This article has been accepted for publication in IEEE Access. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2024.3453068

IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2024.DOI

Federated Learning for Sentiment Analysis in Presence of Non-IID Data: Sensitivity of Deep Learning Models

DAVOUD GHOLAMIANGONABADI, KATARINA GROLINGER, (Senior Member, IEEE)
Western University, Department of Electrical and Computer Engineering, London, ON, N6A 5B9 CA (e-mail: {dgholam, kgroling}@uwo.ca)
Corresponding author: K. Grolinger (e-mail: kgroling@uwo.ca).

ABSTRACT In sentiment analysis, data are commonly distributed across many devices, and traditional machine learning requires transferring these data to a central location exposing data to security and privacy risks. Federated Learning (FL) avoids this transfer by training a model without requiring the clients/devices to share their local data; however, FL performance drops when data are not Independent and Identically Distributed (non-IID), such as when label distribution or data size vary across clients. Although techniques for non-IID data have been proposed primarily in the image domain, the sensitivity of various deep learning models to non-IID data needs to be examined. Consequently, this paper investigates the sensitivity of three dominant techniques in sentiment analysis, feed-forward neural networks, LSTMs, and transformers to common types of non-IID data, specifically data size and label imbalances. The scenarios were designed with increasing degrees of imbalance in terms of data size and label distribution to investigate gradual changes. The results revealed that label imbalance has a higher impact on accuracy than data size imbalance irrelevant of the algorithm. Overall, the transformer achieved the highest accuracy, and, while all models experienced a drop in accuracy with the increased label imbalance, this drop was smaller for the transformer, making it well suited for non-IID data.

INDEX TERMS Federated learning, non-IID, Deep neural networks, Transformer, Natural language processing, Sentiment analysis

I. INTRODUCTION

Natural language data are abundantly available in a wide array of sources such as web pages, social media, mobile applications, books, and newspapers. Devices are generating massive quantities of data: for example, in 2017, there were over 456,000 tweets sent on Twitter every minute [1]. To analyze these vast data, Natural Language Processing (NLP) plays a vital role in many domains such as psychology, marketing, and healthcare. In marketing, for example, sentiment analysis can assess the effectiveness of the company's branding strategy [2].

Machine Learning (ML) approaches have demonstrated great abilities in extracting valuable knowledge from natural language data [3]: common applications of ML in NLP are machine translation, summarization, named-entity recognition, query answering, and sentiment analysis [4]. Sentiment analysis aims to detect and understand people's sentiments or emotions on a topic expressed in a segment of text. By analyzing unstructured data from sources such as social media posts, customer feedback, and survey responses, sentiment analysis provides insights for a variety of applications including marketing, social media monitoring, opinion mining, business intelligence, and drug reviews [5]–[7].

In traditional ML, training is carried out on a centralized server or the cloud. However, NLP data are typically distributed across multiple locations, and stored in various databases or files on different devices; thus, they require transfer to the server or the cloud for ML. Although these centralized ML approaches have been greatly successful in many domains for diverse tasks, they result in increased network congestion, data transfer latencies, and scaling challenges, while also exposing data to security and privacy risk related to data sharing and transfer.

Federated Learning (FL) has emerged as an approach for addressing those challenges by enabling distributed ML training across multiple clients while allowing the clients to retain control of their local data. As clients do not share their local data, FL reduces the risk of private data disclosure

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4

This article has been accepted for publication in IEEE Access. This is the author's version which has not been fully edited and
content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2024.3453068

[8]–[10]. In recent years, FL has demonstrated successes in various domains including load forecasting [11], malware detection [12], smart agriculture [13], and healthcare [14].

FL demonstrated great results with Independent and Identically Distributed (IID) data; however, in real-world scenarios, particularly those involving extensive data spread across numerous clients, data are highly likely to be non-IID, which results in performance degradation. Examples of non-IID data include a different number of samples per device, skewed distribution of data labels across clients, concept shift, and others. It has been widely acknowledged that non-IID data have negative effects on FL and that handling non-IID is the core challenge that needs to be addressed for wider adoption of FL across a diversity of real-world applications [15]. Moreover, several techniques have been proposed for handling different dimensions of non-IID [15], [16]. However, research is needed to understand the sensitivity of deep learning algorithms to non-IID data and to understand the implications of varied degrees of imbalance on sentiment analysis tasks crucial for the selection of the algorithm and for leveraging FL in practice.

Consequently, this paper investigates the impact of non-IID data on the performance of deep learning models in sentiment analysis to gain insights into their behaviour and to provide guidance for sentiment analysis applications. Two common types of non-IID data are investigated, differences in the number of records per client (client size) and variations in label distribution among clients (label imbalance). Scenarios were designed with varied degrees of imbalance in both non-IID dimensions, including a concurrent imbalance in both. On the algorithm side, a baseline model, Feed Forward Neural Network (FFNN), along with two deep learning models dominant in sentiment analysis, Long Short-Term Memory (LSTM), and transformer, were considered. The main contributions of this work include:

- Quantifying the impact of non-IID data, commonly present in real-world applications, on FL performance. Although the challenges of non-IID data in FL have been recognized, to the best of our knowledge, the extent of their impact has not been previously investigated.
- Examining and comparing the sensitivity of common neural network architectures to various degrees of non-IID data in FL settings.
- Demonstrating through extensive experiments that label imbalance has a greater impact on accuracy than size imbalance in FL scenarios.
- Revealing that transformer models exhibit superior handling of non-IID data compared to LSTMs and FFNNs in FL environments.

The remainder of the paper is organized as follows: Section II reviews related work, Section III provides preliminaries, Section IV describes the methodology, and Section V presents results and discusses findings. Finally, Section VI concludes the paper.

II. RELATED WORK

It has been well established that non-IID data cause challenges for FL often significantly reducing accuracy and slowing convergence [17]. Therefore, several studies examined non-IID issues in FL and proposed techniques for addressing diverse non-IID dimensions. Hsieh et al. [18] examined the skewed distribution of data labels across devices/locations for the image classification and evaluated the presented approach on the mammals dataset from Flickr. Considering three FL algorithms, Gaia, FederatedAveraging, and DeepGradientCompression, they showed that skewed data labels are a fundamental and pervasive problem for decentralized learning.

Wang et al. [19] proposed an experience-driven control framework that intelligently chooses the client devices to participate in each round of FL to counterbalance the bias introduced by non-IID data and to speed up convergence. Using deep Q-learning, a mechanism was designed to learn device selection aiming to maximize a reward that encourages higher validation accuracy while minimizing communication rounds. Results showed that the number of communication rounds can be reduced by up to 49% on the MNIST dataset, 23% on FashionMNIST, and 42% on CIFAR-10, as compared to the FedAvg algorithm.

A Bayesian nonparametric framework for FL with neural networks was proposed by Yurochkin et al. [20]. They employed the Dirichlet distribution to generate unbalanced subsets in terms of data size and labels and applied the proposed approach to two image classification datasets, CIFAR and MNIST. On both datasets, their approach achieved comparable accuracy to federated averaging and Downpour SGD (D-SGD) algorithms but with fewer communication rounds.

Similarly, Tang et al. [16] also considered image datasets: they presented a novel decentralized parallel stochastic gradient descent algorithm (D^2) with the objective of achieving robustness under high data variance. Decentralized workers here only have the data pertaining to a subset of labels; in one scenario each worker has data from only one class from 16 classes, while in another scenario, each worker has data from two out of 10 classes. The workers exchange information with their neighbors connected through a graph. Empirical results showed that D^2 algorithm on image classification tasks outperforms the decentralized parallel stochastic gradient descent.

Zhao et al. [15] used Convolutional Neural Network (CNN) and FedAvg algorithm for a severely skewed dataset. To increase FedAvg performance with non-IID data, they proposed a data-sharing technique in which a restricted set of samples is shared globally across all edge devices. The IID setting was simulated by distributing the training dataset evenly among 10 clients, while for the non-IID setting, the dataset was partitioned to create two extreme cases: (a) 1-class non-IID, in which each client received a data partition from only one single class, and (b) 2-class non-IID, in which each client was randomly assigned partitions from two classes. The results of the experiments showed that the

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4

This article has been accepted for publication in IEEE Access. This is the author's version which has not been fully edited and
content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2024.3453068

accuracy on CIFAR-10 dataset increased by approximately 30% by sharing globally only 5% of data.

