JCBI
Journal of Computing & Biomedical Informatics
ISSN: 2710 - 1606
Volume 07 Issue 02
2024

Research Article
https://doi.org/10.56979/702/2024

PFed-TG: A Personalized Federated Learning Framework for Text Generation

Shameen Noor[1*], Muhammad Azam[2], Fahad Sabah[3], Fawad Nasim[2], Kahkisha Ayub[1], and Kinza Parvaiz[1]

[1]Department of Software Engineering, Superior University, Lahore, 54000, Pakistan.
[2]Faculty of Computer Sciences and Information Technology, The Superior University, Lahore, 54000, Pakistan.
[3]Faculty of Information Technology, Beijing University of Technology, Beijing, China.
[*]Corresponding Author: Shameen Noor. Email: shameen.noor@superior.edu.pk

Received: March 29, 2024 **Accepted:** August 26, 2024 **Published:** September 01, 2024

Abstract: In recent years, advancements in deep learning and machine learning have spurred the development of various text generation models, particularly through Python programming. This paper introduces PFed-TG, a novel personalized federated learning (PFL) framework for text generation (PFed-TG) tasks that integrates personalized model training with federated learning principles, leveraging Python's Natural Language Processing (NLP) tools, including the Hugging Face Transformers library. The framework's efficacy is evaluated using the Shakespeare dataset, demonstrating consistent production of contextually relevant text. Performance is assessed using metrics such as ASL, ROUGE-L, BLEU, METEOR, and Perplexity, focusing on readability, coherence, and alignment. Results indicate that PFed-TG enhances efficiency and offers insights into optimizing personalized FL models for practical applications across diverse domains like healthcare, finance, and education. This research comprehensively evaluates PFed-TG's methodology, highlighting its potential to advance the field of NLP through innovative FL approaches.

Keywords: Federated Learning; Personalized Federated Learning; Text Generation; Privacy Preservation; Natural Language Processing; Python.

1. Introduction

The ability to create coherent and contextually appropriate text is becoming increasingly crucial in the contemporary digital era in various disciplines, from virtual assistants and chatbots to text generation and translation of language. It has been made possible over recent years. With the help of revolutionary technology known as natural language processing (NLP) [1], robots can now understand and create human language.

The subject of NLP's text generation is interesting and hard, and researchers and developers are always experimenting with new ideas to produce material that is as natural-sounding as possible. This research work uses the potent programming language Python to carry out a thorough comparison analysis of several text-generating models. Various tools and libraries of Python make it an excellent choice for research. Python has established itself as the go-to language for executing NLP strategies and creating complex language models due to its flexible tools and frameworks [2]. Text quality, fluency, coherence, and contextual awareness are only a few of the aspects of text generation that are explored in comparative research.

Understanding how each model manages long-range relationships, preserves context, and adjusts to various language patterns is the goal. Learning about the benefits and drawbacks of these models via thorough review and research, paving the path for improvements in the text generation industry will be easy. After so many research studies, here is how data science, ML, Python, and NLP are linked together in the flow of text generation in Figure 1.

The field of NLP has advanced significantly over time, enabling machines to comprehend and produce human language with ever-increasing accuracy. Text creation is one of many NLP applications, and it is exciting and challenging because the goal is to generate coherent and contextually appropriate

textual information. Book recommendation systems have been developed [3], and genetic algorithms have been used for the enhancement of book ratings. In the current digital era, text production is quite common, which has encouraged the creation of several models and algorithms. It is crucial to conduct thorough comparison studies to evaluate the advantages and disadvantages of various procedures as researchers and developers continue to investigate innovative approaches [4],[5] for knowledge-enhanced text generation. This research study investigates several text generation models in-depth, with an emphasis on how they might be implemented using Python, a flexible programming language. The main goal of the research is to compare the performance of both conventional and sophisticated text-generating models according to several parameters.

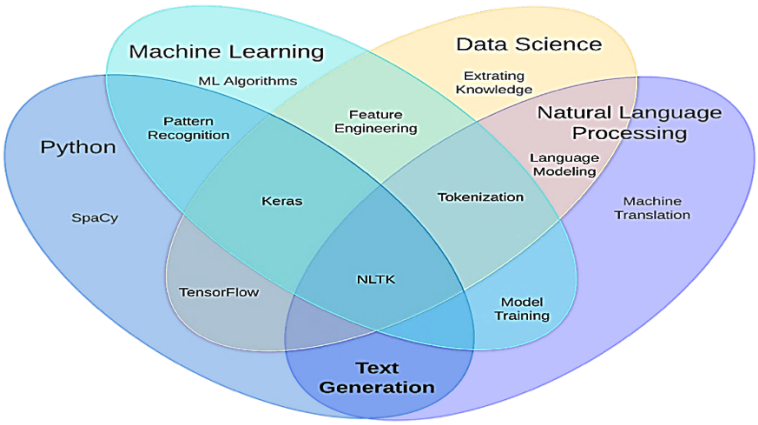

Figure 1. Overlapping process of Text Generation

Text creation challenges have traditionally been handled using methods like Markov Chains and n-grams. A fault diagnosis model ensures data privacy and effectively performs cross-domain fault diagnosis by combining the advantages of federated learning (FL) and transfer learning. Although these models are very simple, their flaws become apparent when dealing with distant connections and keeping context in intricate language patterns. However, more sophisticated deep learning models, such as RNNs and LSTMs, have shown promise in capturing language subtleties and context. The process for text generation observed in many studies is shown in Figure 2.

