\boldsymbol{A}^{s'}) + C_1}{g_1(\boldsymbol{A}^{s'}) + C_2} \right\|_2^2 + C, \tag{9}
$$

where $C_1 = f_2(A^{-s'})$, $C_2 = g_2(A^{-s'})$ are constants independent of $A^{s'}$. Obviously, $A^{s'}$ solved from Eq. (9) may be quite different from the original problem of Eq. (7). Therefore, in each iterative step the optimism of the assignments is not guaranteed.

Considering the above general analysis, as shown in Eq. (6), for model s', to minimize the cost function essentially minimizes the l_2-distance between local neuron $\boldsymbol{w}_{s'j}$ and estimated global neuron $\tilde{\boldsymbol{\theta}}_i$, and selects local neuron $\boldsymbol{w}_{s'j}$ which has smaller l_2-norm. Recall that PFNM assumes neurons obey Gaussian distribution whose covariance matrix is simply $\boldsymbol{\Sigma}_0 = \boldsymbol{I}\sigma_0^2$ for global neurons and $\boldsymbol{\Sigma}_{s'} = \boldsymbol{I}\sigma_{s'}^2$ for local neurons in model s'. For local neurons $\boldsymbol{w}_{s'j}$ assigned to global neuron $\boldsymbol{\theta}_i$, the probability density function of $\boldsymbol{w}_{s'j}$ is

$$
p_{\boldsymbol{\theta}_i}(\boldsymbol{w}_{s'j}) = \frac{1}{\sqrt{(2\pi)^{D+K}|\boldsymbol{\Sigma}_{s'}|}} e^{-\frac{1}{2}\frac{\|\boldsymbol{w}_{s'j}-\boldsymbol{\theta}_i\|_2^2}{\sigma_{s'}^2}},
$$

where $D+K$ is the dimension of neurons. Therefore, when $i \le J_{-s'}$, for fixed global neuron $\boldsymbol{\theta}_i$, to minimize the first part of the cost function related to $\|\tilde{\boldsymbol{\theta}}_i - \boldsymbol{w}_{s'j}\|_2^2$ is essentially picking a local neuron with higher prior probability. When two local neurons $\boldsymbol{w}_{s'j}$ and $\boldsymbol{w}_{s'j^*}$ have almost equal prior probability, i.e., $\|\tilde{\boldsymbol{\theta}}_i - \boldsymbol{w}_{s'j}\|_2^2 = \|\tilde{\boldsymbol{\theta}}_i - \boldsymbol{w}_{s'j^*}\|_2^2$, to minimize the cost function means to pick a local neuron with higher l_2-norm. When $J_{-s'} < i \le J_{-s'} + J_{s'}$, as seen in Eq. (6), since we do not yet have an estimate of global neuron $\boldsymbol{\theta}_i$, we just use the mean of global neurons as a rough estimation of global neuron. The situation is similar for the case when $i \le J_{-s'}$. This is in correspondence with the maximization of posterior probability in PFNM model. However, we will see that the cost function does not optimize well on prior probability of global neurons.

For global neurons, the prior density function of $\boldsymbol{\theta}_i$ is

$$
q(\boldsymbol{\theta}_i) = \frac{1}{\sqrt{(2\pi)^n|\boldsymbol{\Sigma}_0|}} e^{-\frac{1}{2}\frac{\|\boldsymbol{\theta}_i-\boldsymbol{\mu}_0\|_2^2}{\sigma_0^2}}. \tag{10}
$$

As we have seen in Eq. (6), there is no term related to $\|\boldsymbol{\theta}_i - \boldsymbol{\mu}_0\|_2^2$, or to be specific, prior probability of $q(\boldsymbol{\theta}_i)$. Therefore, for fixed local neuron $\boldsymbol{w}_{s'j}$, let us assume that $\boldsymbol{w}_{s'j}$ have almost equal probability to be assigned to global neuron $\boldsymbol{\theta}_i$ and $\boldsymbol{\theta}_{i^*}$, that is, $\|\tilde{\boldsymbol{\theta}}_i - \boldsymbol{w}_{s'j}\|_2^2 = \|\tilde{\boldsymbol{\theta}}_{i^*} - \boldsymbol{w}_{s'j}\|_2^2$ where $\tilde{\boldsymbol{\theta}}_i$, $\tilde{\boldsymbol{\theta}}_{i^*}$ are estimations of global neurons $\boldsymbol{\theta}_i$ and $\boldsymbol{\theta}_{i^*}$, respectively. We also assume $n_i^{-s'} = n_{i^*}^{-s'}$ in the remaining part, since in the initial stage of iteration all $n_i^{-s'}$ are identical. In such case, the cost function has no discrimination on $\boldsymbol{\theta}_i$ and $\boldsymbol{\theta}_{i^*}$. However, based on the model assumptions, the cost function is supposed to pick a global neuron with higher prior probability between $\boldsymbol{\theta}_i$ and $\boldsymbol{\theta}_{i^*}$. We will show that a term derived through Kullback-Leibler divergence can fix this problem.