Studies discussed so far primarily address image classification tasks; in contrast, the study conducted by Chen et al. [21] specifically focuses on an NLP task. They designed a character-level recurrent neural network for learning Out-Of-Vocabulary (OOV) words under the FL setting. Here, the vocabulary includes common words and phrases that the Google keyboard (Gboard) recognizes and suggests to users during typing. OOV words are not initially in the vocabulary but need to be learned for better typing suggestions. The evaluation was carried out with the Reddit conversation corpus considering 492 million comments split between 763 thousand unique users. Their model was successful in learning the top 10^5 unique words, leading to improvement in the accuracy of word suggestions.

Zhu et al. [22] also considered text data; they investigated the federated TextCNN model for the intent classification problem and presented a differentially private FL technique by introducing sample-level privacy protection. On clients, for each batch, gradients are computed and applied to update the local parameters, and then the accumulated difference of parameter values is sent to the central server for cross-client aggregation. To protect privacy, Gaussian noise is added to the gradients before the parameters are updated. Results showed that the FL model performance depends on the sampling ratio (label distributions) among different classes.

A modular framework for assessing learning in federated environments, LEAF, was introduced by Caldas et al. [23]. LEAF contains federated open-source datasets for the evaluation of FL techniques, including FEMNIST, Sent140, Shakespeare, CelebA, and Reddit. Some of these datasets, such as Reddit, Sent140, and Shakespeare are text-based while others are images. Each dataset has keys that refer to particular devices/users, enabling the creation of clients and providing the users with the ability to assess their approaches for FL, meta-learning, and multi-task learning.

Several discussed studies [16], [18]–[20] focused on FL with non-IID data for the image-based applications while our work considers an NLP task. Other studies considered the NLP domain and proposed techniques for addressing non-IID data challenges [15], [21], [22] in FL or presented a framework for the evaluation [23]. In contrast, our work examines the sensitivity of deep learning architectures to varying degrees of non-IID characteristics in sentiment analysis, enhancing our understanding of architectures' behaviour with diverse non-IID characteristics.

III. PRELIMINARIES – FEDERATED LEARNING AND NEURAL NETWORK ARCHITECTURES

This section provides a short overview of the FL process and then presents three network architectures considered in this work: FFNN, LSTM, and transformer.

A. FEDERATED LEARNING

Federated Learning (FL) is a technique where multiple decentralized devices or servers collaboratively train a shared model while retaining all data locally, thereby mitigating privacy concerns associated with data sharing and network transmission. As seen in Figure 1, in FL, the server first initializes a global model and sends its copy to the selected devices (Step 1). Each device trains its own model using its local data (Step 2) and sends the updated model parameters (Step 3) to the server. Notice that only the model parameters are exchanged while the raw data remains local. The server aggregates the received model parameters (Step 4) and sends the new global model parameters to clients (Step 5) for the next round of training. The process is repeated until convergence, and finally, the trained model is deployed to all clients (Step 6) for inference.

The objective of FL is to minimize the loss:

$$\min_{w} L(w), \text{where } L(w) = \frac{1}{K}\sum_{k=1}^{K} \ell_k(w) \tag{1}$$

where K is the number of clients, w are the model weights, and ℓ_k is the local objective function of node k which describes how the model conforms to the dataset at node k.

Each client k, at each training step t, updates its local weights w_k based on its local data and gradient descent as follows:

$$w_k^t = w_k^{t-1} - \eta g_k^t \tag{2}$$

where η is the learning rate and g_k^{t-1} is the local gradient for client k at time step t.

The clients send the updated weights to the server for aggregation. FedAvg algorithm commonly used in FL, aggregates locally trained models into a global model as follows:

$$w^r = \sum_{k=1}^{K} \frac{n_k}{n} w_k^r \tag{3}$$

where w^r are the global model weights at round r, K is the number of clients, n_k is the number of samples at client k, n is the number of samples across all clients, and w_k^r are the weights from client k at round r.

B. NEURAL NETWORK ARCHITECTURES

This section provides a short overview of the three network architectures considered in this work: FFNN, LSTM, and transformer. FFNN is a simple network architecture widely applied across diverse fields [24], [25], consisting of an input layer, hidden layers, and an output layer. Information flows from the input layer through one or more hidden layers to the output layer.

Recurrent Neural Network (RNN) was specifically designed for handling sequential data; however, standard RNNs suffer from exploding and vanishing gradient problems [26]. LSTM was proposed to overcome these issues [27], enabling the retention of information for longer periods of time. As

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4

This article has been accepted for publication in IEEE Access. This is the author's version which has not been fully edited and
content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2024.3453068

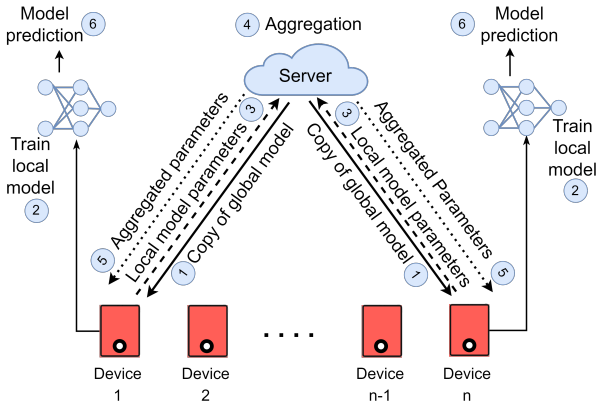

FIGURE 1: Federated learning process.

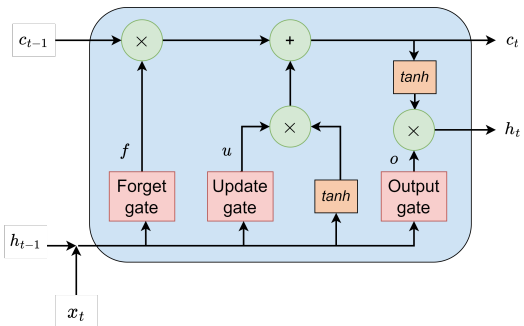

FIGURE 2: The structure of LSTM cell.

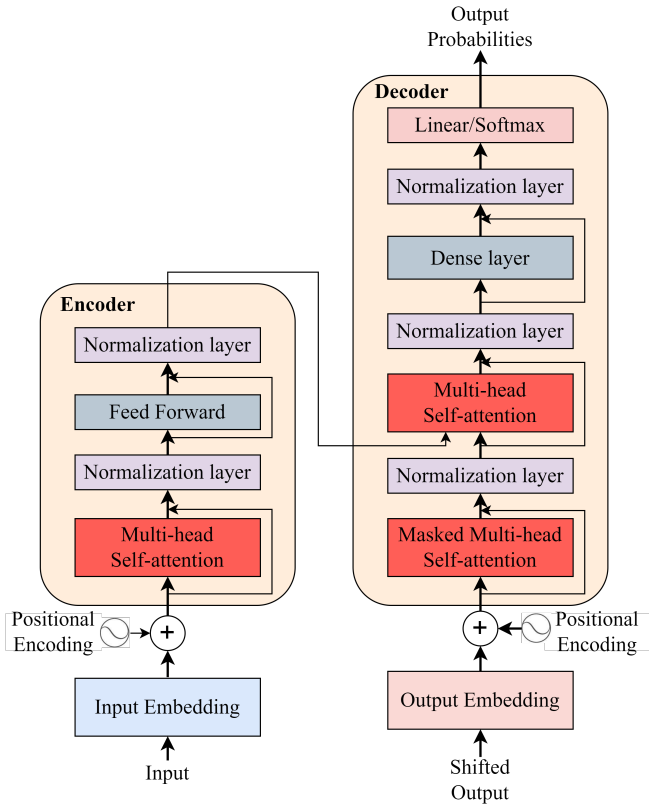

FIGURE 3: Transformer Architecture.

seen in Figure 2, the main components of the LSTM cell are the forget gate, update gate (input gate), and output gate. In LSTM, the output of the LSTM cell y_t can be expressed as follows:

$$y_t = f(x_t, h_{t-1}, C_{t-1}) \tag{4}$$

where f is a non-linear function, x_t is the input at the time step t, and h_{t-1} and C_{t-1} are the hidden state and cell state, respectively, from the previous time step.

