FEDERATED NEAREST NEIGHBOR MACHINE TRANSLATION

Anonymous authors
Paper under double-blind review

ABSTRACT

To protect user privacy and meet legal regulations, federated learning (FL) is attracting significant attention. Training neural machine translation (NMT) models with traditional FL algorithm (e.g., FedAvg) typically relies on multi-round model-based interactions. However, it is impractical and inefficient for machine translation tasks due to the vast communication overheads and heavy synchronization. In this paper, we propose a novel federated nearest neighbor (FedNN) machine translation framework that, instead of multi-round model-based interactions, leverages one-round memorization-based interaction to share knowledge across different clients to build low-overhead privacy-preserving systems. The whole approach equips the public NMT model trained on large-scale accessible data with a k-nearest-neighbor (kNN) classifier and integrates the external datastore constructed by private text data in all clients to form the final FL model. A two-phase datastore encryption strategy is introduced to achieve privacy-preserving during this process. Extensive experiments show that FedNN significantly reduces computational and communication costs compared with FedAvg, while maintaining promising performance in different FL settings.

1 INTRODUCTION

In recent years, neural machine translation (NMT) has significantly improved translation quality (Bahdanau et al., 2015; Vaswani et al., 2017; Hassan et al., 2018) and has been widely adopted in many commercial systems. The current mainstream system is first built on a large-scale corpus collected by the service provider and then directly applied to translation tasks for different users and enterprises. However, this application paradigm faces two critical challenges in practice. On the one hand, previous works have shown that NMT models perform poorly in specific scenarios, especially when they are trained on the corpora from very distinct domains(Koehn & Knowles, 2017; Chu & Wang, 2018). The fine-tuning method is a popular way to mitigate the effect of domain drift, but it brings additional model deployment overhead and particularly requires high-quality in-domain data provided by users or enterprises. On the other hand, some users and enterprises pose high data security requirements since business concerns or regulations from the government (e.g., GDPR and CCPA), meaning that we can not directly access private data from users for model training. Thus, a conventional centralized-training manner is infeasible in these scenarios.

In response to this dilemma, a natural way is to leverage federated learning (FL) that enables different data owners to train a global model in a distributed manner while leaving raw private data isolated to preserve data privacy. Generally, standard FL workflow (e.g., *FedAvg* (McMahan et al., 2017)) contains multi-round model-based interactions between server and clients. At each round, the client first performs training on the local sensitive data and sends the model update to the server. The server then aggregates these local updates to build an improved global model. This straightforward idea has been implemented by prior works (Roosta et al., 2021; Passban et al., 2022) that directly apply *FedAvg* for machine translation tasks while introducing some parameter pruning strategies during node communication. Despite this, multi-round model-based interactions are impractical and inefficient for NMT applications. Current NMT models heavily rely on deep neural networks (Vaswani et al., 2017) as the backbone, whose parameters can reach tens of millions or

even hundreds of millions, bringing vast computation and communication overhead. In real-world scenarios, different clients (i.e., users and enterprises) usually have limited computation and communication capabilities, making it difficult to meet frequent model training and node communication requirements in the standard FL workflow. Meanwhile, due to the capability differences between clients, heavy synchronization also hinders the efficacy of FL workflow. Fewer interactions may ease this problem but suffer from significant performance loss.

Inspired by the recent remarkable performance of memorization-augmented techniques (e.g., the k-nearest-neighbor, kNN) in natural language processing (Khandelwal et al., 2020; 2021; Zheng et al., 2021a;b; He et al., 2021) and computer vision (Papernot & Mcdaniel, 2018; Orhan, 2018), we take a new perspective to deal with above federated NMT training problem. In this paper, we propose a novel **F**ederated **N**earest **N**eighbor (FedNN) machine translation framework, which equips the public NMT model trained on large-scale accessible data with a k-nearest-neighbor (kNN) classifier and integrates the external datastore constructed by private data from all clients to form the final FL model. In this way, we replace the multi-round model-based interactions in the conventional FL paradigm with the one-round encrypted memorization-based interaction to share knowledge across different clients and drastically reduce computation and communication overhead.

Specifically, FedNN follows a similar server-client architecture. The server holds large-scale public accessible data to train the public NMT model for all clients, while the client leverages their local private data to construct an external datastore that is collected to augment the public NMT model via kNN retrieval. Based on this architecture, the key is to collect and broadcast all the datastores built from different clients, while avoiding privacy leakage. We design a two-phase datastore encryption strategy to achieve privacy-preserving during the memorization-based interaction process. This strategy adopts an adversarial mode between server and client. On the one hand, the server builds $(\mathcal{K}, \mathcal{V})$-encryption model for the client to increase the difficulty of reconstructing the private text from the datastores constructed by other clients. The $\mathcal{K}$-encryption model is coupled with the public NMT model to ensure the correctness of kNN retrieval. On the other hand, all clients apply the shared content-encryption model for a local datastore during the collecting process so that the server can not directly access the original datastore. During inference, the client leverages the corresponding content-decryption model to obtain the final integrated datastore.

We set up several FL scenarios (e.g., Non-IID and IID settings) with multi-domain English-German (En-De) translation dataset, and demonstrate that our approach not only drastically reduces computation and communication costs compared with FedAvg, but also achieves the state-of-the-art performance in Non-IID setting. Additional experiments verify that FedNN can easily scale to large-scale clients with sparse data scenarios thanks to the memorization-based interaction across clients.

2 FEDNMT: FEDERATED NEURAL MACHINE TRANSLATION

The current commercial NMT system is built on a large-scale corpus collected by the service provider and directly applied to different users and enterprises. However, this mode is difficult to flexibly satisfy the model customization and privacy protection requirements from users and enterprises. In this work, we focus on the more general application scenario, where users and enterprises are involved in training the NMT model with the service provider collaboratively, but the service provider cannot directly access the private data for model training.