4.2. Kullback-Leibler Divergence Penalty

How to evaluate the performance of model fusion for neural networks without the knowledge of the datasets? A natural measurement is that, the distribution of local neurons should be close to the distribution of global neurons to which those local neurons are assigned to. Hence, we propose Kullback-Leibler (KL) divergence between joint probability distributions of local neurons and their corresponding global neurons, i.e.,

$$\min_{\{\boldsymbol{\theta}_i\},\{\boldsymbol{A}^s\}} \sum_s \mathrm{KL}\Big(\prod_j q\big(\sum_i \boldsymbol{A}^s_{i,j}\boldsymbol{\theta}_i\big)\Big\|\prod_j p_{\boldsymbol{\theta}_i}(\boldsymbol{w}_{sj})\Big), \tag{11}$$

and $q\big(\sum_i \boldsymbol{A}^s_{i,j}\boldsymbol{\theta}_i\big)$ represents the distribution of global neuron that matches to $\boldsymbol{w}_{sj}$. Decomposing over joint distributions in $\mathrm{KL}(\cdot\|\cdot)$ allows us to rewrite problem (11) as

$$\min_{\{\boldsymbol{\theta}_i\},\{\boldsymbol{A}^s\}} \sum_i \sum_{s,j} \boldsymbol{A}^s_{i,j}\,\mathrm{KL}\big(q\|p_{\boldsymbol{\theta}_i}\big). \tag{12}$$

As mentioned previously, for local neuron $\boldsymbol{w}_{s'j}$ which is assigned to global neuron $\boldsymbol{\theta}_i$, $\boldsymbol{w}_{s'j}$ obeys a Gaussian distribution whose mean is $\boldsymbol{\theta}_i$ and covariance matrix is $\boldsymbol{\Sigma}_{s'} = \sigma^2_{s'}\boldsymbol{I}$. For global neuron $\boldsymbol{\theta}_i$, it obeys Gaussian distribution with mean $\boldsymbol{\mu}_0$ and covariance matrix $\boldsymbol{\Sigma}_0 = \sigma_0^2\boldsymbol{I}$. Therefore, KL divergence between distribution of local neuron $\boldsymbol{w}_{s'j}$ and distribution of global neuron $\boldsymbol{\theta}_i$, i.e., $\mathrm{KL}^i_{s'j}[\boldsymbol{\theta}_i]$, can be expanded as

$$\begin{aligned}
&\mathrm{KL}^i_{s'j}[\boldsymbol{\theta}_i]\\
=&\mathrm{KL}(q\|p_{\boldsymbol{\theta}_i})\\
=&\int_{\boldsymbol{x}} q(\boldsymbol{x})\log\frac{q(\boldsymbol{x})}{p_{\boldsymbol{\theta}_i}(\boldsymbol{x})}\mathrm{dx}\\
=&\frac{1}{2}\big((D+K)(\frac{\sigma_0^2}{\sigma_s^2}-1+\log\frac{\sigma_s^2}{\sigma_0^2}+\frac{\|\boldsymbol{\theta}_i-\boldsymbol{\mu}_0\|_2^2}{\sigma_s^2}\big).
\end{aligned} \tag{13}$$

Since only the term $\|\boldsymbol{\theta}_i-\boldsymbol{\mu}_0\|_2^2/\sigma_0^2$ contains $\boldsymbol{\theta}_i$, to minimize $\mathrm{KL}^i_{s'j}[\boldsymbol{\theta}_i]$ on $\boldsymbol{\theta}_i$ is actually minimizing $\|\boldsymbol{\theta}_i-\boldsymbol{\mu}_0\|_2^2/\sigma_0^2$. As we see in Eq. (10), this is equal to maximizing the prior probability of global neuron $\boldsymbol{\theta}_i$.