In recent years, transformers have achieved remarkable success in the NLP field [28], including applications such as ChatGPT. Like RNN, the transformer is designed for sequential data, but, while RNN employs recurrent connections to capture temporal dependencies, the transformer incorporates the self-attention mechanism, which enables the model to concentrate on important parts of the input sequence. The self-attention mechanism can be formulated as follows:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{5}$$

where Q (Query), K (Key), and V (Value) are matrices derived from the input and d_k is the dimensionality of the key vectors [28].

As seen in Figure 3, the two main components of the transformer are an encoder and a decoder. Before entering the encoder block, the input text is converted into a vector through input embedding, and the positional information is added. The encoder simultaneously processes the inputs with the aid of the multi-head self-attention block. The normalization and feed-forward layers then produce an abstract representation of the complex input patterns, which is sent to the decoder. Similar to the encoder, the decoder takes advantage of multi-head self-attention, feed-forward, and normalization layers. In addition, the decoder employs a masked multi-head attention layer that maintains the auto-regressive characteristics of the transformer. The output embedding represents the vector representation of the generated tokens and shifted output refers to the practice of shifting the target sequence by one position to the right when feeding it into the decoder during training, ensuring the model is always trying to predict the next token in the sequence. After the decoder has completed its processing, probabilities are produced through $softmax$ as the output.

IV. METHODOLOGY

This paper investigates the effect of non-IID data on the sentiment analysis task in the FL setting with various deep learning architectures, which is essential due to the prevalence of non-IID data in the real-world. Understanding the performance and generalization ability of deep learning architectures under various degrees of data skew will provide the foundation for the wider adoption of FL across sentiment analysis applications.

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4

This article has been accepted for publication in IEEE Access. This is the author's version which has not been fully edited and
content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2024.3453068

To accomplish the objective of quantifying the effect of non-IID data on FL-based sentiment analysis, three base architectures were considered: FFNN, LSTM, and transformer. The FFNN model is a relatively simple model included in the examination as a baseline. LSTM was selected for its ability to capture long-term temporal dependencies in sequential data, its effectiveness in mitigating the vanishing/exploding gradient problem inherent in vanilla RNNs, and its use in sentiment analysis tasks [29]. Finally, the transformer architecture was chosen due to its remarkable successes across a range of NLP tasks, including its recent prevalence in sentiment analysis [30]. Although generative models can be applied for sentiment analysis, their computational requirements make it impractical or even impossible to deploy them in an FL setting on devices such as smartphones; thus, we do not consider them in our study.

The task of sentiment analysis in general aims to determine if the emotional tone of the text segment is positive or negative. In the FL setting, ML models for this task can be largely affected by the differences among people: for example, some people are overall more positive than others. The analysis was carried out on the Sentiment140 Twitter dataset [31] which contains 1.6M tweets on general topics, equally divided into positive and negative tweets. This dataset is then manipulated to generate different degrees of skew.

The overall process for sentiment analysis with different architectures is shown in Figure 4 while the details of data preprocessing and considered deep learning models are provided in the following subsections. Next, Scenario Design subsection describes the strategy used to create scenarios evaluating the model behavior in the presence of non-IID data.

A. DATA PREPROCESSING

Text segments (tweets) are first preprocessed to transform them into a form suitable for the considered deep learning models; specifically, the tweets are transformed into numeric vectors, which are then used for training the deep learning models. Figure 4 includes the six main preprocessing steps. First, emojis in the tweets are replaced with meaningful words: for example, the emoji ":-)" is substituted with the word "smile.". Next, all words in the tweets are converted to lowercase, and contractions such as "I'm" are expanded to their full forms, such as "I am.". Tags, punctuation, special characters, and numbers are removed from the tweets as they can be disregarded without altering the meaning. Tokenization is then applied to the tweets to split them into word segments and enable subsequent text analysis. Following this, the stop words are removed. It is important to notice that the words "no, not, none", often considered a stop word, are not eliminated from the tweets during this process as it significantly impacts the sentence meaning.

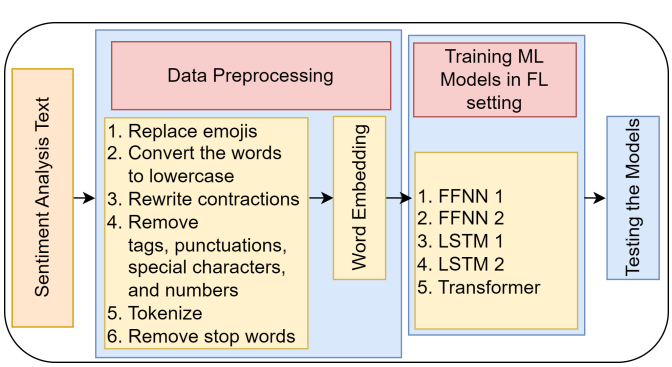

FIGURE 4: The overall process of sentiment analysis.

After this, the text is ready to be converted into a numeric form. A mapping function is applied to tweets to convert each word to a unique integer and, to make all the tweets' representations the same length, zero padding is applied. These vectors are transformed into real-valued vectors using the embedding layer which is the first layer of the neural network. Although the embedding layer is often considered a part of the network itself, here it is described in the preprocessing as it is common for all considered architectures.

B. MODEL ARCHITECTURES

This section presents an overview of the three primary network architectures: FFNN, LSTM, and transformer, and their variants used in the experiments.

All architectures were trained using the Federated Averaging (FedAvg) which is a common technique in FL for aggregating parameters from the clients [32]. In FedAvg, the model parameters obtained from the individual clients are aggregated through weighted averaging. Thus, only the parameters are exchanged between the server and clients which contributes to data privacy and security. For all architectures, an embedding layer described in Subsection IV-A creates word embeddings. Also, for all architectures, the number of inputs is equal to the number of features, and the number of outputs corresponds to the number of classes. As we are considering positive and negative tweets, there are two classes.

1) Feed Forward Neural Network (FFNN)

Two FFNN variants are considered in order to investigate the impact of the network size on model accuracy in the presence of non-IID training data.

- **FFNN1:** This FFNN architecture consists of three layers; an embedding, a flattening, and a dense layer. The flattening layer gets the output from the embedding layer and converts it into a one-dimensional vector, allowing it to be passed into the subsequent dense layer with two output neurons. This dense layer with *softmax* activation function generates a probability distribution over the two output classes.
- **FFNN2:** This FFNN architecture consists of four layers; embedding, flattening, and two dense layers. This architecture is similar to the previous one, with an additional dense layer with 64 neurons after the flattening layer to improve model capacity.

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4

This article has been accepted for publication in IEEE Access. This is the author's version which has not been fully edited and
content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2024.3453068

2) Long Short-Term Memory (LSTM)

As with FFNN, two LSTM architecture variants are considered to examine the impact of the network size on the network sensitivity to non-IID data in FL.

- **LSTM1:** This model consists of an embedding layer, flattening layer, LSTM layer, and dense layer. The LSTM layer with 32 hidden units enables the model to capture temporal (sequential) dependencies in the data. The output from the embedding layer is flattened into a one-dimensional vector with the flattening layer, and after the LSTM layer, a dense layer with two output units and *softmax* activation generates output probabilities.
- **LSTM2:** This model is similar to the LSTM1, but with an addition of a dense layer with 64 neurons after the LSTM layer. This dense layer provides additional capacity for non-linear transformations.

3) Transformer

Figure 5 shows the architecture of the transformer employed in this study. The original transformer consists of an encoder and decoder (Figure 3), whereas the transformer used here only employs the encoder part. The encoder of a transformer model is capable of extracting representations of the input sequence needed for sentiment detection. Since sentiment detection is a classification task, there is no need for sequential decoding to create sequences as done in tasks such as machine translation or text generation. Using only the encoder, the computational complexity is also reduced.