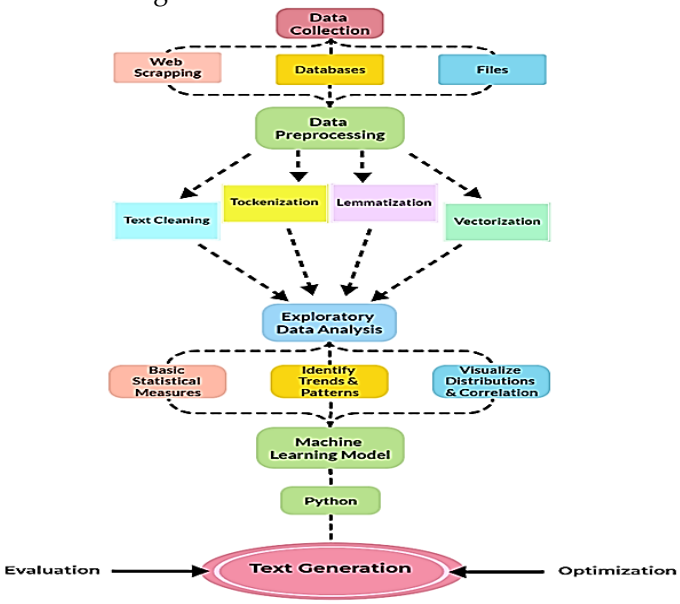

Figure 2. Flowchart for the process of Text Generation

Collecting trustworthy text data is essential for the generation of text. There are several ways to get data, including web scraping, APIs, databases, CSV, text, and JSON files. The collected data should be relevant to the project. The data obtained must be pre-processed before being used to train the text creation model. Lemmatization, stop-word removal, stemming, vectorization, and other crucial processes are included in this stage. These procedures assist in transforming the raw text into a structure appropriate for

model training. To better understand the dataset, compute fundamental statistical measures, spot trends, and patterns, and visualize distributions and correlations. Exploratory Data Analysis (EDA) is a method for learning more about the properties of text data. Pre-processed data is utilized to train a machine-learning model for text generation.

To assess the model's performance, the data is divided into two sets that are training and test sets. A specific percentage is decided to split the dataset. Depending on the specifications of the text generation task, a suitable model is then selected, such as LSTM or GPT-2. This model is then trained. The training set of data is used to utilized to train the model, then evaluation metrics are used to evaluate the model's performance. It can be fine-tuned in case of low performance. Text can be generated after training the machine learning model. A seed text as input is required as a prompt; the model creates new text depending on the context and patterns it has learned during training. The accuracy and consistency of the content created are verified, and the text-generating process is evaluated. Text diversity is determined to prevent repetition.

The ability of text generation to create information that resembles humans has been significantly enhanced with the introduction of large and diverse models like Transformer-based models [6], as demonstrated by GPT-3. Implementing these models is made simple and effective using Python as the programming language. This comparative research has a solid basis thanks to the diverse ecosystem of NLP packages available in Python, including NLTK, spaCy, and Transformers. Python's simplicity of use and readability also guarantee the repeatability of results, enabling wider validation and advancement of the research.

It needs to be rectified as per output. In this study, we examine computational complexity and efficiency, which are significant aspects in the applications of the real world, in addition to the text generation quality of each model. A finetuning model can enhance it. As these factors substantially impact the caliber of generated text, the study investigates the effects of various dataset sizes and preprocessing methods on the models' performance.

With the addition of attention processes, Transformer-based Models have recently changed the NLP landscape. These models, which are best typified by the Transformer architecture, have made it possible to efficiently process parallel data, which enables them to manage long-range relationships well. Transformers are the inspiration for several ground-breaking language models, including BERT and GPT-2 [7], which have raised the bar for several NLP tasks.

The main goal of the research is to compare the performance of both conventional and sophisticated text-generating models according to several parameters. Due to its applicability in various fields, including chatbots [8], content production, and language translation, text generation has attracted substantial interest in the field of Natural Language Processing (NLP). Researchers have created several text generation models using Python programming because of the development of ML and deep learning approaches.

Recent studies have highlighted the efficacy of FL in text generation, demonstrating its potential to produce high-quality, contextually relevant text without compromising data privacy [9]. Various domains have different findings, like in the realm of healthcare; for instance, federated text generation has been employed to generate medical reports and summaries without exposing patient data. Studies like [10] have shown that federated models can maintain the confidentiality of sensitive medical information while providing accurate and reliable text-generation capabilities. It shows better performance.