For fixed local neuron $\boldsymbol{w}_{s'j}\in\mathbb{R}^n$ and two global neurons $\boldsymbol{\theta}_i\in\mathbb{R}^n$ and $\boldsymbol{\theta}_{i^*}\in\mathbb{R}^n$, we suppose $\boldsymbol{w}_{s'j}$ has almost equal probability to be assigned to global neuron $\boldsymbol{\theta}_i$ and $\boldsymbol{\theta}_{i^*}$, i.e.,

$$\|\tilde{\boldsymbol{\theta}}_i-\boldsymbol{w}_{s'j}\|_2 = \|\tilde{\boldsymbol{\theta}}_{i^*}-\boldsymbol{w}_{s'j}\|_2.$$

The difference of KL penalty between them,

$$\mathrm{KL}(q\|p_{\boldsymbol{\theta}_i})-\mathrm{KL}(q\|p_{\boldsymbol{\theta}_{i^*}})$$

is equivalent to $\|\boldsymbol{\theta}_i-\boldsymbol{\mu}_0\|_2^2-\|\boldsymbol{\theta}_{i^*}-\boldsymbol{\mu}_0\|_2^2$. It indicates that while the cost function in PFNM model can not discriminate between $\boldsymbol{\theta}_i$ and $\boldsymbol{\theta}_{i^*}$, to minimize KL divergence will essentially pick a global neuron with higher prior probability. This indeed fixes the problem of the cost function in PFNM model. Besides, when PFNM model can discriminate between $\boldsymbol{\theta}_i$ and $\boldsymbol{\theta}_{i^*}$, which means $\boldsymbol{\theta}_i$ and $\boldsymbol{\theta}_{i^*}$ have different l_2-distance to local neuron $\boldsymbol{w}_{s'j}$, to minimize KL divergence will still probably pick a global neural with higher prior probability. We also need some assumptions as listed in Assumption 2.

Assumption 2 *For one fixed local neuron $\boldsymbol{w}_{s'j} \in \mathbb{R}^n$ and two global neurons $\boldsymbol{\theta}_i \in \mathbb{R}^n$ and $\boldsymbol{\theta}_{i^*} \in \mathbb{R}^n$, we assume*

(A) $\|\boldsymbol{\theta}_i - \boldsymbol{w}_{s'j}\|_2 = \|\boldsymbol{\theta}_{i^*} - \boldsymbol{w}_{s'j}\|_2 + \epsilon$, where $\epsilon \geq 0$;
(B) $\|\boldsymbol{\theta}_i\|_2 = R_1$ is fixed;
(C) $\boldsymbol{\theta}_i$ and $\boldsymbol{\theta}_{i^*}$ are sampled from $N(\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0) = N(0, \sigma_0^2 \boldsymbol{I})$.

Based on Assumption 2, we have Proposition 3.

Proposition 3 *Under Assumption (2), let $P\Big(\mathrm{KL}_{s'j}^{i^*}[\boldsymbol{\theta}_{i^*}] \geq \mathrm{KL}_{s'j}^{i}[\boldsymbol{\theta}_i]\Big)$ be the probability that $\mathrm{KL}_{s'j}^{i^*}[\boldsymbol{\theta}_{i^*}]$ is greater than $\mathrm{KL}_{s'j}^{i}[\boldsymbol{\theta}_i]$, then*

$$\begin{cases} P\Big(\mathrm{KL}_{s'j}^{i^*}[\boldsymbol{\theta}_{i^*}] \geq \mathrm{KL}_{s'j}^{i}[\boldsymbol{\theta}_i]\Big) = 0, & \epsilon \geq 2\|\boldsymbol{w}_{s'j}\|_2, \\ P\Big(\mathrm{KL}_{s'j}^{i^*}[\boldsymbol{\theta}_{i^*}] \geq \mathrm{KL}_{s'j}^{i}[\boldsymbol{\theta}_i]\Big) \geq B_n(\epsilon), & \epsilon < 2\|\boldsymbol{w}_{s'j}\|_2, \end{cases} \tag{14}$$

where $B_n(\epsilon)$ is a monotonically decreasing non-negative function of ϵ.

The proof can be found in Supplement Sect. 5.2. When there are two global neurons that have different l_2-distances to a local neuron, minimizing the cost function of PFNM will directly pick the global neuron closer to the local neuron while the prior probabilities of global neurons are not used. As shown in the above proposition, ϵ is the difference between distances of global neurons to the same local neuron. As a lower bound, the monotonically decreasing non-negative function of ϵ implies that to minimize the KL divergence can make a balance between l_2-distance and prior probability, instead of only using distance to select global neurons. When the distance difference of two global neurons between one local neuron is small (which means it can hardly distinguish different global neurons via l_2-distance), the KL term has a higher probability of selecting a higher prior probability global neuron; and when the distance discrepancy of different global neurons is large, it will select global neurons closer to local neurons as the original PFNM does. Therefore, an extended PFNM model by adding a KL penalty theoretically performs better than the original PFNM method. In the remaining part, we prove Eq. (11) can also be derived as a linear sum assignment problem similar to the form of Eq. (6).