The input sequence is embedded along with positional information and passed to the encoder block. The self-attention mechanisms in the encoder block and the positional encoding are the same as in the original transformer. As seen in Figure 5, the transformer includes two encoder blocks, a global average-pooling layer, dropout, dense, dropout, and dense (output) layers. The global average pooling layer is added as it helps summarize the encoded information from the preceding encoder blocks into a single representation while also reducing dimensionality. In each encoder block, there are multi-head attention, dropout, normalization, dropout, and dense layers.

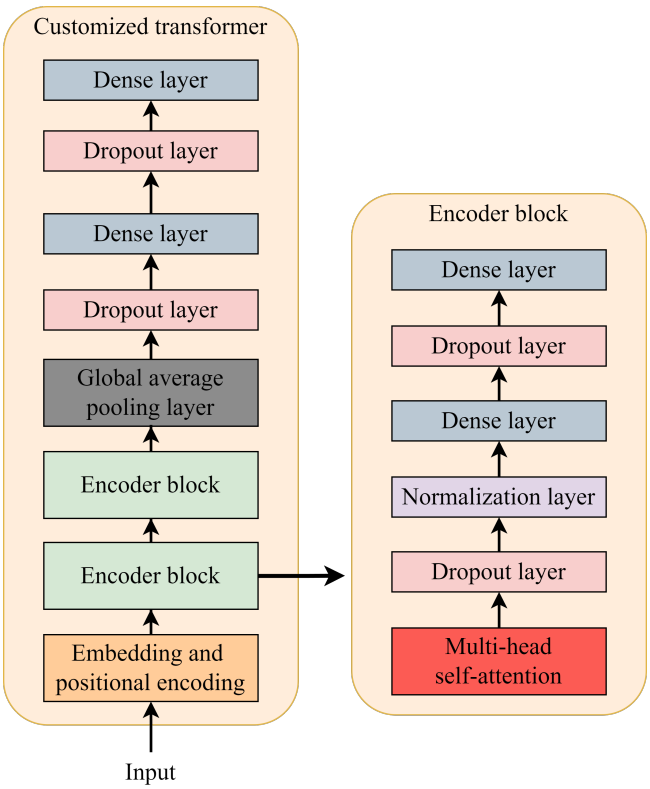

FIGURE 5: Architecture of the transformer.

C. SCENARIO DESIGN

FL commonly encounters statistical heterogeneity, where data are not independent and identically distributed (non-IID). This means that the distribution of data among different sources or clients may differ, or the distribution of data from a single client may change over time [33]. Kairouz et al. [33] highlighted five ways in which data can be non-IID: feature distribution skew (covariate shift), label distribution skew (prior probability shift), concept drift, concept shift, and quantity skew (unbalancedness). Label distribution skew and quantity skew are common among different clients in FL, therefore, this study focuses on those two types of non-IID data [18], [34].

Label distribution skew refers to a situation where the distribution of labels or target variables in the training data differs across the devices or clients. One instance of label distribution skew can be observed in the use of positive/negative messages, when certain individuals consistently generate more positive messages, while others predominantly produce negative ones. It has been shown that label distribution skew can have a significant impact on ML models, and can lead to biased predictions and sub-optimal performance [35].

Quantity skew refers to the uneven distribution or imbalance in the amount of data contributed by different clients in a decentralized learning or FL setup. However, due to variations in user populations, data collection capabilities, or other factors, some clients may have more data available to contribute than others. Clients with larger datasets may have a large influence on the model's training [36], potentially leading to biased results.

Here, we use the term *global dataset* to refer to the training dataset that encompasses data from all clients participating in the FL system. This dataset is balanced in terms of labels, meaning that each class contains an equal number of samples. The term *local datasets* refers to each client's dataset and can be balanced or imbalanced in terms of labels. In other words, the number of positive and negative records for each client can be equal or unequal.

Table 1 shows the overview of the 36 scenarios designed to examine the impact of imbalance in terms of labels and data quantity on the model performance in the FL setting. The rows represent the variations in terms of data size while columns consider different unbalance levels in terms of label

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4

This article has been accepted for publication in IEEE Access. This is the author's version which has not been fully edited and
content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2024.3453068

TABLE 1: Scenarios for the examination of non-IID impact on FL models.

		Label Distribution (L)					
	Size (S)	50-50 (L50)	45-55 (L45)	35-65 (L35)	25-75 (L25)	15-85 (L15)	5-95 (L5)
Size (Equal)	S0	S0L50	S0L45	S0L35	S0L25	S0L15	S0L5
Size (Unequal)	S10	S10L50	S10L45	S10L35	S10L25	S10L15	S10L5
	S30	S 30L50	S 30L45	S 30L35	S 30L25	S 30L15	S 30L5
	S50	S 50L50	S 50L45	S 50L35	S 50L25	S 50L15	S 50L5
	S70	S 70L50	S 70L45	S 70L35	S 70L25	S 70L15	S 70L5
	S90	S 90L50	S 90L45	S 90L35	S 90L25	S 90L15	S 90L5

distribution.

When considering a different number of samples across clients, the objective is to examine the impact of quantity skew on FL performance. In the table, the size of the local dataset is denoted with S, and $S0$ indicating that all clients have the same number of samples. Assuming N is the number of clients, the training dataset of size D is split equally between these clients resulting in each client having D/N samples.

The number following the symbol S indicates the degree of quantity skew. For example, in scenario $S10$, the number of records for each client is in the range $\frac{D}{N} \pm 10\%$, determined using a uniform distribution. This means the number of records for each client is $\pm 10\%$ from the number of records per client in the dataset balanced in terms of size. In the $S10$ scenario, each client can have a number of records between $0.9 \times \frac{D}{N}$ and $1.1 \times \frac{D}{N}$ while in $S90$ scenario, each client can have a number of records between $0.1 \times \frac{D}{N}$ and $1.9 \times \frac{D}{N}$.

Columns in Table 1 represent varying degrees of label distribution skew. Although the global dataset in this study is balanced, the local dataset can be balanced or imbalanced in terms of labels. In the table, the label distribution is indicated by L. Scenario $L50$ indicates that the local datasets (each client dataset) are balanced in terms of labels: 50% of records on each client are positive and 50% are negative. On the other hand, when L is not 50, it implies that the local data are imbalanced in terms of labels. For instance, in $L35$, the label distribution for all clients is $35-65\%$, which means, for each client, exactly 35% of records are negative and 65% positive records, or the other way around.

Since the sentiment analysis is a binary classification task and the dataset contains negative and positive records, the number of positive records may be higher than the number of negative records for some clients, and vice versa for other clients. If L is close to 50, it indicates that the client's dataset is close to being balanced. For example, if a scenario is $L5$, the datasets on clients are more imbalanced in terms of labels compared to those in $L45$ scenario. The positive and negative records for each client have been selected randomly without substitution.

Assessing and characterizing the impact of the concurrent presence of an imbalance in terms of size and labels is crucial in order to examine their compounded effect. In real-world scenarios, it is realistic to expect imbalances in both the size and label distribution among clients. As shown in Table 1, combinations of imbalances are examined. Consider $S30L35$ scenario: it has the number of records for each client in the range $\frac{D}{N} \pm 30\%$ while $L35$ indicates that the clients' data are imbalanced, and for some clients, 35% of records are negative and 65% are positive, whereas, for others, 35% are positive and 65% are negative. In Table 1, when we move from left to right, the label distribution imbalance is increased, while moving through rows from the top to the bottom, the difference between clients in terms of number of records is increasing.

In total, $6 * 6 = 36$ different scenarios examine the effect of non-IID data. Each of the 36 scenarios is examined with five different neural network architectures (as described in Subsection IV-B), including two FFNNs, two LSTMs, and one transformer, to investigate the sensitivity of different architectures to non-IID data. The performance of the FL approaches is compared to traditional centralized ML (without FL) where the *global dataset* containing all data together is used for training each of the five architectures in a traditional (centralized) manner.

V. EVALUATION

This section begins by introducing the dataset and experiments, followed by the results and a discussion of the findings.

A. DATASET AND EXPERIMENTS

The experiments were carried out with Sentiment140 dataset [31], a balanced dataset containing 1.6 million tweets with their labels: 800K positive tweets and 800K negative tweets. The dataset contains six features including target label, id, date, flag, user, and text. For sentiment analysis in this study, the target label and text features are used. The text is the tweet itself while the target label indicates the target sentiment, positive or negative.