Formally, this application scenario consists of $|C|$ clients (i.e., user or enterprise) and a central server (i.e., service provider). The central server holds vast accessible translation data $\mathcal{D}_s = \{(\mathbf{x}_s^i, \mathbf{y}_s^i)\}_{i=1}^{|\mathcal{D}_s|}$, where $\mathbf{x}^i = (x_1^i, x_2^i, ..., x_{|\mathbf{x}^i|}^i)$ and $\mathbf{y}^i = (y_1^i, y_2^i, ..., x_{|\mathbf{y}^i|}^i)$ (for brevity, we omit the subscript s here) are text sequences in the source and target languages, respectively. The central server can easily train a public NMT model f_θ based on this corpus, where θ denotes model parameters. For each client c, it contains private data $\mathcal{D}_c = \{(\mathbf{x}_c^i, \mathbf{y}_c^i)\}_{i=1}^{|\mathcal{D}_c|}$, which is usually sparse in practice (i.e., $|\mathcal{D}_c| \ll |\mathcal{D}_s|$) and only accessible to itself.

This application scenario actually falls into the federated framework. The straightforward idea is to apply the vanilla FL method (i.e., *FedAvg* (McMahan et al., 2017)) or its variants (Roosta et al., 2021; Passban et al., 2022). Generally, *FedAvg* contains multi-round model-based interaction updates between server and clients. At each round r, each client c downloads a global model f_{θ^r} from the server and optimizes it using its local private data $\mathcal{D}_c$. Then, the local updates θ_c^r are uploaded to the server to form a new model $f_{\theta^{r+1}}$. The server aggregates clients' information via a simple parameter averaging technique: $\theta^{r+1} = \sum_{m=1}^{C} \frac{n_m}{n} \theta_m^r$, where n_m the number of data points in the m-th client's private dataset, and n is the total number of all training data. However, such FL workflow is impractical and inefficient for the above application scenario. The parameters of the NMT model typically reach tens of millions or even hundreds of millions, bringing vast computation and communication overhead. In addition, there is usually system heterogeneity between server and clients, i.e., mismatch of bandwidth, computation resources, etc. It is difficult to meet frequent model training and node communication requirements in the standard FL workflow.

3 FEDNN: FEDERATED NEAREST NEIGHBOR MACHINE TRANSLATION

Inspired by the advanced memorization-augmented techniques, e.g., kNN-MT (Khandelwal et al., 2021), that have shown the promising capability of directly incorporating the pre-trained NMT model with external knowledge via kNN retrieval, we explore to leverage one-round memorization-based interaction to achieve knowledge sharing across different clients, instead of multi-round model-based interactions. In this work, we propose a novel **F**ederated **N**earest **N**eighbour (FedNN) machine translation framework, which extends the promising capability of kNN-MT in the federated scenario and introduces two-phase datastore encryption strategy to avoid data privacy leakage. The whole approach complements the public NMT model built by the central server with a kNN classifier and safely collects the local datastore constructed by private text data in all clients to form the final FL model. The entire workflow of FedNN is illustrated in Figure 1, consisting of initialization, one-round memorization-based interaction and model inference on clients.

3.1 INITIALIZATION

The FedNN fremework starts with the public NMT model and encryption models. The central server is responsible for optimizing the public NMT model f_θ with the accessible translation data $\mathcal{D}_s$. Following kNN-MT Khandelwal et al. (2021), the memorization (also called as datastore) is a set of key-value pairs. Given a parallel sentence pair $(\mathbf{x}_s, \mathbf{y}_s) \in \mathcal{D}_s$, the public NMT model generates the context representation $f_\theta(\mathbf{x}_s, y_{s,<t})$ in the last decoder layer at each timestep t. The whole datastore $\mathcal{M}_s = (\mathcal{K}_s, \mathcal{V}_s)$ is constructed by taking the representation $f_\theta(\mathbf{x}_s, y_{s,<t})$ as key and ground-truth y_t as value:

$$\mathcal{M}_s = (\mathcal{K}_s, \mathcal{V}_s) = \bigcup_{(\mathbf{x}_s, \mathbf{y}_s) \in \mathcal{D}_s} \{(f_\theta(\mathbf{x}_s, y_{s,<t}), y_{s,t}), \forall y_{s,t} \in \mathbf{y}_s\}. \tag{1}$$

Based on $\mathcal{M}_s$, the central server further constructs $\mathcal{K}$-Encryption model $f_{\mathcal{K}\text{E}}(.)$ that is coupled with the public NMT model. This design is for the client, which aims to increase the difficulty of reconstructing the private text from the datastores constructed by other clients. The $f_{\mathcal{K}\text{E}}(.)$ should also satisfy the correctness of kNN retrieval during inference and the detailed $\mathcal{K}$-Encryption algorithm selection see Sec. 3.4. For all clients, they prepare the shared content-encryption model $f_{\text{CE}}(.)$ and corresponding content-decryption model $f_{\text{DE}}(.)$, which is applied to the local datastore so that the server can not directly access the original datastore. The content-encryption algorithm selection is relatively loose, which is detailed in Sec. 3.4.

3.2 MEMORIZATION-BASED INTERACTION

The entire memorization-based interaction can be decomposed into two main processes, i.e., private memorization construction and global memorization aggregation. The central server broadcasts the public NMT

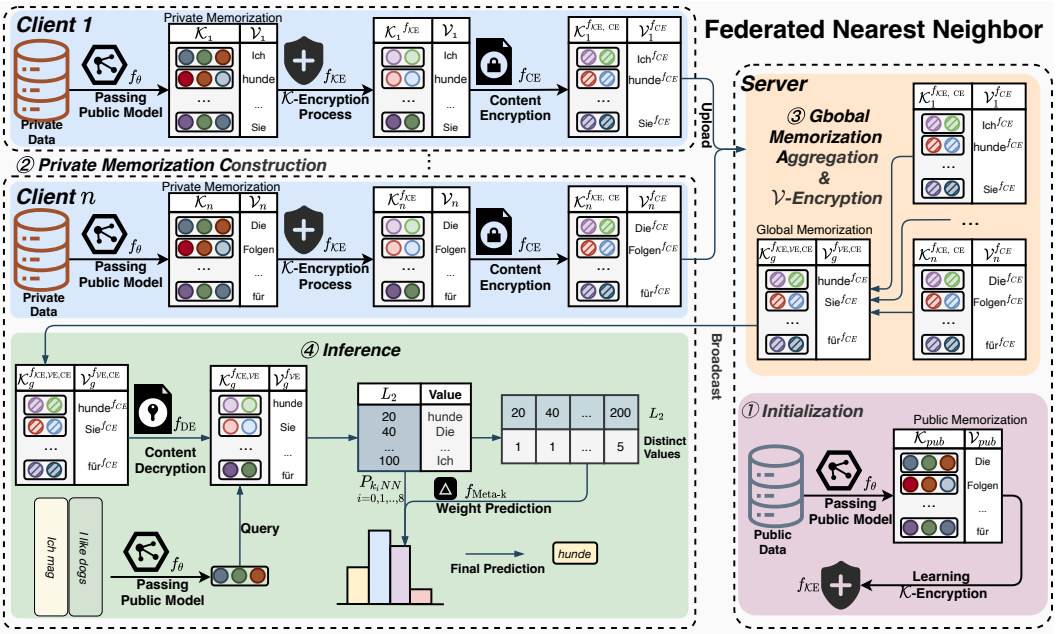

Figure 1: The workflow of our FedNN.