For the iterative optimization procedure, we have the following proposition where Eq. (11) also enjoys a linear sum assignment form like $\sum_i \sum_j \boldsymbol{A}_{i,j}^{s'} \widetilde{\boldsymbol{C}}_{i,j}^{s'}$:

Proposition 4 *The assignment cost specification for finding $A^{s'}$ corresponding to Eq. (11) is $\widetilde{\boldsymbol{C}}_{i,j}^{s'} =$*

$$\begin{cases} \dfrac{\left\| \frac{\boldsymbol{w}_{s'j} - \boldsymbol{\mu}_0}{\sigma_{s'}^3} + \sum\limits_{s \in -s', j} \boldsymbol{A}_{i,j}^{s} \frac{\boldsymbol{w}_{sj} - \boldsymbol{\mu}_0}{\sigma_s^3} \right\|^2}{\big(\frac{1}{\sigma_0^2} + \frac{1}{\sigma_{s'}^2} + \sum\limits_{s \in -s', j} \boldsymbol{A}_{i,j}^{s} / \sigma_s^2\big)^2} (n_i^{-s'} + 1) - \dfrac{\big\| \sum\limits_{s \in -s', j} \boldsymbol{A}_{i,j}^{s} \frac{\boldsymbol{w}_{sj}}{\sigma_s^3} \big\|^2}{\big(\frac{1}{\sigma_0^2} + \sum\limits_{s \in -s', j} \boldsymbol{A}_{i,j}^{s} / \sigma_s^2\big)^2} n_i^{-s'}, & i \leq J_{-s'}, \\ \dfrac{\big\| \frac{\boldsymbol{w}_{s'j} - \boldsymbol{\mu}_0}{\sigma_{s'}^3} \big\|^2}{\big(\frac{1}{\sigma_0^2} + \frac{1}{\sigma_{s'}^2}\big)^2}, & J_{-s'} < i \leq J_{-s'} + J_{s'}. \end{cases} \tag{15}$$

The proof can be found in the Supplement Sect. 5.3.

Reverse KL. How about using the reverse KL term $\mathrm{KL}\big(p_{\boldsymbol{\theta}_i} \| q\big)$ instead of $\mathrm{KL}\big(q \| p_{\boldsymbol{\theta}_i}\big)$ in euqation (12)? Although $\mathrm{KL}\big(p_{\boldsymbol{\theta}_i} \| q\big)$ is different from $\mathrm{KL}\big(q \| p_{\boldsymbol{\theta}_i}\big)$ in common sense, both of them have identical effect in the cost specification. According to equation (13) in our paper, only $\frac{\|\theta_i - \mu_0\|_2^2}{\sigma_s^2}$

has effect in the cost specification. And for $\mathrm{KL}\big(p_{\boldsymbol{\theta}_i} \| q\big)$, this term becomes $\frac{||\mu_0-\theta_i||_2^2}{\sigma_0^2}$, and the front term is $\frac{\sigma_s^2}{\sigma_0^2}$ scale to the behind term. As all σ_s are the same in our setting, the front term is proportional to the behind term. Thus they both have indentical effect in the cost specification of the linear sum assignment formulation.

As we discuss above, PFNM only utilizes l_2-distance to measure the importance of neurons and ignores prior probability of neurons in the fused model. KL divergence can fix the drawback of PFNM by making a balance between l_2-distance and the prior probability of neurons. Hence, we directly acquire a more reasonable model by treating Eq. (11) as a penalty to adding to Eq. (4), and thus we acquire a new model:

$$\min_{\{\boldsymbol{\theta}_i\},\{\boldsymbol{A}^s\}} -\sum_i \Big(\sum_{s,j} \boldsymbol{A}^s_{i,j} \log(p(\boldsymbol{w}_{sj}|\boldsymbol{\theta}_i)) + \log(q(\boldsymbol{\theta}_i))\Big) - \log(P(\{\boldsymbol{A}^s\})) + \lambda \sum_s \mathrm{KL}\Big(\prod_j q\big(\sum_i \boldsymbol{A}^s_{i,j}\boldsymbol{\theta}_i\big) \Big\| \prod_j p_{\boldsymbol{\theta}_i}(\boldsymbol{w}_{sj})\Big), \tag{16}$$

and the cost specifications of the new model is:

$$\mathbb{C} = \boldsymbol{C} + \lambda \widetilde{\boldsymbol{C}}, \tag{17}$$

which makes a balance between l_2-distance and prior probability instead of only using distance to select global neurons. We call this matched aggregation with global information (MAGI). Coefficient λ is the adjusting ratio.