Stratified sampling was applied to split the dataset into 70% for training and 30% for testing ensuring the balance in terms of labels for both the training and test datasets. This equal representation of different labels provides a fair and unbiased evaluation of the model's performance. The training dataset, before it is divided and distributed to clients is referred to as the *global dataset*. From this dataset, subsets for clients are created with different degrees of imbalance

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4

This article has been accepted for publication in IEEE Access. This is the author's version which has not been fully edited and
content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2024.3453068

as described in Subsection IV-C and Table 1 to quantify the effect of non-IID data on the model performance under the federated setting. For all experiments, the trained models are evaluated on the test dataset.

All FL experiments considered 250 clients and in each round 20 clients were randomly selected for training. The number of FL rounds for training was 100, the number of epochs for local training in each round was 10, the learning rate was 0.05, and the batch size was 32. To prevent overfitting and enhance the generalization, early stopping was used to halt the training process when the model's performance on the validation set stops improving. The FL settings can vary among studies as they explore different configurations and tune the models to suit specific applications and experimental setups; however, in the context of our work, these settings are fixed for all experiments as the focus is on examining the impact of non-IID data, and not on finding the best hyper-parameters for FL or the ML models. Nevertheless, two FFNN and two LSTM variants with different numbers of layers and neurons (key hyperparameters) were examined to evaluate the impact of network complexity on the FL performance in the presence of non-IID data.

The analysis of the preprocessed tweets showed that the majority of tweets have fewer than 30 tokens; therefore, the embedding layer was set to a maximum input sequence length of 30. The word embedding technique represented each word with an 8-dimensional vector.

The experiments were implemented in Python and for training deep neural networks in the federated setting, the *tensorflow-federated* library was utilized. The experiments were conducted on a computer running Ubuntu operating system, equipped with an AMD Ryzen processor 4.20 GHz, 128 GB DIMM RAM, and four NVIDIA graphics cards. To handle the computational demands of training the proposed deep learning models, GPU acceleration was employed.

B. RESULTS

This section first presents results for FFNNs, LSTMs, and the transformer, followed by a comparison among these deep learning models.

1) FFNN

Here, results pertaining to FFNN1 and FFNN2 architectures described in Subsection IV-B are presented and analyzed. Tables 2 and 3 show the results of FFNN1 and FFNN2 respectively for FL scenarios presented in Table 1, together with results for traditional centralized ML. For both, FFNN1 and FFNN2, the best accuracy in the federated setting is achieved in scenario $S0L50$: 75.98% for FFNN1 and 77.11% for FFNN2. This is to be expected as scenario $S0L50$ is balanced in terms of both labels as well as data size.

The lowest value observed for FFNN1 is 72.64% which was achieved for $S50L5$ configuration. On the other hand, the lowest accuracy for FFNN2 was 74.35% which was recorded for scenario $S90L5$. As scenario $S90L5$ is the most imbalanced, it is to be expected that it will lead to the lowest accuracy. Although for FFNN1, the most imbalanced scenario did not lead to the lowest accuracy, its value of 72.69% is very close to the lowest value of 72.64%. This could be explained by the randomness in the training process; for example, the random selection of clients in training rounds could lead to some variations in performance.

Comparing the results of FL to the traditional centralized training, which achieved 78.53% for FFNN1 and 78.65% accuracy for FFNN2, it can be observed that balanced ($S0L50$) or almost balanced scenarios (e.g., $S10L50$, $S0L45$, and $S10L45$) yield results slightly lower than traditional training. Comparing averages and standard deviations between the two tables, it can be noticed that FFNN2 achieves better performance than FFNN1 in terms of averages and standard deviations across the scenarios with the same imbalance in terms of size (averages for rows) as well as across the scenarios with the same imbalance in terms of labels (averages for columns). More parameters in FFNN2 than in FFNN1 allow FFNN2 to better capture the complexities of sentiment detection.

Figures 6 and 7 show the same data as Tables 2 and 3, but graphical representation allows for observations of trends. Within those plots, accuracy values are shown for extreme scenarios $S10$ and $S90$, while other numbers are omitted for clarity. It can be observed that changes in the label distribution (indicated by L values) have a more noticeable effect than changes in data quantity (S values). The drop in accuracy is especially pronounced when transitioning from a $L25$ label distribution to a $L5$ distribution. In contrast, for

TABLE 2: FFNN1 - Accuracy in percentages for FL scenarios and traditional centralized training.

Federated learning	L50	L45	L35	L25	L15	L5	Avg.	Std.
S0	**75.98**	75.92	75.71	75.49	75.03	72.92	75.18	1.16
S10	75.91	75.83	75.67	75.47	75.04	72.88	75.13	1.15
S30	75.64	75.88	75.6	75.38	74.92	72.71	75.02	1.18
S50	75.66	75.62	75.75	75.28	74.78	**72.64**	74.96	1.19
S70	75.77	75.65	75.64	75.25	74.98	72.73	75.00	1.15
S90	75.83	75.62	75.67	75.3	74.66	72.69	74.96	1.19
Avg.	75.8	75.75	75.67	75.36	74.9	72.76		
Std.	0.14	0.14	0.05	0.1	0.15	0.11		

Traditional centralized training: 78.53

TABLE 3: FFNN2 - Accuracy in percentages for FL scenarios and traditional centralized training.

Federated learning	L50	L45	L35	L25	L15	L5	Avg.	Std.
S0	**77.11**	77.01	76.99	76.92	76.61	74.59	76.54	0.97
S10	77.03	76.98	76.94	76.89	76.58	74.48	76.48	0.99
S30	76.94	76.94	76.84	76.84	76.56	74.5	76.44	0.96
S50	76.93	76.91	76.88	76.78	76.47	74.58	76.43	0.92
S70	76.83	76.93	76.86	76.78	76.5	74.49	76.4	0.95
S90	76.91	76.82	76.83	76.65	76.48	**74.35**	76.34	0.99
Avg.	76.96	76.93	76.89	76.81	76.53	74.5		
Std.	0.1	0.07	0.06	0.1	0.06	0.09		

Traditional centralized training: 78.65

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4

This article has been accepted for publication in IEEE Access. This is the author's version which has not been fully edited and
content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2024.3453068

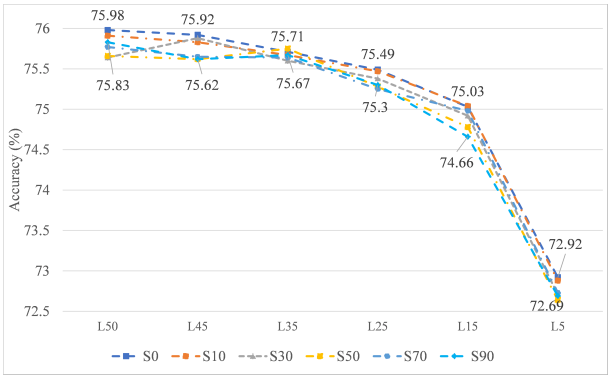

FIGURE 6: FFNN1: Comparison of the network behavior for different degrees of non-IID data in terms of quantity and label distributions.

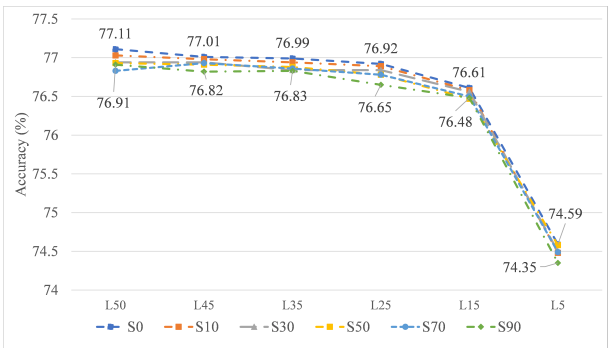

FIGURE 7: FFNN2: Comparison of the network behavior for different degrees of non-IID data in terms of quantity and label distributions.