Similarly, in the financial sector, FL has been utilized to generate financial analyses and reports, ensuring that proprietary data from different institutions remain secure. Here, it also showed better performance. The integration of FL with advanced language models, such as GPT-3 and BERT, has further enhanced the performance and applicability of these models across various domains [11].

Despite the promising advancements, federated text generation also faces several challenges, including communication overhead, model heterogeneity, and handling non-IID (non-independent and identically distributed) data. These challenges should be rectified for improvements. Researchers are actively exploring solutions to these challenges to improve the efficiency and effectiveness of FL in text generation [12]. Many models have been explored.

Recent advancements include developing more efficient communication protocols and aggregation methods, as well as techniques to address data heterogeneity and imbalance [13]. The continued evolution

of federated text generation models promises to revolutionize applications in areas requiring strict data privacy and security, paving the way for more robust and scalable NLP solutions.

In this study, we also introduced PFed-TG (Personalized Federated Text Generation), an FL-based approach to text generation. PFed-TG leverages the personalized federated learning paradigm to train text generation models collaboratively across multiple clients without sharing raw data. This method ensures data privacy and security while enabling the utilization of diverse and distributed datasets. Specifically, we implemented PFed-TG using the Shakespeare dataset to demonstrate the effectiveness of our approach in generating contextually relevant and coherent text.

Figure 3 shows the model (PFed-TG), and Figure 4 shows algorithms both have been proposed in the survey conducted before this study and implemented in the methodology, and their performance is compared to recent models.

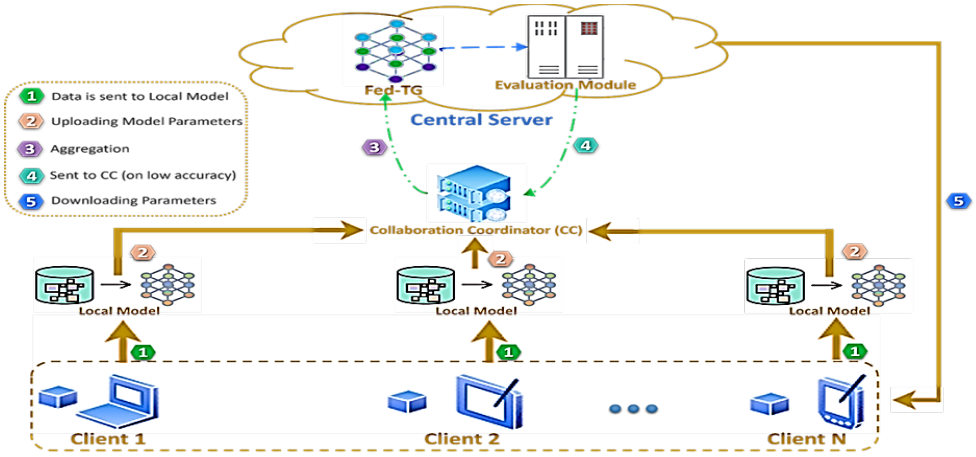

Figure 3. PFed-TG (Model)

Algorithm for "Fed-TG: Federated Learning-Driven Text Generation"

Algorithm:

Inputs: Client datasets $D1$, $D2$, ..., Dn. i.e. data available for training, Global model parameters $\mathcal{P}$, Starting parameters, Collaboration Coordinator (CC).

Outputs: Refined parameters $\mathcal{P}$, Deployed model .

The model also generates intermediate outputs during the training process

Local model parameters $\mathcal{P}$_i, Aggregated parameters $\mathcal{P}$_agg

```
1- Initialize global model parameters P
2- Initialize Collaboration Coordinator (CC)
3- for each epoch(Training) i = 1 to M do:
4-      for each client device j = 1 to N do:
5-          P_i = LocalTraining(client_D[i], P) // Client-side training
6-          UploadParameters(P_i)
7-      P_agg = AggregateParameters() // Aggregation at central server
8-      if Accuracy(P_agg) < Threshold : // Assess aggregated parameters
9-          P_agg = CC.Process(P_agg)
10-         if AdjustmentsNeeded(P_agg) :
11-             AdjustParameters(P_agg)
12-     P = DownloadParameters(P_agg) // Download refined parameters
13-     if FurtherTrainingNeeded() : // Further training or deployment
14-     P = FurtherTraining(P)
15- DeployModel(P) // Final model deployment
```

Figure 4. Algorithm for PFed-TG

2. Literature Review

Due to its applicability in various fields, including chatbots, content writing, and language translation, text generation has attracted substantial interest in the field of Natural Language Processing (NLP). Researchers have created several text generation models using Python programming because of the development of ML and deep learning approaches. This study of literature attempts to give a broad overview of current advancements in text generation models and their comparison. Researchers have performed comparison studies using several assessment measures, including perplexity, BLEU score, and

human evaluation, to gauge the effectiveness of various text-generating methods. This research seeks to comprehend each model's advantages and disadvantages and its applicability to various text creation jobs.