As the study in Wang et al. (2020) demonstrates, directly applying the matching algorithms fails on deep architectures designed for more complex tasks. Thus to alleviate this problem, we also extend MAGI to layer-wise matching scheme which can be found in the Supplement Sect. 3. Besides, although some studies (Wang et al., 2020) shows how to apply the PFNM to CNNs, it doesn't enable batch normalization layer to the matching algorithm. However, widely used deep CNNs such as U-net often contain batch normalization layer in their architectures. In this paper, we utilize a common setup which merges the batch normalization layer with a preceding convolution to incorporate MAGI with batch normalization layer. This is also included in Supplement Sect. 4.

Algorithm 2 MAGI

```
Input:
    Local weights $\boldsymbol{w}_{sj}$ from $S$ clients;
Output:
    Global weights $\{\boldsymbol{\theta}_i\}$, matching assignments $\{\boldsymbol{A}^s\}_{s=1}^S$;
1: Form each assignment cost matrix via Eq. (17);
2: Use Hungarian algorithm to compute $\{\boldsymbol{A}^s\}_{s=1}^S$;
3: List all resulting distinct global neurons and then apply Eq. (5) to infer associated global
   weights from all instances of global neurons across $S$ datasets;
4: Aggregate the global neurons to construct the new global model.
```

5. Experiments

This section presents an empirical study of MAGI and compares it with PFNM, FedAvg (McMahan et al., 2017), and FedProx (Li et al., 2018). Our experiments are conducted over three different datasets with various neural networks, that is, FCNN, shallow CNN and U-net. And the experiments below show that our framework can aggregate multiple NNs into an efficient global one. We also present how the number of neurons change along with, λ, the weight of the KL divergence term. The source code is publicly available on Github: `https://github.com/moon24x/MAGI`.

Datasets, models and metrics We evaluate our algorithm on three datasets: MINST, CIFAR 10 and Carvana Image Masking Challenge (CIMC). The first two are standard image classification datasets and each contains ten classes on handwriting digits and objects in real life respectively. The third one, CIMC, is a binary semantic segmentation dataset composed of photos of cars, and the task is to split out the car and the background. For MNIST, we apply a FCNN model and evaluate it with accuracy; for CIFAR 10, we apply a ConvNet with with 3 convolutional and 2 fully-connected layer and evaluate it with accuracy; for CIMC, we apply the U-net (Ronneberger et al., 2015) architecture and evaluate it with dice coefficient.

Partition strategies of client data Here we consider a heterogeneous partition strategy to simulate a federated learning scenario where the number of data points and class proportions in each client is unbalanced. In heterogeneous partition of client data, for three datasets, we follow prior works (Yurochkin et al., 2018) which apply K-dimensional Dirichlet distribution $Dir(\alpha)$ to create non-iid data, in which a smaller α indicates higher data heterogeneity. Specifically, for dataset with class number K, we sample the proportion of the instances of class k to client s, $p_{k,s}$, via $p_{k,s} \sim Dir_k(\alpha)$. For MNIST and CIFAR10, $K = 10$. For CIMC, $K = 1$ as it is a binary semantic segmentation dataset. In each dataset, we execute 5 trials to obtain mean and standard deviation of the performances.

Table 1: Performance overview on different tasks, S denotes the number of local models, N denotes the number of layers in a neural network

Datasets (Architectures)	S	N	Local NN	FedAvg	FedProx	PFNM	MAGI
MNIST (FCNN)	15	1	71.9 ± 2.57	75.47 ± 5.90	75.65 ± 5.93	83.93 ± 0.14	**86.25** ± **0.88**
	20	1	69.44 ± 2.50	75.02 ± 4.03	75.17 ± 3.99	83.23 ± 3.31	**85.64** ± **3.09**
	25	1	67.99 ± 2.00	75.46 ± 3.33	75.24 ± 3.30	84.87 ± 1.66	**86.95** ± **3.21**
	30	1	65.90 ± 2.66	73.43 ± 4.49	73.11 ± 4.44	83.39 ± 2.23	**86.68** ± **2.64**
	10	2	73.21 ± 3.05	66.56 ± 6.82	66.34 ± 6.75	79.09 ± 4.73	**82.77** ± **4.17**
	10	3	70.28 ± 2.97	52.01 ± 6.52	51.79 ± 6.49	60.71 ± 4.59	**70.66** ± **4.83**
CIFAR10 (ConvNet)	5	5	25.21 ± 2.30	51.26 ± 2.97	51.43 ± 3.01	50.39 ± 0.94	**51.83** ± **0.81**
	10	5	18.92 ± 1.41	46.78 ± 2.36	46.94 ± 2.32	46.27 ± 1.73	**48.32** ± **1.32**
	15	5	15.85 ± 0.51	42.40 ± 2.25	42.06 ± 2.20	42.82 ± 2.46	**45.34** ± **1.78**
	20	5	14.11 ± 0.53	34.20 ± 2.54	34.02 ± 2.52	42.61 ± 2.07	**44.44** ± **1.85**
CIMC (U-net)	8	19	67.60 ± 9.38	53.28 ± 10.63	41.53 ± 0.82	90.31 ± 2.90	**96.01** ± **0.84**
	16	19	27.84 ± 12.16	42.62 ± 0.87	48.50 ± 9.52	75.47 ± 3.54	**82.57** ± **2.91**