model and $\mathcal{K}$-Encryption model for all clients to build the local encrypted datastore. Specifically, for each client c, we adopt a similar construction way as Equation 1 to yield the local datastore $\mathcal{M}_c^{f_{\mathcal{K}\text{E}}}$ via private data $\mathcal{D}_c$, with the difference that the $\mathcal{K}$-Encryption model is further applied to preserve private information:

$$\mathcal{M}_c^{f_{\mathcal{K}\text{E}}} = (\mathcal{K}_c^{f_{\mathcal{K}\text{E}}}, \mathcal{V}_c) = \bigcup_{(\mathbf{x}_c,\mathbf{y}_c)\in\mathcal{D}_c} \{(f_{\mathcal{K}\text{E}}(f_\theta(\mathbf{x}_c, y_{c,<t})), y_{c,t}), \forall y_{c,t} \in \mathbf{y}_c\}. \tag{2}$$

Next, in order to ensure that above datastore is not explicitly available to the server, we encrypt key-value pairs by $f_{\text{CE}}(.)$ before uploading them to the server, formalized as:

$$\mathcal{M}_c^{f_{\mathcal{K}\text{E},\,\text{CE}}} = (\mathcal{K}_c^{f_{\mathcal{K}\text{E},\,\text{CE}}}, \mathcal{V}_c^{f_{\text{CE}}}) = \bigcup_{(\mathbf{x}_c,\mathbf{y}_c)\in\mathcal{D}_c} \{f_{\text{CE}}(f_{\mathcal{K}\text{E}}(f_\theta(\mathbf{x}_c, y_{c,<t})), y_{c,t}), \forall y_{c,t} \in \mathbf{y}_c\}. \tag{3}$$

Once the central server has received the private memorization from all clients, it directly aggregates all datastores via simple key-value pair concatenation and then performs $\mathcal{V}$-encryption operation (i.e., shuffling on key-value pair to avoid clients identifying the source of datastore) to obtain the global memorization $\mathcal{M}_g^{f_{\mathcal{K}\text{E},\,\mathcal{V}\text{E},\,\text{CE}}}$ which is sent to all clients for model inference. Then each client c decrypts the contents of the received global memorization to obtain an accessible integrated datastore $\mathcal{M}_g^{f_{\mathcal{K}\text{E},\,\mathcal{V}\text{E}}}$.

3.3 Model Inference on Clients

For model inference on clients, we follow the adaptive kNN-MT (AK-MT) (Zheng et al., 2021a) to incorporate the public NMT model with integrated datastore $\mathcal{M}_g^{f_{\mathcal{K}\text{E},\,\mathcal{V}\text{E}}}$ via adaptive kNN retrieval. AK-MT introduces a lightweight Meta-k Network $f_{\text{Meta-}k}$ to dynamically determine the number of retrieved tokens to consider at each step, and has promising generalization ability that we could train $f_{\text{Meta-}k}$ with a very small data in any vertical scenario and then directly apply it to other scenarios. Since the parameters in Meta-k Network are negligible, we ignore the additional training and communication costs of AK-MT in our FedNN.

Concretely, given the already generated words $\hat{y}_{<t}$ and source input $\mathbf{x}$, AK-MT augments the probability distribution of t-th target token y_t vis kNN retrieval based on the context representation $f_\theta(\mathbf{x}, \hat{y}_{<t})$. It considers a set of possible ks that are smaller than K, i.e., $k \in \mathcal{S}$ where $\mathcal{S} = \{0\} \cup \{k_i \in \mathbb{N} \mid \log_2 k_i \in \mathbb{N}, k_i \leq K\}$. Then K nearest neighbors of the current representation $f_\theta(\mathbf{x}, \hat{y}_{<t})$ are retrieved from the integrated datastore $\mathcal{M}_g^{f_{\mathcal{KE}}, \mathcal{V}_{\mathrm{E}}}$ according to the squared L_2 distance $d(\cdot, \cdot)$. The L_2 distances from $f_\theta(\mathbf{x}, \hat{y}_{<t})$ to each neighbor (h_i, v_i) is denoted as $d_i = d(h_i, f_\theta(\mathbf{x}, \hat{y}_{<t}))$ and the count of distinct values in top-i is denoted as c_i. The normalized weights of applying different kNN retrieval results are computed as:

$$p_{\text{Meta}}(k) = \text{softmax}(f_{\text{Meta}}([d_1, ..., d_K; c_1, ..., c_K])),$$

where $p_{\text{Meta}}(k)$ is Meta-k Network containing two feed-forward networks with non-linearity between them. For $k_i \in \mathcal{S}$, the kNN retrieval distribution over the vocabulary is computed as:

$$p_{k_i\text{NN}}\left(y_t|\mathbf{x}, \hat{y}_{<t}\right) \propto \sum_{(h_i, v_i)} \mathbb{1}_{y_t = v_i} \exp\left(\frac{-d^2\left(h_i, f_\theta\left(\mathbf{x}, \hat{y}_{<t}\right)\right)}{T}\right), \tag{4}$$

where T is the temperature to control the sharpness of softmax function. The final prediction probability is a weighted ensemble:

$$p\left(y_t|\mathbf{x}, \hat{y}_{<t}\right) = \sum_{k_i \in \mathcal{S}} p_{\text{Meta}}\left(k_i\right) \cdot p_{k_i\text{NN}}\left(y_t|\mathbf{x}, \hat{y}_{<t}\right). \tag{5}$$