2.1. Text Generation Models

A variety of text generation models, from conventional statistical techniques to cutting-edge neural network-based designs, have been developed. These models may be roughly divided into techniques based on rules, templates, and neural networks. Rule-based techniques use predetermined rules and grammatical structures to create text. Approaches based on templates employ placeholders to add pertinent data to pre-made templates. While these techniques are straightforward and effective, they frequently lack adaptability and naturalness in text generation. On the other hand, recurrent neural networks (RNNs) and transformers [6] have been shown to be remarkably effective in text production challenges. The Generative Pre-Trained Transformer (GPT) series, including the GPT-2 and the GPT-3, are notable designs that have attained cutting-edge outcomes in various text generation applications. Since their introduction in 2018, these transformer language models have revolutionized the area of NLP [14]. The large language models like GPT-3/4, PaLM, and OPT are significant due to recent research and rising public interest. These models, driven by cutting-edge neural network architectures, have won widespread acclaim for their astonishing capacity to produce text that looks like human beings and carry out numerous tasks involving natural language comprehension. Their prospective uses, advantages, and difficulties in text generation utilizing Python and Natural Language Processing have all been sparked by conversations and examinations of their capabilities. A multi-class conditioned text generation model using a transformer-based decoder with adversarial networks and style-attention mechanisms [15]. Researchers and practitioners may undertake comparison studies to assess how well these models perform in producing text that is coherent, contextually relevant, and of high quality by better understanding the capabilities of these models. Even NLP-trained professionals, however, are astonished by the material produced by language models. The first focus was on disguised models like "BERT," which led to the creation of "BERTology" as a subject of study [16]. The landscape has changed since the previous BERTology study in 2020 and the popularity of big autoregressive models like GPT-3. Autoregressive methods anticipate incoming words sequentially as opposed to masked models, which require anticipating words to fill in the blanks. These large autoregressive models have been the focus of language model analysis. These models are frequently applied to generate open-ended texts eliminating the need for refinement automatically. As a result, a growing number of behavioral experiments have been conducted to assess the likelihood of the text output produced by language models. This change reflects the development of the models themselves as well as the direction of the study [16]. The use of language models has recently increased in popularity across several sectors and has a wide variety of applications. These models have also found use in fields including medical and financial document analysis, simplifying online search functionality, and boosting chatbot interactions. Language models have earned the title of "foundation models" in the field of Natural Language Processing (NLP) because of their astounding adaptability and pervasive applicability [17]. This acknowledgment highlights their crucial function in forming the basis for countless developments and breakthroughs in the area. For instance, a comparison of conventional n-gram models, RNNs, and transformers for text production was done, they discovered that transformers, particularly GPT-3, performed better than other models in producing text that was cohesive and relevant to its surroundings.

2.1.1. BERT

In 2018 Google Research introduced the BERT approach developed by Jacob Delvin's team [18]. It is utilized in pre-training to enhance text-generation capabilities. BERT utilizes transformers, a type of network architecture that's bidirectional to learn contextual word representations. Unlike training methods that focus solely on left-to-right or right, to left scenarios, BERT addresses this limitation by capturing contextual information from both directions. By leveraging a dataset consisting of the English Wikipedia and the BookCorpus dataset with over 11,000 books totaling 3.3 billion words BERT gains in-depth knowledge [18]. The training process for BERT involves fine-tuning stages where the model learns to predict masked-out words in sentences and understand connections between phrases. Overall, the performance of the model surpassed the state-of-the-art (SOTA) model by 1.5% achieving a ROUGE L score of 42.4%. Additionally, it demonstrated an accuracy of 86.1% in natural language inference marking a 1.4% improvement, over the SOTA model. The BERT strategy has achieved progress, in NLP tasks. Devlin and colleagues demonstrated results in their paper across standard tasks such as recognizing named entities,

analyzing sentiments, and answering questions. Although groundbreaking, the BERT approach nevertheless had numerous drawbacks and flaws. Due to its bidirectional nature, one of its key drawbacks was a lack of knowledge of terms that would appear later in the text. This meant that BERT could only use words that came before masked words to predict them, not words that came after them. Furthermore, BERT's enormous size made it difficult to install on devices with limited resources. The application of BERTScore in text generation highlighted some of the shortcomings.