Baselines We compare our method with original PFNM, FedAvg, and FedProx. Here the FedAvg and FedProx are operated in local neural networks trained with the same random initialization as proposed by McMahan et al. (2017). We note that a federated averaging variant without the shared initialization would likely be more realistic when trying to aggregate pre-trained models, but it performs significantly worse than all other baselines.
Training setup We use PyTorch (Paszke et al., 2017) to implement these networks and train them by the Adam (Kingma and Ba, 2014), SGD (Bottou, 2010) and RMSprop (Hinton et al., 2012) with default hyperparameters. All hyperparameter settings are summarized in Supplement Sect. 6.1.
Performance Overview For applications of Federated Learning in real world, the discrepancy of

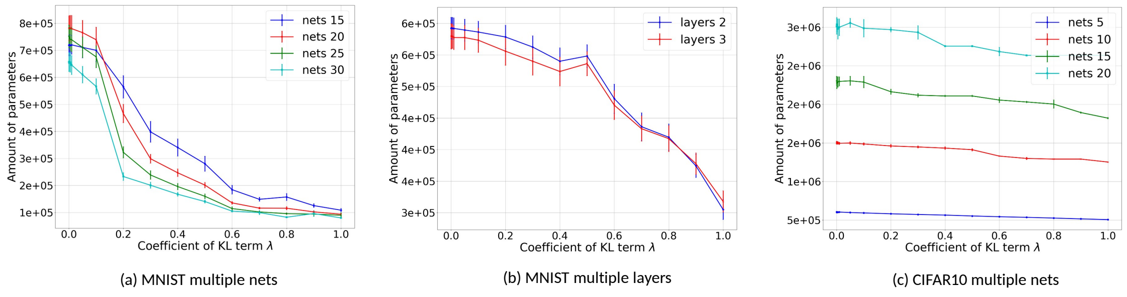

Figure 2: Amount of parameters varies with the KL regularization coefficients.

data distribution among clients and communication cost will grow up naturally as the number of clients increases due to the variability of data generation paradigms in the system (Li et al., 2020). Consequently, the model fusion problem is harder under configuration of more clients. This implies that it is meaningful to test the neural networks fusing algorithms for various number of clients. For MNIST, we firstly apply various methods such as FedAVG, FedProx and PFNM, with 15, 20, 25 and 30 local clients with one hidden layer FCNN. Local NN in Table 1 reports the average of separately tested network accuracies. The performance of local NNs define the lower extremes of aggregating. As the number of clients S increases, the performance of each method decreases. Because in our partition setting, each client contains fewer training data and probably fewer labels as S increases. As shown in Table 1, accuracy of MAGI on MNIST is 3% higher than PFNM on average. We also test how number of hidden layers N affect the performance of model fusion methods. We train 10 neural networks with 2 and 3 hidden layers respectively and then use various methods to fuse them. Similarly, MAGI achieves the best performance among all the methods. In addition, the priority of MAGI increases as deepness of the neural networks increases. This can be explained by an accumulating error effect. On the one hand, in Sect. 3 and Sect. 4, we analyzed the problem of PFNM and concluded that PFNM is unable to discriminate those global neurons who have the same l_2-distance to a local neuron, and MAGI furnished with KL divergence can fix this. On the other hand, the fusing process of neural networks is actually going layer-by-layer, from the first layer to the last layer. For each layer, when fusing local models, the drawback of PFNM we described will take into effect and global neurons are not generated correctly. Therefore, for the whole local models, as the iteration process directed by Hungarian algorithm goes on, the incorrectness of PFNM on picking global neurons is superimposed and the performance discrepancy between PFNM and MAGI is amplified as presented in Table 1.

In real-world applications of deep learning, CNNs are far more widely used than FCNNs. Consequently for this, we set ConvNet (2 convolutional layers and 3 fully connected layers) on

5, 10, 15 and 20 local clients and apply various methods to fuse the models trained in heterogeneous dataset partition of CIFAR 10. For PFNM and MAGI, we apply them in iteratively layer-wise way, thus we also set the communications rounds of FedAvg and FedProx to 5 equal to the number of layers in ConvNet for equality. As shown in Table 1, MAGI outperforms the other methods on fusing convolutional neural networks. Concretely, accuracy of MAGI on CIFAR 10 is 2% higher than PFNM on average for various number of clients.