3.4 Encryption and Privacy Discussions

- **$\mathcal{K}$-Encryption.** In this work, we adopt the Product Quantizer Jégou et al. (2011) algorithm to build $\mathcal{K}$-encryption model, which decomposes the space into a Cartesian product of low-dimensional subspaces and quantifies each subspace separately into segment code representations. Further, we map the representation k to the above shortcode representation, which cannot be reversed to the original k, further reducing the possibility of reverse private data. This way also satisfy the correctness of kNN retrieval after encryption. Note that any algorithm that can make k distorted and irreducible can be adopted, such as PCA.
- **Content-Encryption.** We need to generate the ciphertext by content-encryption model and ensure that it is indistinguishable from the chosen plaintext attacks Oded (2004). Since each record of the memorization can be regarded as a string, it can be encrypted and decrypted using any asymmetric encryption algorithm, such as Paillier, Elgamal (Gamal, 1984) and RSA (Rivest et al., 1978), etc.
- **Threat Models and Leakage Quantifying.** We consider all clients involved in the training are semi-honest following prior works (Zhang et al., 2021; Bonawitz et al., 2016). In the semi-honest setting, each client adheres to the designed protocol but it may attempt to infer information about other participant's input (i.e., memorization). Under this setting, our method has achieved different protection levels for the client and server side. Specifically, for the server side, our mechanism can achieve the same protection level to the conventional Public-key cryptography system (e.g., RSA). Thus, the server cannot obtain any useful information from the encrypted data. For the client side, since the client can obtain shared information generated by Product Quantizer (i.e.,$f_{\mathcal{KE}}(f_\theta(\mathbf{x}_c, y_{c,<t})), y_{c,t})$) from other clients, the aim of this paper is to prevent the shared information from reconstruction attacks (i.e., recovering private text data from the datastore). To this end, we introduce a new metric to quantify the privacy leakage of the shared datastore information (see more details in Sec. 4.5).

4 Experiments

4.1 Setup

We adopt the WMT14 En-De data (Bojar et al., 2014) and multi-domain En-De dataset (Koehn & Knowles, 2017) to simulate two typical FL scenarios for model evaluation: 1) the non-independently identically

Table 1: BLEU score [%] of different methods on clients and server test sets. "△" refers to the improvement of the method compared with the Public Model. The subscript "$1/\infty$" indicate that the server perform model aggregation after one or infinite epochs (i.e., client model convergence) of client model updates, respectively. The superscript "s" indicates that the server data is also involved in model training of FedAvg. "Comm." and "Comp." refer to communication and computational overhead in "GB" and "K Step" respectively.

	Methods	Client Test			Server Test	Overall Performance				Cost		Inference
		IT	Law	Medical	WMT14	Client	△	Global	△	Comm.	Comp.	Speed
	Centralized Model	37.30	49.72	47.40	26.58	44.81	—	40.25	—	—	—	1.00×
	Public Model	26.62	35.91	30.27	26.63	30.93	—	29.86	—	—	—	1.00×
non-IID	**FedAvg$_1^s$**	26.99	37.65	32.36	26.63	32.33	+1.40	30.91	+1.05	388.12	300	1.00×
non-IID	**FedAvg$_1$**	28.26	53.00	45.90	13.45	42.39	+11.45	35.15	+5.30			
non-IID	**FedAvg$_\infty^s$**	27.04	38.37	32.32	26.33	32.66	+1.72	31.08	+1.22	4.85	30	1.00×
non-IID	**FedAvg$_\infty$**	17.03	47.06	30.61	13.33	31.57	+0.63	27.01	- 2.85			
non-IID	**FT-Ensemble**	30.11	38.14	39.15	17.13	35.80	+4.87	31.13	+1.28	4.85	30	0.39×
non-IID	**FedNN**	35.62	55.57	49.21	22.29	46.80	+15.87	40.67	+10.82	5.08	5	0.75×
IID	**FedAvg$_1^s$**	30.83	43.47	39.22	26.36	37.84	+6.91	34.97	+5.11	388.12	300	1.00×
IID	**FedAvg$_1$**	37.99	54.53	50.23	14.80	47.58	+16.65	39.39	+9.53			
IID	**FedAvg$_\infty^s$**	29.23	39.67	34.49	26.28	34.46	+3.53	32.42	+2.56	4.85	30	1.00×
IID	**FedAvg$_\infty$**	34.71	48.68	44.79	16.23	42.73	+11.79	36.10	+6.25			
IID	**FT-Ensemble**	36.74	51.34	47.49	16.42	45.19	+14.26	38.00	+8.14	4.85	30	0.39×
IID	**FedNN**	34.64	54.45	47.98	23.15	45.69	+14.76	40.06	+10.20	5.08	5	0.75×

distribution (Non-IID setting) where each client distributes data from different domains; 2) the independently identically distribution (IID setting) where each client contains the same data distribution from all domains. In our experiment, WMT14 and multi-domain En-De dataset are viewed as the server's data and clients' private data, respectively. The multi-domain data provides a natural division for exploring the Non-IID setting, of which we assign IT, Medical, and Law domain data to each of the three clients. For the IID setting, we mix the above domain data and randomly sample the same number of samples from it for each client. More dataset and implementation details can be found in the Appendix.

We compare our method **FedNN** with several baselines: 1) **Centralized Model**: A standard centralized-training method uses all clients and server data to train a global NMT model; 2) **Public Model**: A generic NMT model is trained on only server data and used for initializing the client-side model; 3) **FedAvg**: A vanilla FL approach (McMahan et al., 2017) that iteratively trains a global model through multi-round model-based interactions; 4) **FT-Ensemble**: This method uses the client's private data to fine-tune the public model. During inference, the output probability distributions of all fine-tuned models are integrated for token prediction.

4.2 Main Results

Table 1 illustrates the performance of FedNN and baseline methods. We observe that the public NMT model has a 13.88 BLEU gap on average compared to the centralized model. Restricted by privacy protection, it is not always possible to access private data for centralized training, so we attempt to build high-performance global models using FL technique. Specifically, we evaluate two FL settings as follows.