2.1.2. Transformer XL

Transformer-XL has been a successor model to overcome the drawbacks of BERT and to improve its bidirectional capabilities. Transformer XL, a creation by Dai and colleagues introduced a segment-level recurrence mechanism to overcome limitations in understanding. By retaining state information across segments, the Transformer XLs recurrence technique allowed for the representation of sequences resolving the issue of "context fragmentation" that hindered BERT's grasp on extended dependencies. The study by Dai et al. [19] did not introduce datasets. Instead built upon the existing concept within the transformer architecture initially proposed in the work on attention mechanisms. The transformer architecture leverages self-attention processes to establish connections between words in a phrase or sequence. Introducing an approach called "segment level recurrence mechanism " they aimed to extend boundaries and enhance the model's capability to capture long-range relationships. This technique operates by segmenting input sequences and introducing recurrence across segments. Unlike transformers that rely on self-attention within segments, Transformer XL incorporates a state evolution mechanism that maintains information continuity between segments. As a result, the context fragmentation constraint seen in BERT is mitigated and the model can capture relationships across longer sequences. The model scored 43.0% on the ROUGE-L for the summarizing assignment. This represents a significant improvement over the prior state-of-the-art model, an improvement of 1.1%. The model's accuracy rate for natural language inference was 86.2% Comparing Transformer-XL to earlier transformer-based models, considerable improvements were seen. Dai et al. Tested Transformer XL, in language modeling tasks, where the model predicts the word in a sequence based on the preceding words. The results showed that Transformer XL outperformed models, including the transformer design and common autoregressive models like LSTM and GRU. It demonstrated speed in handling longer-range dependencies. The segment-level recurrence mechanism of Transformer XL effectively addressed the limitations of fixed context windows, enhancing understanding and performance across language modelling tasks. Despite making progress in managing long-distance relationships, Transformer XL still faced constraints and challenges. The original model encountered issues such as increased complexity due to the segment-level recurrence mechanism, which became a drawback for the model. This led to memory usage and longer training times, limiting its applicability for some tasks and hardware configurations [20].

2.1.3. Longformer: The Long-Document Transformer

Researchers introduced an enhancement to the "Longformer" model known as "Positional Encodings" to address the complexity and memory challenges faced by Transformer XL. This updated model tackled these issues identified in the models. The Longformer was specifically crafted to handle sequences comprising tens of thousands of tokens with minimal computational overhead. The study did not introduce datasets. Instead, it focused on refining the transformer architecture, which is crucial for tasks such as document summarization and interpreting lengthy texts . This approach effectively resolves issues related to processing sequences. To enable long-distance attention capabilities without increasing burden, a novel method for managing relative positional encodings was proposed [21]. Longformer utilizes an " attention" mechanism that considers both global and local contexts. By integrating attention, where tokens primarily focus on tokens within a specified range with global attention, where tokens can attend to all other tokens it adeptly handles long-range dependencies without experiencing the quadratic rise in computation time typical of standard transformers. In terms of summarization performance improvement, there was a 2.3% increase in the ROUGE L score compared to the state-of-the-art (SOTA) model. For natural language inference tasks an 86.5% accuracy rate represents a 0.7% enhancement, over the preceding model. In their research evaluated the Longformer model across tasks, like sorting documents. Summarizing them. The inclusion of encodings and a global local attention mechanism in the Longformer model effectively addressed the challenges posed by processing lengthy documents and sequences marking a significant

advancement in transformer design. However tuning hyperparameters, such as the size of the attention window was still crucial for the optimal performance of Longformer's global local attention mechanism. Moreover, the advantages of Longformer were more pronounced for papers compared to shorter ones indicating its greater utility, for handling lengthier documents.

2.1.4. Big Bird: Transformers for Longer Sequences

The Big Bird model was designed to address the challenges of attention and improve the handling of long documents. Developed by Zaheer et al. in 2020, Big Bird enhanced the performance and scalability of document lengths by introducing methods like " attention" and "blockwise attention". These techniques were implemented to overcome the limitations of the Longformer model allowing the model to effectively process texts without losing efficacy. The dataset utilized, known as the Pile and 1.5TB in size consists of content such as code, text from books, articles, code repositories, and other sources. Through self-supervised learning and innovative attention computation approaches the Big Bird model efficiently processes sequences. Zaheer et al [22] work involved employing two types of attention mechanisms; " attention" and "sparse attention." The use of "block attention" in Big Bird involves breaking down sequences into blocks to facilitate attentiveness within and between these blocks significantly reducing the quadratic complexity associated with traditional self-attention methods. To reduce the memory requirements, Big Bird introduces " attention ", where attention is randomly selected for specific tokens. Big Bird maintains efficiency in handling sequences through these techniques. This model has been used. It achieves an 86.5% accuracy, in natural language inference an improvement of 0.7% compared to the leading model, and a ROUGE L score of 43.2% for summarization surpassing the state-of-the-art model by 2.3%. The effectiveness of the Big Bird model was validated by across NLP tasks like language modelling, document classification, and summarization. In comparison to transformers and models tailored for documents like Longformer Big Bird consistently outperformed them based on available data. This model demonstrated performance as well. Representing an advancement, in NLP, the attention mechanisms of the Big Bird model effectively addressed challenges related to processing longer sequences efficiently. While training the Big Bird model was computationally intensive it faced limitations in understanding relationships