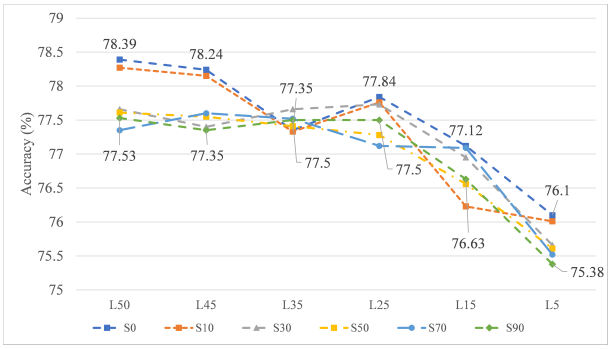

FIGURE 8: LSTM1: Comparison of the network behavior for different degrees of non-IID data in terms of quantity and label distributions.

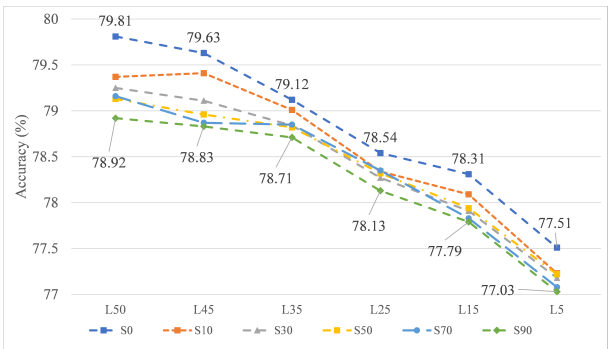

FIGURE 9: LSTM2: Comparison of the network behavior for different degrees of non-IID data in terms of quantity and label distributions.

both networks, changes in the quantity distribution have very little impact on the accuracy as evident from the proximity of the lines representing scenarios $S10$ to $S90$.

2) LSTM

Moving to the LSTM evaluation, Figures 8 and 9 show the results for LSTM1 and LSTM2, respectively. For FFNN, tables were provided, but we omitted tables for LSTM as the same information is shown in Figures 8 and 9. Again, within the plot, only the accuracy numbers for extreme scenario $S0$ and $S90$ are shown for clarity. Similarly to FFNNs, Figures 6 and 7, for both LSTM networks, Figures 8 and 9, there is a downward trend observed when imbalance in terms of labels increase (moving from $L50$ towards $L5$).

For LSTM1, the line corresponding to $S0$ is mostly above the remaining lines indicating that this balanced scenario in terms of data quantity on average achieved better accuracy than imbalanced scenarios. The accuracies of $S0L50$ and $S10L50$ are relatively close to each other, while there is a distinct gap between these scenarios and the others. However, when the label distribution changes towards $L35$, the gap between scenarios diminishes, and in the $L35$ scenario, all results become more closely aligned. When the label distribution changes to $L5$, there is a drop in accuracy, and the gap between the $S0$/$S10$ and other scenarios appears.

For LSTM2, there is more difference among scenarios with different levels of data size imbalance (lines further apart) than for LSTM1 or FFNNs: more balanced scenarios in terms of data size archive better accuracy, and the balanced scenario $S0$ achieves the highest accuracy. Again, the effect of the label distribution has a higher impact on the model performance compared to the size differences among clients.

The traditional centralized training achieved an accuracy of 80.01% for LSTM1 and 80.05% for LSTM2. In the federated setting, the best accuracies in the federated setting, 78.39% for LSTM1 and 79.81% for LSTM2, are close to the traditional training.

3) Transformer

Figure 10 shows the results for the transformer, for all FL scenarios. The same as in the previous figures, only numbers for $S0$ and $S90$ are shown within the plot for clarity.

Similar to previous models, FFNNs and LSTMs, there is a drop in accuracy when imbalances increase in terms of labels – moving from $L50$ to $L5$. $S0$ scenario mostly achieves higher accuracy than other imbalanced scenarios in terms of data size. Once again, it is obvious that the effect of label distribution is more prominent than the impact of data sizes.

With traditional centralized training, the transformer

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4

This article has been accepted for publication in IEEE Access. This is the author's version which has not been fully edited and
content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2024.3453068

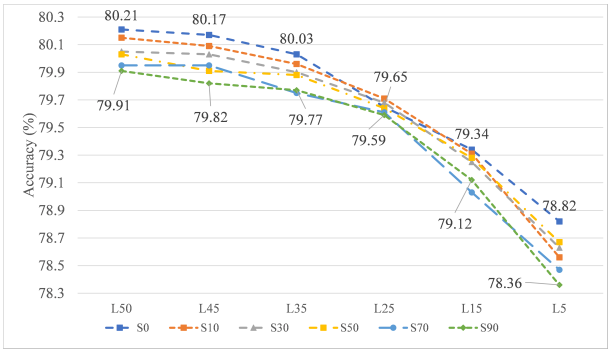

FIGURE 10: Transformer: Comparison of the network behavior for different degrees of non-IID data in terms of quantity and label distributions.

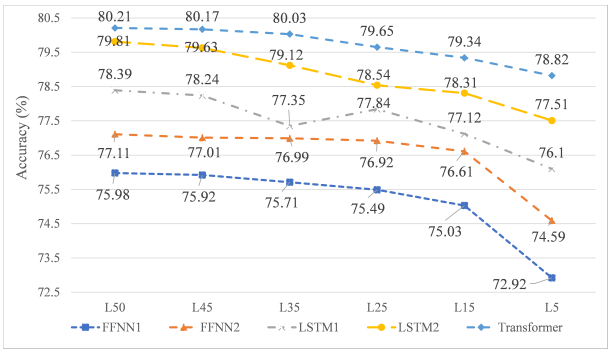

FIGURE 11: Comparison among models: clients balanced in terms of size ($S0$) but imbalanced in terms of labels.

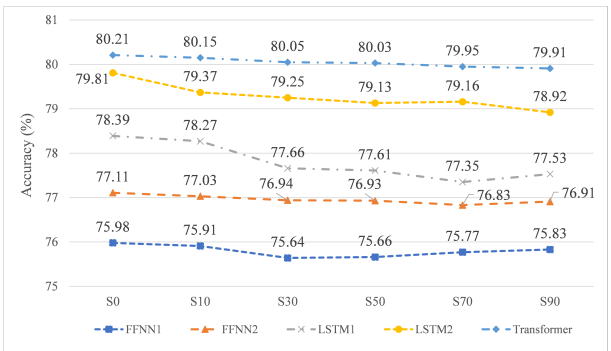

FIGURE 12: Comparison among models: clients imbalanced in terms of size but balanced in terms of labels ($L50$).

achieved an accuracy of 80.37%. FL, as expected, achieved the best accuracy of 80.21% for the balanced dataset, scenario $S0L50$, which is very close to the accuracy of traditional centralized training with the advantage of FL facilitating security and privacy. Moreover, slightly imbalanced scenarios, such as $S0L45$ or $S10L50$ achieved results very close to the balanced scenario.

4) Comparison among deep learning models

This section contrasts different deep learning architectures, FFNN, LSTM, and the transformer to compare their sensitivity to non-IID data. Figure 11 depicts the results of the five architectures for varied label distribution while keeping the balance in terms of data sizes ($S0$). Through all scenarios, the transformer performs the best, followed closely by LSTM2. In contrast, FFNN1 performs worse than the other models. Overall, more complex models perform better than their simple counterparts: FFNN2 is better than FFNN1, and LSTM2 is better than LSTM1.

As we move from a $L50$ to $L5$, the increase of imbalance in terms of labels results in a drop of accuracy for all models. Notably, transitioning from $L15$ to $L5$ results in a sharper drop of accuracy for most models than when transitioning between other L scenarios.

Figure 11 considered clients balanced in terms of data size ($S0$), whereas Figure 12 depicts results for clients balanced in terms of labels ($L50$). Again, the transformer achieves better accuracy than the other models through all scenarios, and FFNN1 shows the lowest accuracy. Increasing imbalance in terms of size from $S0$ to $S90$ results in a mostly negligible decrease in accuracy indicating that the imbalance in terms of the size has a minimal impact. One possible reason for this pattern is that clients with a larger data size compensate for the impact of other clients with smaller datasets.