In the Non-IID setting, we have the following findings. (1) all methods outperform Public Model overall on the client test set but show degradation on the server test set. This implies that FL can fuse helpful information from multiple parties but also suffers from a varying degree of knowledge conflict and catastrophic forgetting. (2) The performance of FedAvg is affected by whether the server data is involved in training and model

aggregation frequency. Specifically, the overall performance tends to be similar to the Public Model when server data is involved in training. On the contrary, the performance tends to improve the client test set performance, and the gain shows Law>Medical>IT. The reason is that the weights assigned to the client model when FedAvg aggregates show a positive correlation with the size of the client data set. For the aggregation frequency, FedAvg_1 is much better than FedAvg_∞. Because frequent iterative aggregation can significantly reduce the parameter conflicts between models, but at the same time, it brings high communication cost. (3) FT-Ensemble is better than FedAvg_∞ overall, which implies less knowledge conflict due to the fusion of output probabilities. (4) FedNN has a significant performance improvement of 4.41/1.99 BLEU scores on average on the client test set compared to FedAvg1 and Centralized Model, and maintains a competitive performance on the server test set. This is due to FedNN's ability to capture client-side knowledge by memroization and integrate it with the Public Model. (5) Although FedNN slightly slows inference time, it significantly reduces communication and computation overhead and improves translation quality, which is tolerable for clients.

In the IID setting, we can observe some different findings. (1) All baseline methods outperform centralized learning overall on the client test set when server data is not involved in training. The reason is that there is no statistical heterogeneity among clients, so fewer parameter conflicts and less conflict of probability outputs. (2) FedNN performance is slightly weaker than in the Non-IID setting. It indicates that the benefit of the memorization-based interaction is more significant when the data distribution is more heterogeneous.

Overall, FedNN shows stable performance with less communication and computational overhead in Non-IID and IID settings, which demonstrates the practicality of memorization-based interaction mechanisms.

4.3 Impact of Client's Number

In this section, we explore the ability of FedNN and other FL methods on a larger number of clients. We adopt the number of clients ranging from $(3, 6, 12, 18)$ for quick experiments.[1] The detailed results of the non-IID and IID settings are shown in Figure 2.

- **Comparisons with FL Methods.** As the number of clients increases, we observe that: (1) Both FedAvg_1 and FT-Ensemble show varying degrees of performance degradation on the client test sets, especially FT-Ensemble. We conjecture that the limited local data cannot support the training of the client-side model and retains most of the knowledge of the Public Model. (2) FedNN outperforms the other baselines in both private and global performance in the Non-IID setting; it maintains a similar private test set performance to FedAvg_1 in the IID setting and keeps a higher global performance. This indicates that FedNN fused with the memorization mechanism can easily scale to large-scale client scenarios and avoid performance loss due to insufficient local private data. The cost analysis of FL methods in this setting is illustrated in Appendix.
- **Comparisons with Personalized Methods.** We also compare FedNN with the personalized methods FT (fine-tuning the public model with only client-side data) and AK-MT (constructing a datastore with only client-side data and inferencing with assisted Meta-k network). We find that AK-MT and FT perform similarly, thanks to the ability of AK-MT to capture the personalized knowledge by local memorization. In addition, the performance of both AK-MT and FT tends to decrease in the Non-IID setting as clients' number increase, while FedNN hardly decreases. In the IID setting, although the performance of all methods degrades, FedNN still achieves the best performance on all clients test sets. This is thanks to the global memorization, which can be retrieved by kNN search to capture more similar patterns than the local memorization to assist in inference.

[1] Due to the limited resources in our experiments, there are no more domains to ensure the non-IID setting as the number of clients increases. Thus, we directly separate the non-IID and IID data distributions in main experiment with the ratio of $(1, \frac{1}{2}, \frac{1}{4}, \frac{1}{6})$. Note that the non-IID setting here is not the strictly non-IID setting, but it is worth exploring.

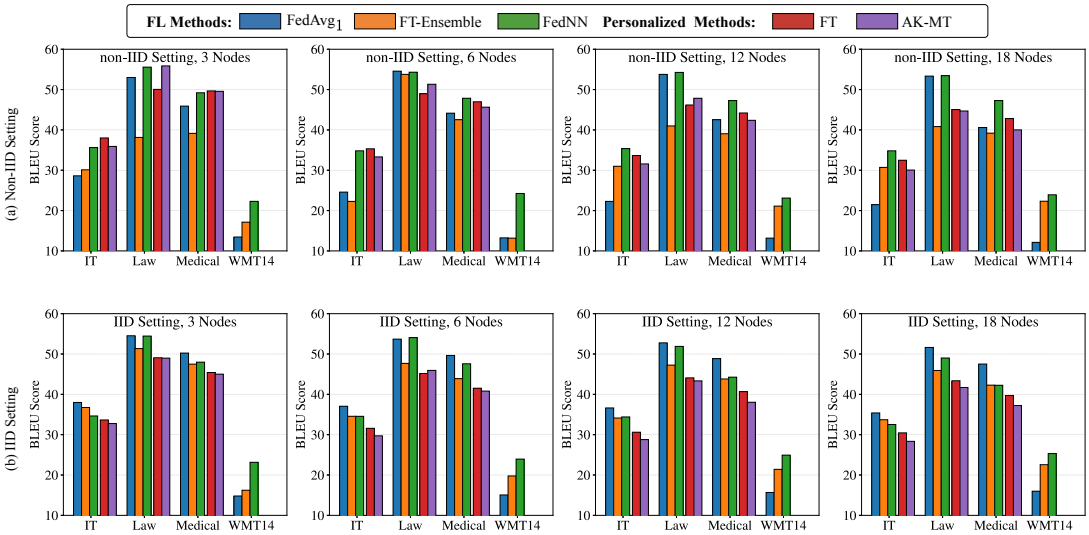

Figure 2: The impact of the number of clients.