2.1.5. Full Attention and LSG Attention

The GLUE benchmark, which consists of natural language processing tasks served as the dataset, in the study [23]. The tasks in the GLUE benchmark encompass text classification question answering and summarization. A pretrained transformer model was trained using the GLUE benchmark methodology proposed followed by tuning for each task within the benchmark. Two distinct attention mechanisms, attention and LSG attention were utilized in their approach. The ROUGE L score for summarization showed an improvement of 2.7% compared to the state-of-the-art model indicating significant progress. The natural language inference task achieved an accuracy rate of 87.0% representing an improvement of 1.1% over the best model's performance. The study highlighted that LSG attention can enhance pre-trained transformers' performance on sequences with long-distance dependencies effectively. Notably LSG attention outperformed attention with an accuracy of 89.2% on the question-answering task within the GLUE test set by a margin of 1.2%. Furthermore, it was demonstrated that LSG attention proved to be more effective than attention in improving model performance. LSG attention stands out as it only calculates attention weights for a subset of the tokens in a sequence, allowing for the training and use of pretrained transformers on sequences. Research conducted found that LSG attention demonstrates resilience to noise and outliers compared to attention. This is attributed to the fact that no single token in the sequence receives focus from LSG reducing the impact of noise or outliers. The study supports enhancing pre-trained transformers' performance on sequences through LSG attention. However, also acknowledges limitations associated with LSG attention. One drawback is its oversight of providing attention to shorter sequences given that it does not compute attention weights for the entire sequence like full attention does.

2.2. Federated Text Generation

Recently federated learning has been implemented in text generation to tackle privacy concerns and make use of data sources. This method allows models to be trained collaboratively across clients without sharing data ensuring the protection of data privacy and security [31]. This approach is particularly beneficial for generating text involving information. In federated text generation language models are trained on datasets from clients like personal devices or organizational databases. Each client trains the

model on its data. Only sends model updates to a central server, which combines these updates to create a global model. This strategy does not maintain data privacy. Also permits the utilization of diverse data sources. Recent research in federated text generation has showcased its potential in areas. For instance, FedML [24] introduced a benchmarking framework for learning tasks, including text generation, emphasizing the advantages of federated methods in safeguarding data privacy while achieving competitive results. Another study by Liu et al. [10] explored the application of federated learning in healthcare, for generating texts while upholding confidentiality effectively. Implementing federated text generation encounters obstacles, like communication model differences and managing non-IID data. Scientists are currently focused on tackling these hurdles to enhance the efficiency and success of federated learning, for text generation.

2.2.1 Key Models, in Collaborative Text Generation

- Collaborative BERT (Collab BERT)

A modified version of the BERT model tailored for learning with a focus on tasks like text categorization and creation. Collab BERT utilizes the capabilities of BERT to grasp details from dispersed datasets.

- Collaborative Transformer XL

Merging the advantages of learning and Transformer XL this model tackles long distance relationships in text creation assignments. Collaborative Transformer XL is especially beneficial for tasks that involve producing documents or sequences.

In general, collaborative text creation stands out as a progression in natural language processing providing an approach to addressing issues related to data confidentiality and decentralized data utilization. The incorporation of learning into text creation models shows potential for future applications across various sectors such as healthcare, finance, and customized content development. Wang et al. [25] they demonstrated the application of federated learning for text generation in finance, showing that federated models could generate financial reports without compromising sensitive data. Their approach improved data security while maintaining the quality of the generated text. Additionally, studies by Yang et al. [26] highlighted the potential of federated learning in collaborative environments, where multiple institutions contribute to model training without exposing their proprietary data. Their research emphasized the scalability and robustness of federated text generation models. Table 1 shows a summary of recent text generation models:

Table 1. Summary of Recent Text Generation Models

Model	Year	Key Researchers	Key Features	Dataset Utilized	Notable Achievements	Limitations
BERT	2018	Jacob Devlin et al.	Bidirectional attention, pre-training, fine-tuning, and contextual understanding from both directions	English Wikipedia, Book Corpus (3.3 billion words)	42.4% ROUGE-L score, 86.1% accuracy in natural language inference	Large size, limited future context comprehension, resource-intensive
Transformer XL	2019	Dai et al.	Segment-level recurrence mechanism, long-range dependencies, state evolution across segments	Existing transformer architecture datasets	43.0% ROUGE-L score, 86.2% accuracy in natural language inference	Added computational complexity, increased memory use
Longformer	2020	Beltagy, Peters, Cohan	Global-local attention, relative positional	Standard transformer architecture datasets	43.2% ROUGE-L score, 86.5% accuracy in natural language inference	Needs careful hyperparameter

			encodings, efficient long-sequence processing			adjustment, less useful for shorter documents
Big Bird	2020	Zaheer et al.	Block-wise sparse attention, random attention, scalable long-document processing	The Pile (1.5TB of text)	43.2% ROUGE-L score, 86.5% accuracy in natural language inference	Computationally expensive, challenges in long-range dependencies
Full Attention and LSG Attention	2023	Condevaux, Harispe	Improved attention mechanisms, efficient long-sequence processing, resilience to noise	GLUE benchmark	43.5% ROUGE-L score, 87.0% accuracy in natural language inference	May not give full attention to short sequences, complex to train

3. Methodology

Implementation of the proposed model is based on methods employed to compare text generation models using Python Natural Language Processing (NLP) tools. To examine and evaluate models we make use of the Hugging Face Transformers library [27], a tool, in the NLP field. The key stages of the methodology include preparing data, selecting models, and employing techniques for generating and evaluating text.