For advanced deep learning methods, the architectures are often complex, typically, being furnished with skip connection across layers and number of layers is many more than three. In previous Federated Learning methods, those complex networks are rarely tested and it is not clear whether those large-scale neural networks can be fused as well as those light-weighted nets by those model fusion methods. U-Net (Ronneberger et al., 2015) is widely used on medical image segmentation (Du et al., 2020), and heterogenous medical images such as tumor scan images generated by multiple institutions are generally forbidden to be exchanged for privacy issues. Therefore, it is urgent to make popular image segmentation algorithms federated. To this end, we apply several popular federated learning methods on U-Net. The architecture details of U-Net can be found in Ronneberger et al. (2015). In this paper, we technically matched aggregate neurons over skip connection. In addition, U-Net also applies batch normalization technique on feature maps which distinguish U-Net from general FCNNs. As shown in Table 1, the performances of FedAvg and FedProx are very poor. FedAvg and FedProx cannot converge to a stationary result in such complex network architecture (the results go up and down during communication rounds), the result shown in the table is taken from the best result during all 19 communication rounds which is equal to the number of layers in U-net. As shown in Table 1, compared to the case of 2 or 3 layer neural networks, the priority of MAGI is much more significant among popular Federated Learning methods. Accuracy of MAGI on CIMC is at least 6% higher than PFNM with 8 local clients and almost 7% higher with 16 local clients.

In MAGI, we induce a hyper-parameter λ, the weight of the KL-divergence term. We testify the sensitivity of λ which can be referred in Supplement Sect. 6.2. In summary, MAGI is robust on hyper-parameter λ under various conditions.

Number of Neurons in Global Model In deep learning, over-parameterization is still not clearly understood (Zhang et al., 2021; Zhou, 2021). Deep neural networks often contain a great number of parameters even larger than the number of training examples, however, it seems that these over-parameterized models do not suffer from overfitting. Due to this, neural network can be compressed, e.g., pruned to a lighter neural network with fewer parameters (Blalock et al., 2020), while maintaining an acceptable performance. For two neural networks those achieve the same performance on a certain task, the one which comes with fewer parameters would be viewed as a more efficient model. Since generally the more parameters a neural network contains, the more computational resources to deploy it would consume (Cheng et al., 2017). In our study, MAGI is found to have an additional compressing function on neural networks. Considering the discrepancy of dimensions among neurons in different layers, we use the scale of parameters to show the amount of neurons.

As shown in Fig. 2, compared to PFNM, global model obtained by MAGI contains significantly less neurons. In addition, as the weight of the KL divergence term λ in MAGI increases, the amount of neurons in the global neural network declines almost linearly. Compared to PFNM, on MNIST, MAGI with λ close to 1 fuses local models to a global model with almost 8 times fewer parameters for one hidden layer FCNNs, and close to 3 times fewer for multi-layer FCNNs. For

CIFAR 10, MAGI reduces the number of global neurons to around 80% of that fused by PFNM. This phenomenon can be explained as follows. In Proposition 3, we have shown that to minimize the KL penalty essentially picks global neurons with higher prior probability. During the optimizing process directed by the Hungarian algorithm, MAGI tends to match neurons in each local neural network to global neurons with higher prior probability. Therefore, compared to PFNM, in MAGI, those global neurons with high prior probability are more promisingly matched to by neurons in different local models while the other global neurons are less promisingly matched to. Thus, the amount of global neurons in MAGI is less than PFNM. As the weight of KL term increases, the tendency to pick global neuron with high prior probability in MAGI strengthens and the number of global neurons declines as in Fig. 2.

6. Conclusion

In this paper, we have proposed a new federated neural matching method by incorporating the global information into PFNM. It is empirically shown that the new method outperforms other state-of-the-art algorithms for federated learning of neural networks. In future work, it is interesting to extend our model to more advanced architectures. Additionally, the success of KL-divergence suggests it is also likely to try other divergences of probability distributions.

References

Amir Beck. *First-order methods in optimization.* SIAM, 2017.

Davis Blalock, Jose Javier Gonzalez Ortiz, Jonathan Frankle, and John Guttag. What is the state of neural network pruning? *arXiv preprint arXiv:2003.03033*, 2020.

Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Springer, 2010.

Cristian Buciluǎ, Rich Caruana, and Alexandru Niculescu-Mizil. Model compression. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 535–541, 2006.