4.4 Impact of Data Heterogeneity.

To further explore the effect of data heterogeneity between three clients on FL performance, we introduce the mixed ratio $\alpha \in \{0, 0.2, 0.4, 0.6, 0.8\}$ to construct the data distribution that we want. We randomly take a proportion of α from each domain to construct the IID dataset. Then remain domain data is mixed with one-third of this IID dataset to form the final data distribution. As $\alpha \to 0$, partitions tend to be more heterogeneous (Non-IID), and conversely, the data distribution is more uniform. The detailed results are shown in Figure 3. First, the performance of the personalized methods (FT and AK-MT) is degraded as data heterogeneity decrease, which is caused by the reduction of domain-specific data available in the client. FT-Ensemble performance also decreases across all clients, limited by the FT. In addition, FedAvg shows opposite performance trends between Law and IT, Medical. This is because when FedAvg aggregates, the model weight of each client is proportional to the data size. And as α increases, the data size between clients tends from $|\mathcal{D}_{Law}| \gg |\mathcal{D}_{Medical}| \gg |\mathcal{D}_{IT}|$ to equally to $\frac{1}{3}(|\mathcal{D}_{Law}|+|\mathcal{D}_{Medical}|+|\mathcal{D}_{IT}|)$. Our FedNN maintains stable and remarkable performance across all clients and significantly outperforms other methods in the server test set. It indicates that the memorization-based interaction mechanism used in FedNN can capture and retain the knowledge of all clients, avoiding the knowledge conflict based on traditional model parameters fusion.

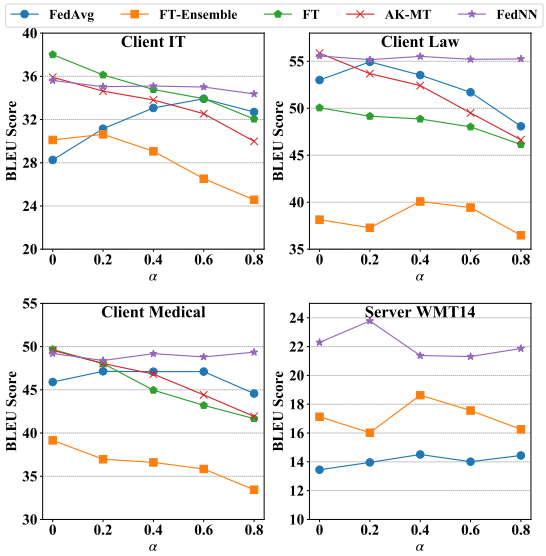

Figure 3: The impact of data distribution heterogeneity on performance.

4.5 QUANTITATIVE ANALYSIS OF PRIVACY

In this section, we quantify the risk of privacy leakage of global memorization. Since all clients can obtain the public NMT model, meaning that they could also construct the text-record pair to train a reverse attack model to reconstruct the private data. In this experiment, we train a reconstruction threat model on the datastore built by the IT data, and then attempt to recover private text from the datastore constructed from Medical and Law data. The detailed model structure and data set construction are in the Appendix. We use the BLEU (cite) to evaluate the degree of privacy leakage, are shown in Table 2. We can observe that the $(\mathcal{K}, \mathcal{V})$-encryption model increases the difficulty of reconstructing the private text from the datastores constructed by other clients. Whether the input is an unencrypted key or a key encrypted by $f_{\mathcal{K}\mathrm{E}}$, the threat model scores very low in both Law and Medical, i.e., it is difficult to recover valid information from global memorization.

Table 2: The reconstruction BLEU score (%) of the attack model.

Input	**IT**	**Law**	**Medical**
$\mathcal{K}$	11.25	8.73	9.21
$\mathcal{K}^{f_{\mathcal{K}\mathrm{E}}}$	7.81	6.81	6.52

5 RELATED WORK

Federated Learning for NLP. The first federated learning algorithm (McMahan et al., 2017) was proposed for language modeling and image classification tasks. Since then, the theory and framework of federated learning have been applied to many fields, including machine learning Yang et al. (2019), computer vision (Lim et al., 2020), data mining (Chai et al., 2021), and edge computing (Ye et al., 2020). Recently, researchers have started to explore applications of federation learning in privacy-preserving NLP, such as next word prediction (Hard et al., 2018; Chen et al., 2019), aspect sentiment classification (Qin et al., 2021), and relation extraction Sui et al. (2021). Recent researchers (Roosta et al., 2021; Passban et al., 2022) have applied FL to NMT task while introducing parameter pruning strategies during node communication. Nevertheless, multi-rounds model-based interactions are impractical and inefficient for NMT because of the huge computational and communication overheads associated with large NNT models. Unlike them, our work proposes an efficient and effective federated nearest neighbor machine translation framework that requires only one-round memorization interaction to obtain a high-quality global translation system.

Memorization-Augmented Neural Networks. Recently, networks equipped with retrieval modules by equipping pre-trained models have attracted much attention from the community and have achieved good performance on language modeling Khandelwal et al. (2020); He et al. (2021), named entity recognition (Wang et al., 2022), machine translation (Khandelwal et al., 2021; Zheng et al., 2021a;b; He et al., 2021), and some computer vision tasks. For NMT tasks, kNN-MT Khandelwal et al. (2021) is the first NMT system to be enhanced with kNN retrieval and has achieved significant improvements. Our work extends the promising capability of kNN-MT in the federated scenario and introduces two-phase datastore encryption strategy to avoid data privacy leakage.

6 CONCLUSION

In this paper, we present a novel federated nearest neighbor machine translation framework to handle the federated NMT training problem that involves multi-round model-based interactions. This memorization-augmented FL framework equips the public NMT model trained on large-scale accessible data with a k-nearest-neighbor classifier and safely collects the local datastore constructed by all clients via two-phase datastore encryption strategy to form the global FL model. Experimental results show that our proposed approach significantly reduces computational and communication costs compared with FedAvg, while achieving promising performance in different FL settings. In the future, we would like to explore this approach on other sequence-to-sequence tasks. Another interesting direction is to further investigate the effectiveness of our method on a larger number of clients, such as hundreds of clients.

REFERENCES

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473, 2015.

Ondrej Bojar, Christian Buck, Christian Federmann, Barry Haddow, Philipp Koehn, Johannes Leveling, Christof Monz, Pavel Pecina, Matt Post, Herve Saint-Amand, Radu Soricut, Lucia Specia, and Ales Tamchyna. Findings of the 2014 workshop on statistical machine translation. In *WMT@ACL*, 2014.

Kallista A. Bonawitz, Vladimir Ivanov, Ben Kreuter, Antonio Marcedone, H. Brendan McMahan, Sarvar Patel, Daniel Ramage, Aaron Segal, and Karn Seth. Practical secure aggregation for federated learning on user-held data. *CoRR*, abs/1611.04482, 2016. URL `http://arxiv.org/abs/1611.04482`.