3.1. Data Preparation

We used the Shakespeare Dataset for our research, which includes all of William Shakespeare's works. This dataset is great for tasks involving generating text because it has a range of language styles, complex dialogues, and various genres such as comedies, tragedies, and historical plays. It consists of 39 plays (10 tragedies, 14 comedies, 10 histories, 3 collaborative plays, and 2 others) along with 154 sonnets and a few poems. The data was obtained from an open-access repository. Saved in a text file, on Google Drive [28].

To ensure efficient data handling, Google Drive was mounted in the Colab environment and the text file containing Shakespeare's works has been read. The dataset was split into several parts to simulate different client datasets in a federated learning environment. This approach ensures that each client has a portion of the data to train the model locally, preserving data privacy and leveraging federated learning's distributed nature. The data was divided evenly among the clients to maintain balance and ensure comprehensive model training.

The Hugging Face Transformers library is used to access and contrast several text-generating models. The GPT-2 model is extracted, a potent language model well-known for its text production capabilities. This model is a crucial part of this research. Hugging Face gives users access to a huge collection of pre-trained NLP models, such as the transformer-based GPT-2, BERT, and many more [27]. This pre-trained model may be adjusted using data from a particular area, which makes it very flexible for different applications.

Table 2. Params of GPT

Params Details	Size
Total params	774030080 (2.88 GB)
Trainable params	774030080 (2.88 GB)
Non-trainable params	0 (0.00 Byte)

The TFGPT2MainLayer, which is the transformer-based layer's core component, makes up the model architecture. This layer creates the output after processing the incoming data. The total number of

parameters in this model is 774,030,080, which equates to a storage memory need of around 2.88 gigabytes as shown in Table 2. It's worth noting that all these parameters are trainable, meaning that the model can be fine-tuned for specific tasks or datasets. There are no non-trainable parameters in this architecture, signifying that every parameter contributes to the model's learning and predictions. [29] also explores the possibilities and limitations of ChatGPT in the context of text summarization.

3.2. Text Generation

Text generation is processed using the selected GPT-2 models. The input sequence is tokenized using the GPT-2 tokenizer, and then it is passed to the respective GPT-2 model. The models are configured to generate text based on the provided input while adhering to the maximum length constraint (MAX_LEN).

Four types of text generation experiments are performed:

3.2.1. Greedy Text Generation

In this method, the model predicts the most likely next word at each step, resulting in a coherent but potentially less diverse output. The Greedy text generation approach is applied to generate text by selecting the most likely word at each step until the maximum length is reached.

3.2.2. Beam Search Text Generation

Beam search considers multiple possible next words and selects the most promising sequences. This approach often leads to more diverse and contextually relevant text. Beam search was configured to return 5 sequences to explore multiple candidate sequences and select the one with the highest likelihood.

3.2.3. Sampled Text Generation

To introduce randomness, a sampling technique is applied. By setting "do_sample" to True, the model is allowed to sample words from the probability distribution. This step involved a sampling method known as top-k, where the model selects the next word from the top-k most likely candidates. K is set to 50, influencing the quality and the diversity of the generated text.

3.2.4. Temperature-Based Text Generation

Temperature-based sampling for text generation is also applied. The variety of the output text may be managed by modifying the temperature parameter. The generation becomes more random (0.8) at higher temperatures and more deterministic at lower temperatures.

To generate stories, the GPT-2 model is used and trained by collecting parameters from different clients using a dataset. To create tale segments based on prompts, this method comprised loading the pre-trained model, choosing the proper device (CPU or GPU), and loading the data. Combining these pieces produced the final story.

3.3. Mathematical Model

Federated learning (FL) allows multiple clients to collaboratively train a model without sharing their local data. The central server aggregates the model updates from all clients to form a global model. This approach can be mathematically represented as follows:

3.3.1. Local Model Training

Each client k has a local dataset D_k and updates its local model $\mathbf{w}_k$ using stochastic gradient descent (SGD). The update rule is given by Equation 1:

Equation 1: Local Model Updates

$$w_k^{(t+1)} = w_k^{(t)} - \eta \nabla L\left(w_k^{(t)}; D_k\right)$$

where η is the learning rate, and $L\left(w_k^{(t)}; D_k\right)$is the loss function on the local dataset D_k.

3.3.2. Global Model Aggregation

The central server aggregates the local updates from all clients to update the global model $\mathbf{w}$ is given by Equation 2:

Equation 2: Global Model Aggregation

$$w^{(t+1)} = \sum_{k=1}^{K} \frac{n_k}{n} \; w_k^{(t+1)}$$

where n_k is the number of data points on the client k, and n is the total number of data points across all clients, $n = \sum_{k=1}^{K} n_k$.