To examine the relative change in accuracy as the label imbalance increases, Figure 13 shows the accuracy reduction with respect to label-balanced scenario $L50$ considering only scenarios balanced in terms of size ($S0$). Here accuracy reduction for $L45/L50$ is calculated as $(accuracy_{L50} - accuracy_{L45})/accuracy_{L50} * 100\%$. It can be observed that for $L45$ there is hardly any change in accuracy compared to $L50$ for all models: accuracies for $L45$ are very close to those for $L50$. However, when tipping the scale toward the higher label imbalance, the accuracy decline for all models is indicated by the increase in accuracy reduction. For severely imbalanced data $L5$, the transformer experienced a smaller drop in accuracy than the other models.

Figure 13 considers only scenarios balanced in terms of size ($S0$), while Figure 14 includes all imbalance levels in terms of size. The boxes in the plot indicate the range of the accuracy reduction for all levels of size imbalance: for example, the box for $L45/L50$, FFNN1, shows the range of values for $S0$, $S10$, to $S90$. A distinctive trend for each deep learning model shows that the accuracy reduction increases with an increase in imbalance. Again, the transformer is less affected by an imbalance in labels than the remaining models.

C. DISCUSSION

The main objective of this work is to examine the sensitivity of the FL models in sentiment analysis to data imbalance in terms of size and labels. To achieve this, we created scenarios involving varied degrees of size and label imbalance and consequently, evaluated the three architectures, FFNN, LSTM, and transformer. Across all deep learning models, the imbalance in terms of label distribution has a much higher impact on the accuracy than the imbalance in terms of data

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4

This article has been accepted for publication in IEEE Access. This is the author's version which has not been fully edited and
content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2024.3453068

size. The size imbalance has small effects even with large imbalances as seen from Figures 6 to 10 and 12. On the other hand, the impact of label imbalance changes with the increase of imbalance: for small imbalances ($L45$ and $L35$), the reductions in accuracy are minimal but there is a sharp drop in accuracy when moving from $L15$ to $L5$ as seen from Figures 6 to 10 and 11.

Overall, the transformer achieved higher accuracy than FFNNs and LSTMs for almost all degree of size and label imbalance. Moreover, the transformer was less sensitive to such imbalances than other approaches as demonstrated through a lower drop of the accuracy with the increase of imbalance seen in Figures 13 and 14.

To compare the behaviour of the algorithms in the FL setting with the traditional ML, Figure 15 shows the accuracy of four models: traditional centralized training, FL with balanced data $S0L50$, FL with slight imbalance $S10L45$, and very imbalanced FL $S90L5$. Overall, both, the transformer and LSTM2 demonstrate excellence in sentiment analysis; however, the transformer exhibits a slightly better performance in all settings, traditional and federated.

Overall, the results show that the transformer is better suited for the FL setting than FFNNs or LSTMs. Moreover, in the FL setting, the transformer archives performance very similar to traditional centralized ML even with some imbalance in terms of data size and labels. However, FL has the tremendous advantage of not requiring clients to share their local data, therefore reducing privacy and security risks.

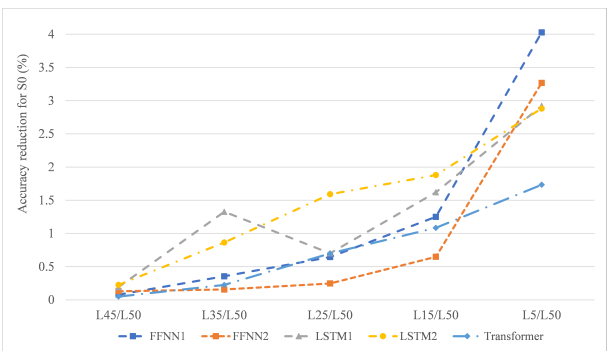

FIGURE 13: Comparison among models in terms of relative change for clients balanced in terms of size ($S0$).

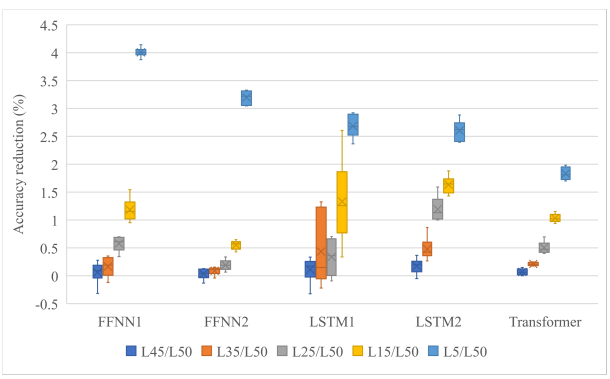

FIGURE 14: Comparison among models in terms of relative change including a range of size imbalances.

VI. CONCLUSION

Federated learning in sentiment analysis and many other domains has the potential to address several challenges posed by traditional centralized ML including security and privacy risks associated with sharing and transferring local data to a centralized location. However, it is well known that FL performance degrades in the presence of non-IID data.

This study investigates the sensitivity of ML algorithms to data imbalances in the sentiment analysis task with the objective of understanding the algorithms' behaviors in the FL setting with imbalanced data. Three sentiment analysis algorithms – FFNN, LSTM, and transformer – we investigated considering the impact of data size and label imbalances, which are common types of non-IID data. To investigate the impact of the degree of imbalance, scenarios were designed considering different levels of imbalance in terms of data size and labels. The findings from Figures 6 to 10 reveal that the label imbalance has a much higher impact on the model accuracy than the data size imbalance irrelevant of the algorithm. From Figures 11, 12, and 15, we can draw this conclusion that the transformer achieved higher accuracy than the other algorithms for almost all degrees of imbalance. Although all algorithms experienced a drop in accuracy with the increase of label imbalance, for large imbalances, the drop was the lowest for the transformer based on Figures 13 and 14. This makes the transformer well suited for FL-based sentiment analysis with higher resiliency to non-IID data than FFNNs and LSTMs. In domains where explanations for decisions are required, such as in many medical tasks, deep learning-based solutions may not provide the desired outcome and might need to be extended with explainability techniques.

Future work will explore the presence of other types of non-IID data in sentiment analysis and investigate the sen-

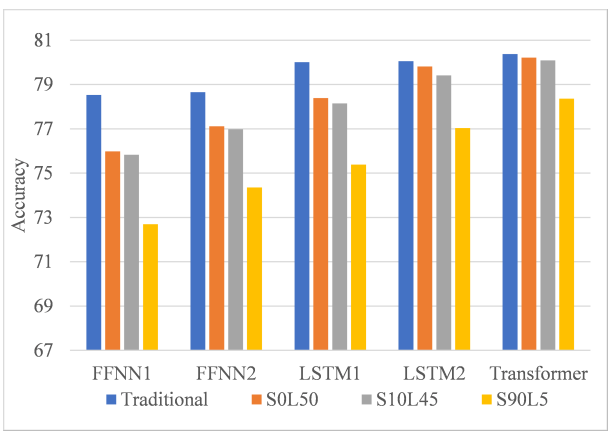

FIGURE 15: Comparison of traditional centralized machine learning with FL in the presence of balanced and imbalance data.

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4

This article has been accepted for publication in IEEE Access. This is the author's version which has not been fully edited and
content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2024.3453068

sitivity of the algorithms to those variations. Moreover, the applicability of image domain non-IID handling techniques for sentiment analysis will be examined.

ACKNOWLEDGMENTS

This research has been supported by NSERC under grant RGPIN-2018-06222.