Di Chai, Leye Wang, Kai Chen, and Qiang Yang. Secure federated matrix factorization. *IEEE Intelligent Systems*, 36:11–20, 2021.

Mingqing Chen, Ananda Theertha Suresh, Rajiv Mathews, Adeline Wong, Cyril Allauzen, Franccoise Beaufays, and Michael Riley. Federated learning of n-gram language models. In *CoNLL*, 2019.

Chenhui Chu and Rui Wang. A survey of domain adaptation for neural machine translation. In *COLING*, 2018.

Taher El Gamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Trans. Inf. Theory*, 31:469–472, 1984.

Andrew Hard, Kanishka Rao, Rajiv Mathews, Françoise Beaufays, Sean Augenstein, Hubert Eichner, Chloé Kiddon, and Daniel Ramage. Federated learning for mobile keyboard prediction. *ArXiv*, abs/1811.03604, 2018.

Hany Hassan, Anthony Aue, Chang Chen, Vishal Chowdhary, Jonathan Clark, Christian Federmann, Xuedong Huang, Marcin Junczys-Dowmunt, William D. Lewis, Mu Li, Shujie Liu, Tie-Yan Liu, Renqian Luo, Arul Menezes, Tao Qin, Frank Seide, Xu Tan, Fei Tian, Lijun Wu, Shuangzhi Wu, Yingce Xia, Dongdong Zhang, Zhirui Zhang, and Ming Zhou. Achieving human parity on automatic chinese to english news translation. *ArXiv*, abs/1803.05567, 2018.

Junxian He, Graham Neubig, and Taylor Berg-Kirkpatrick. Efficient nearest neighbor language models. In *EMNLP*, 2021.

Hervé Jégou, Matthijs Douze, and Cordelia Schmid. Product quantization for nearest neighbor search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33:117–128, 2011.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*, 7:535–547, 2021.

Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. Generalization through memorization: Nearest neighbor language models. *ArXiv*, abs/1911.00172, 2020.

Urvashi Khandelwal, Angela Fan, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. Nearest neighbor machine translation. *ArXiv*, abs/2010.00710, 2021.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2015.

Philipp Koehn and Rebecca Knowles. Six challenges for neural machine translation. In *NMT@ACL*, 2017.

Wei Yang Bryan Lim, Nguyen Cong Luong, Dinh Thai Hoang, Yutao Jiao, Ying-Chang Liang, Qiang Yang, Dusit Tao Niyato, and Chunyan Miao. Federated learning in mobile edge networks: A comprehensive survey. *IEEE Communications Surveys & Tutorials*, 22:2031–2063, 2020.

H. B. McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *AISTATS*, 2017.

Goldreich Oded. Foundations of cryptography: Volume 2, basic applications. 2004.

A. Emin Orhan. A simple cache model for image recognition. *ArXiv*, abs/1805.08709, 2018.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. fairseq: A fast, extensible toolkit for sequence modeling. In *NAACL*, 2019.

Nicolas Papernot and Patrick Mcdaniel. Deep k-nearest neighbors: Towards confident, interpretable and robust deep learning. *ArXiv*, abs/1803.04765, 2018.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *ACL*, 2002.

Peyman Passban, Tanya Roosta, Rahul Gupta, Ankit R. Chadha, and Clement Chung. Training mixed-domain translation models via federated learning. In *NAACL*, 2022.

Han Qin, Guimin Chen, Yuanhe Tian, and Yan Song. Improving federated learning for aspect-based sentiment analysis via topic memories. In *EMNLP*, 2021.

Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, 21:120–126, 1978.

Tanya Roosta, Peyman Passban, and Ankit R. Chadha. Communication-efficient federated learning for neural machine translation. *ArXiv*, abs/2112.06135, 2021.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. *ArXiv*, abs/1508.07909, 2016.

Dianbo Sui, Yubo Chen, Kang Liu, and Jun Zhao. Distantly supervised relation extraction in federated settings. In *EMNLP*, 2021.

Ashish Vaswani, Noam M. Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *ArXiv*, abs/1706.03762, 2017.

Shuhe Wang, Xiaoya Li, Yuxian Meng, Tianwei Zhang, Rongbin Ouyang, Jiwei Li, and Guoyin Wang. knn-ner: Named entity recognition with nearest neighbor search. *ArXiv*, abs/2203.17103, 2022.

Qiang Yang, Yang Liu, Yong Cheng, Yan Kang, Tianjian Chen, and Han Yu. Federated learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 13(3):1–207, 2019.

Yunfan Ye, Shen Li, Fang Liu, Yonghao Tang, and Wanting Hu. Edgefed: Optimized federated learning based on edge computing. *IEEE Access*, 8:209191–209198, 2020.

Junpeng Zhang, Mengqian Li, Shuiguang Zeng, Bin Xie, and Dongmei Zhao. A survey on security and privacy threats to federated learning. In *2021 International Conference on Networking and Network Applications, NaNA 2021, Lijiang City, China, October 29 - Nov. 1, 2021*, pp. 319–326. IEEE, 2021. doi: 10.1109/NaNA53684.2021.00062. URL https://doi.org/10.1109/NaNA53684.2021.00062.

Xin Zheng, Zhirui Zhang, Junliang Guo, Shujian Huang, Boxing Chen, Weihua Luo, and Jiajun Chen. Adaptive nearest neighbor machine translation. *ArXiv*, abs/2105.13022, 2021a.

Xin Zheng, Zhirui Zhang, Shujian Huang, Boxing Chen, Jun Xie, Weihua Luo, and Jiajun Chen. Non-parametric unsupervised domain adaptation for neural machine translation. *ArXiv*, abs/2109.06604, 2021b.

A APPENDIX

A.1 IMPLEMENTATION DETAILS AND EVALUATION.