Hong-You Chen and Wei-Lun Chao. Fedbe: Making Bayesian model ensemble applicable to federated learning. *arXiv preprint arXiv:2009.01974*, 2020.

Yu Cheng, Duo Wang, Pan Zhou, and Tao Zhang. A survey of model compression and acceleration for deep neural networks. *arXiv preprint arXiv:1710.09282*, 2017.

Sebastian Claici, Mikhail Yurochkin, Soumya Ghosh, and Justin Solomon. Model fusion with Kullback–Leibler divergence. *arXiv preprint arXiv:2007.06168*, 2020.

Yuyang Deng, Mohammad Mahdi Kamani, and Mehrdad Mahdavi. Distributionally robust federated averaging. *arXiv preprint arXiv:2102.12660*, 2021.

Getao Du, Xu Cao, Jimin Liang, Xueli Chen, and Yonghua Zhan. Medical image segmentation based on U-net: A review. *Journal of Imaging Science and Technology*, 64(2):20508–1, 2020.

David B Dunson. Commentary: practical advantages of bayesian analysis of epidemiologic data. *American journal of Epidemiology*, 153(12):1222–1226, 2001.

Geoffrey Hinton, Nitish Srivastava, and Kevin Swersky. Neural networks for machine learning lecture 6a overview of mini-batch gradient descent. *Cited on*, 14(8):2, 2012.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. Scaffold: Stochastic controlled averaging for federated learning. In *International Conference on Machine Learning*, pages 5132–5143. PMLR, 2020.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Harold W Kuhn. The Hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955.

Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. *arXiv preprint arXiv:1812.06127*, 2018.

Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3):50–60, 2020.

Tao Lin, Lingjing Kong, Sebastian U Stich, and Martin Jaggi. Ensemble distillation for robust model fusion in federated learning. *arXiv preprint arXiv:2006.07242*, 2020.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, pages 1273–1282, 2017.

Mehryar Mohri, Gary Sivek, and Ananda Theertha Suresh. Agnostic federated learning. In *International Conference on Machine Learning*, pages 4615–4625, 2019.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

Jürgen Schmidhuber. Learning complex, extended sequences using the principle of history compression. *Neural Computation*, 4(2):234–242, 1992.

Sidak Pal Singh and Martin Jaggi. Model fusion via optimal transport. *Advances in Neural Information Processing Systems*, 33, 2020.

Virginia Smith, Chao-Kai Chiang, Maziar Sanjabi, and Ameet S Talwalkar. Federated multi-task learning. In *Advances in Neural Information Processing Systems*, pages 4424–4434, 2017.

Romain Thibaux and Michael I. Jordan. Hierarchical Beta processes and the Indian buffet process. *Journal of Machine Learning Research*, 2(3):564–571, 2007.

Hongyi Wang, Mikhail Yurochkin, Yuekai Sun, Dimitris Papailiopoulos, and Yasaman Khazaeni. Federated learning with matched averaging. *arXiv preprint arXiv:2002.06440*, 2020.

Peng Xiao, Samuel Cheng, Vladimir Stankovic, and Dejan Vukobratovic. Averaging is probably not the optimum way of aggregating parameters in federated learning. *Entropy*, 22(3):314, 2020.

Mikhail Yurochkin, Zhiwei Fan, Aritra Guha, Paraschos Koutris, and XuanLong Nguyen. Scalable inference of topic evolution via models for latent geometric structures. *arXiv preprint arXiv:1809.08738*, 2018.

Mikhail Yurochkin, Mayank Agarwal, Soumya Ghosh, Kristjan Greenewald, and Nghia Hoang. Statistical model aggregation via parameter matching. *Advances in Neural Information Processing Systems*, 32:10956–10966, 2019a.

Mikhail Yurochkin, Mayank Agarwal, Soumya Ghosh, Kristjan Greenewald, Nghia Hoang, and Yasaman Khazaeni. Bayesian nonparametric federated learning of neural networks. In *International Conference on Machine Learning*, pages 7252–7261, 2019b.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning (still) requires rethinking generalization. *Communications of the ACM*, 64(3): 107–115, 2021.

Xinwei Zhang, Mingyi Hong, Sairaj Dhople, Wotao Yin, and Yang Liu. Fedpd: A federated learning framework with optimal rates and adaptivity to non-iid data. *arXiv preprint arXiv:2005.11418*, 2020.

Zhi-Hua Zhou. Why over-parameterization of deep neural networks does not overfit? *Science China Information Sciences*, 64(1):1–3, 2021.

Zhuangdi Zhu, Junyuan Hong, and Jiayu Zhou. Data-free knowledge distillation for heterogeneous federated learning. *arXiv preprint arXiv:2105.10056*, 2021.