REFERENCES

[1] Q. Hou, M. Han, F. Qu, and J. S. He, "Understanding social media beyond text: a reliable practice on twitter," Computational Social Networks, vol. 8, no. 4, pp. 1–20, 2021.
[2] F. Benedetto and A. Tedeschi, "Big data sentiment analysis for brand monitoring in social media streams by cloud computing," Sentiment Analysis and Ontology Engineering: An Environment of Computational Intelligence, pp. 341–377, 2016.
[3] A. J. Trappey, C. V. Trappey, J.-L. Wu, and J. W. Wang, "Intelligent compilation of patent summaries using machine learning and natural language processing techniques," Advanced Engineering Informatics, vol. 43, p. 101027, 2020.
[4] D. W. Otter, J. R. Medina, and J. K. Kalita, "A survey of the usages of deep learning for natural language processing," IEEE Transactions on Neural Networks and Learning Systems, vol. 32, no. 2, pp. 604–624, 2020.
[5] A. Van Looy, "Sentiment analysis and opinion mining (business intelligence 1)," Springer Texts in Business and Economics, pp. 147–163, 2022.
[6] F. K. Sufi and I. Khalil, "Automated disaster monitoring from social media posts using AI-based location intelligence and sentiment analysis," IEEE Transactions on Computational Social Systems, pp. 1–11, 2022.
[7] C. Colón-Ruiz and I. Segura-Bedmar, "Comparing deep learning architectures for sentiment analysis on drug reviews," Journal of Biomedical Informatics, vol. 110, p. 103539, 2020.
[8] J. C. Jiang, B. Kantarci, S. Oktug, and T. Soyata, "Federated learning in smart city sensing: Challenges and opportunities," Sensors, vol. 20, no. 21, p. 6230, 2020.
[9] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," vol. 2, (Austin, TX, USA), pp. 429–450, 2020.
[10] N. Rieke, J. Hancox, W. Li, F. Milletari, H. R. Roth, S. Albarqouni, S. Bakas, M. N. Galtier, B. A. Landman, K. Maier-Hein, et al., "The future of digital health with federated learning," Nature Partner Journal Digital Medicine, vol. 3, no. 1, p. 119, 2020.
[11] M. N. Fekri, K. Grolinger, and S. Mir, "Asynchronous adaptive federated learning for distributed load forecasting with smart meter data," International Journal of Electrical Power & Energy Systems, vol. 153, p. 109285, 2023.
[12] Y.-W. Chang, H.-Y. Chen, C. Han, T. Morikawa, T. Takahashi, and T.-N. Lin, "FINISH: efficient and scalable nmf-based federated learning for detecting malware activities," IEEE Transactions on Emerging Topics in Computing, vol. 11, no. 4, 2023.
[13] D. Vimalajeewa, C. Kulatunga, D. P. Berry, and S. Balasubramaniam, "A service-based joint model used for distributed learning: Application for smart agriculture," IEEE Transactions on Emerging Topics in Computing, vol. 10, no. 2, pp. 838–854, 2021.
[14] A. P. Kalapaaking, I. Khalil, and X. Yi, "Blockchain-based federated learning with smpc model verification against poisoning attack for healthcare systems," IEEE Transactions on Emerging Topics in Computing, vol. 12, no. 1, 2023.
[15] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, "Federated learning with non-IID data," ArXiv:1806.00582, p. 1806, 2018.
[16] H. Tang, X. Lian, M. Yan, C. Zhang, and J. Liu, "D^2: decentralized training over decentralized data," in International Conference on Machine Learning, (Stockholm, Sweden), 2018.
[17] X. Ma, J. Zhu, Z. Lin, S. Chen, and Y. Qin, "A state-of-the-art survey on solving non-IID data in federated learning," Future Generation Computer Systems, vol. 135, pp. 244–258, 2022.
[18] K. Hsieh, A. Phanishayee, O. Mutlu, and P. Gibbons, "The non-IID data quagmire of decentralized machine learning," in International Conference on Machine Learning, (Virtual), 2020.
[19] H. Wang, Z. Kaplan, D. Niu, and B. Li, "Optimizing federated learning on non-IID data with reinforcement learning," in Conference on Computer Communications, (Virtual), 2020.
[20] M. Yurochkin, M. Agarwal, S. Ghosh, K. Greenewald, N. Hoang, and Y. Khazaeni, "Bayesian nonparametric federated learning of neural networks," in International Conference on Machine Learning, (Long Beach, CA, USA), 2019.
[21] M. Chen, R. Mathews, T. Ouyang, and F. Beaufays, "Federated learning of out-of-vocabulary words," Google Research, p. 1903, 2019.
[22] X. Zhu, J. Wang, Z. Hong, and J. Xiao, "Empirical studies of institutional federated learning for natural language processing," in Findings of the Association for Computational Linguistics: The Conference on Empirical Methods in Natural Language Processing, (Virtual), 2020.
[23] S. Caldas, S. M. K. Duddu, P. Wu, T. Li, J. Konečnỳ, H. B. McMahan, V. Smith, and A. Talwalkar, "Leaf: A benchmark for federated settings," ArXiv:1812.01097, p. 1812, 2018.
[24] M. Hamdi, S. Mestiri, and A. Arbi, "Artificial intelligence techniques for bankruptcy prediction of tunisian companies: An application of machine learning and deep learning-based models," Journal of Risk and Financial Management, vol. 17, no. 4, p. 132, 2024.
[25] D. Gholamiangonabadi, N. Kiselov, and K. Grolinger, "Deep neural networks for human activity recognition with wearable sensors: Leave-one-subject-out cross-validation for model selection," IEEE Access, vol. 8, pp. 133982–133994, 2020.
[26] F. Landi, L. Baraldi, M. Cornia, and R. Cucchiara, "Working memory connections for lstm," Neural Networks, vol. 144, pp. 334–341, 2021.
[27] Y. Yan, Y. Wang, W.-C. Gao, B.-W. Zhang, C. Yang, and X.-C. Yin, "$LSTM^2$: Multi-label ranking for document classification," Neural Processing Letters, vol. 47, pp. 117–138, 2018.
[28] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, et al., "Transformers: State-of-the-art natural language processing," in Conference on Empirical Methods in Natural Language Processing: System Demonstrations, (Virtual), 2020.
[29] G. Kaur and A. Sharma, "A deep learning-based model using hybrid feature extraction approach for consumer sentiment analysis," Journal of big data, vol. 10, no. 1, 2023.
[30] Q. Zhang, L. Shi, P. Liu, Z. Zhu, and L. Xu, "ICDN: integrating consistency and difference networks by transformer for multimodal sentiment analysis," Applied Intelligence, vol. 53, no. 12, pp. 16332–16345, 2023.
[31] A. Go, R. Bhayani, and L. Huang, "Twitter sentiment classification using distant supervision," CS224N project report, Stanford, vol. 1, no. 12, p. 2009, 2009.
[32] M. N. Fekri, K. Grolinger, and S. Mir, "Distributed load forecasting using smart meter data: Federated learning with recurrent neural networks," International Journal of Electrical Power & Energy Systems, vol. 137, p. 107669, 2022.
[33] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings, et al., "Advances and open problems in federated learning," Foundations and Trends in Machine Learning, vol. 14, no. 1–2, 2021.
[34] J. Xiao, C. Du, Z. Duan, and W. Guo, "A novel server-side aggregation strategy for federated learning in non-IID situations," in International Symposium on Parallel and Distributed Computing, (Virtual), 2021.
[35] J. Zhang, Z. Li, B. Li, J. Xu, S. Wu, S. Ding, and C. Wu, "Federated learning with label distribution skew via logits calibration," in International Conference on Machine Learning, (Baltimore, MD, USA), 2022.
[36] Z. Charles, Z. Garrett, Z. Huo, S. Shmulyian, and V. Smith, "On large-cohort training for federated learning," Advances in Neural Information Processing Systems, vol. 34, pp. 20461–20475, 2021.

DAVOUD GHOLAMIANGONABAI received the BSc degree in applied math from Ferdowsi University, Iran, and the MSc degree in industrial engineering-system and productivity management from Amirkabir University of Technology (Tehran Polytechnic), Tehran, Iran. Following the completion of his second master's degree in software engineering at Western University in London, Canada, he is currently pursuing his PhD in software engineering.

His current research interests include machine learning, deep learning, federated learning, and natural language processing.

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4

This article has been accepted for publication in IEEE Access. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2024.3453068

KATARINA GROLINGER (M'11, SM '24) is an Associate Professor of software engineering in the Department of Electrical and Computer Engineering at Western University, London, Canada. She is also Canada Research Chair (Tier 2) in Engineering Application of Machine Learning and a faculty affiliate at Vector Institute for Artificial Intelligence. Dr. Grolinger received the BSc and MSc degrees in mechanical engineering from the University of Zagreb, Croatia, and the M.Eng. and PhD degrees in software engineering from Western University, London, Canada. She has been involved in the software engineering area in academia and industry, for over 20 years. Her research interests include machine learning, federated learning, sensor data analytics, and IoT.

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License. For more information, see https://creativecommons.org/licenses/by-nc-nd/4