We follow the recipe [2] to perform data pre-processing. The Moses [3] toolkit is used to tokenize all sentences and learn bpe-code in the publicly available corpus WMT14. Based on it, we split all the words of the above datasets into subword units (Sennrich et al., 2016). All experiments are implemented based on the FAIRSEQ(Ott et al., 2019) toolkit. We train the public model on the WMT14 En-De dataset and use it as the initialization model for all other methods. We adopt Transformer (Vaswani et al., 2017) as model structure of all baselines, it consists of 6 transformer encoder layers, and 6 transformer decoder layers. The input embedding size of the transformer layer is 512, the FFN layer dimension is 2048, and the number of self-attention heads is 8. During training, we deploy the Adam optimizer (Kingma & Ba, 2015) with a learning rate of 5e-4 and 4K warm-up updates to optimize model parameters. Both label smoothing coefficient and dropout rate are set to 0.1. The batch size is set to 16K tokens. We train all models with four Tesla-V100 GPU and set patience to 5 to select the best checkpoint on the validation set. The FAISS (Johnson et al., 2021) is leveraged to construct the datastore and and use its IndexIVFPQ strategy to implement Product Quantizer $\mathcal{K}$-encryption and fast nearest neighbor search. We utilize the FAISS to learn 4096 cluster centroids on public datastore, and apply it to client's datastore. During inference, the beam size and length penalty are set to 5 and 1 for all methods and we search 64 clusters for each target token when using FAISS. In all experiments, we report the case-sensitive BLEU score (Papineni et al., 2002) using sacreBLEU[4].

Table 3: Statistics of the training sets of multi-domain and WMT14.

	WMT14	IT	Medical	Laws
Tokens (DE)	$113,526,532$	$2,907,789$	$5,650,447$	$16,418,294$
Sentences	$3,900,502$	$177,734$	$208,201$	$449,771$

A.2 COST ANALYSIS OF FL METHODS ON DIFFERENT CLIENT'S NUMBER

The communication and computational costs of different FL methods are illustrated in Table 4. For communication, the cost of FedAvg is much higher than that of FedNN and FT-Ensemble. The reason is that FedAvg requires multi-round communication based on the model, while both FedNN and FT-Ensemble require only one-round memorization-based communication. For computation, the cost of FT-Ensemble is linearly related to the number of nodes. It cannot be extended to practical applications because of the number of local models that need to be integrated for inference. In contrast, the cost of FedNN is only $1/60$ and $N/2$ of FedAvg$_1$ and FT-Ensemble, respectively. Considering many clients' limited communication bandwidth and computational resources, FedNN is a promising framework selection to save a lot of communication time and computational consumption.

A.3 THREAT MODEL AND DATASET CONSTRUCTION

Dataset Construction Given a parallel sentence pair $(\mathbf{x}_c, \mathbf{y}_c) \in \mathcal{D}_c$, the public NMT model generates the context representation $k = f_\theta(\mathbf{x}_c, y_{c,<t})$ in the last decoder layer at each timestep t. We concatenate k and the original sequence $\mathbf{x}_c$, and $y_{c,<t}$ into a new sequence $r_c = (k, \mathbf{x}_x, <\text{2tgt}>, y_{c,<t})$, which later becomes the final dataset: $\mathcal{R}_c = \{r_c\}^{|\mathcal{R}_c|}$.

[2]https://github.com/facebookresearch/fairseq/blob/main/ examples/translation/prepare-wmt14en2de.sh

[3]https://github.com/moses-smt/mosesdecoder

[4]https://github.com/mjpost/sacrebleu, with a configuration of 13a tokenizer, case-sensitiveness, and full punctuation

Table 4: The communication cost and computation cost of different methods, where "M, N, R and D" respectively represent the model size (414MB), number of client, rounds of communication (160) and the total size of all encrypted datastores (1785MB).

	Communication Cost (GB)					Computation Cost (K Step)				
	Compl.	**3**	**6**	**12**	**18**	**Compl.**	**3**	**6**	**12**	**18**
FedAvg	M×N×R×2	388.12	776.25	1552.50	3105.00	300	300	300	300	300
FT-Ensemble	M×N×(N+1)	4.85	16.98	63.07	138.27	10×N	30	60	120	180
FedNN	(D+M)×N	5.08	12.08	26.10	40.12	5	5	5	5	5

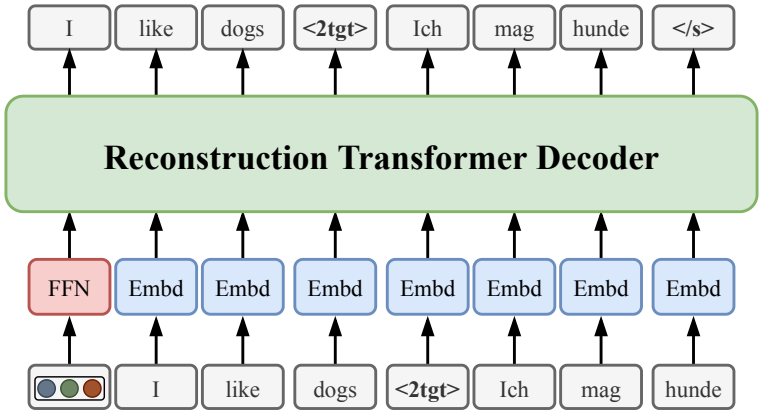

Figure 4: Threat Model.

Threat Model Design Our threat model contains a reconstructed transformer decoder. Its goal is to translate the original text and the translation corresponding to any given key from memorization. We first transform the first input token k to the same dimension as the word embedding using a linear layer, and then autoregressive perform left-to-right reconstruction modeling.

A.4 Impact of Public Model's Quality.

Since FedNN performs federated learning based on the Public Model, we investigate the impact of the Public Model's quality on performance in this section. We introduced WMT20 En-De data to train the Public Model, which contains 40 million parallel corpora, and conducted fast experiments in the Non-IID setting. From Figure 3, we can observe that as the quality of the Public Model improves, all baselines show better performance.

Table 5: Impact of Public Model's Quality. "△" refers to the improvement of the method compared with the model mentioned in Table 1.

Methods	IT		Law		Medical		WMT14		Clients Avg.		Global Avg.	
	BLEU	△	**BLEU**	△	**BLEU**	△	**BLEU**	△	**BLEU**	△	**BLEU**	△
Public Model	30.72	+4.10	38.69	+2.78	35.90	+5.63	29.77	+3.14	35.10	+4.17	33.77	+3.91
FedAvg$_1$	28.63	+0.37	58.32	+5.32	49.08	+3.18	16.05	+2.60	45.34	+2.95	38.02	+2.87
FedNN	38.24	+2.62	55.76	+0.19	50.65	+1.44	22.65	+0.36	48.22	+1.42	41.83	+1.16