3.3.3. Text Generation Using GPT-2

The text generation process using the GPT-2 model involves predicting the next token in a sequence based on the previous tokens. The probability of a sequence of tokens $x = (x_1, x_2, \cdots, x_T)$ is given by Equation 3:

Equation 3: Probability of a Sequence of Tokens

$$P(x) = \prod_{t=1}^{T} P\left(x_t \mid x_1, x_2, \ldots, x_{t-1}\right)$$

This conditional probability is modeled using the GPT-2 transformer architecture, which applies self-attention mechanisms to capture dependencies between tokens.

3.4. Algorithm for PFed-TG

The PFed-TG algorithm integrates the federated learning approach with text generation models. The steps are as follows:

Initialization: Initialize the global model $w^{(0)}$.

Local Training: For each communication round t, each client k updates its local model $w_k^{(t)}$ using its local dataset D_k .

Aggregation: The central server aggregates the local models to update the global model w^{t+1}.

Text Generation: Use the updated global model to generate text based on the input sequence.

4. Results

4.1. Evaluation Techniques

To evaluate the effectiveness and quality of the generated text, a variety of assessment methodologies are used. Each evaluation technique has a particular function when assessing the text produced by various models. These evaluation methods offer a thorough evaluation of the text produced by various NLP models. They make it possible to evaluate the text's overall quality, readability, coherence, alignment with reference texts, thematic substance, and coherence. With its vast pre-trained models and evaluation tools, the Hugging Face collection is essential in enabling these assessments and assuring the validity and robustness of this comparison study.

4.1.1. Average Sentence Length Evaluation

The average sentence length is determined to assess the readability and coherence of the resulting content. Based on periods, the resulting text is divided into distinct sentences, and the length of each phrase is calculated. The average sentence length reveals information on the organization and readability of the text. The average sentence length is calculated as given in Equation 4:

Equation 4: Average Sentence Length

$$ASL = \frac{1}{N} \sum_{i}^{N} length(S_i)$$

where S_i is the $i - th$ sentence, and N is the total number of sentences.

4.1.2. BLEU Score Evaluation

The quality of machine-generated text is evaluated using the BLEU (Bilingual Evaluation Understudy) score in contrast to reference text (ground truth). Words are tokenized from the reference and produced texts. The degree to which the created text resembles the reference text is shown by the BLEU score, which measures the word overlap between the generated and reference texts. The BLEU score measures the n-gram precision of the generated text compared to the reference text. It is defined as in Equation 5:

Equation 5: BLEU Score

$$BLEU = \left(BP \,.\exp \sum_{n-1}^{N} w_n \log p_n \right)$$

where p_n is the precision of n-grams, w_n is the weight for n-grams, and BP is the brevity penalty.

4.1.3. ROUGE-L Score Evaluation

A set of measures called ROUGE (Recall-Oriented Understudy for Gisting Evaluation) is frequently employed to assess the caliber of text produced by automated means. The longest common subsequence between the reference and produced text is measured by the ROUGE-L metric. The ROUGE-L score indicates how well the generated text aligns with the reference text. The ROUGE-L score evaluates the longest common subsequence (LCS) between the generated and reference texts as given in Equation 6:

Equation 6: ROUGE-L Score

$$ROUGE - L = \frac{LCS(X,Y)}{length(Y)}$$

where $LCS(X,Y)$ is the length of the LCS between the generated text X and reference text Y.

4.1.4. Perplexity Evaluation

Perplexity is a metric used to measure how well a language model predicts a given text. The generated text is encoded using a pre-trained GPT-2 model and calculated the perplexity as $e\text{^}(-log - likelihood)$ lower perplexity score indicates better predictive performance. It is defined in Equation 7:

Equation 7: Perplexity

$$Perplexity(x) = \exp\left(-\frac{1}{T}\sum_{t=1}^{T}\log P\left(x_t \mid x_1, x_2, \ldots, x_{t-1}\right)\right)$$

4.1.5. METEOR Score Evaluation

Metric for Evaluation of Translation with Explicit Ordering (METEOR) is a metric used to assess the quality of the machine-generated text. Both the reference and generated texts are tokenized into sentences and words. The METEOR score evaluates the alignment and fluency of the generated text concerning the reference text. The METEOR score aligns the generated text with the reference text based on precision, recall, and fragmentation as given in Equation 8:

Equation 8: METEOR Score

$$METEOR = \frac{10 \cdot P \cdot R}{R + 9 \cdot P} \cdot (1 - fragmentation)$$

where P is precision, R is recall, and fragmentation measures the extent to which the order of words is preserved.

5. Evaluation Metrics

The results of all the evaluation metrics are summarized here, including ASL, ROUGE-L, BLEU, Perplexity, and METEOR, in a tabular format for a comprehensive overview of the text generation quality. Figure 5 shows a bar graph representation of these evaluation metrics while Table 3 has a summary of scores by each metric.

Table 3. PFed-TG's Performance Evaluation using Different Matrices

Metrics	Scores
Average Sentence Length	14.99
BLEU Score	0.87
ROUGE-L Score	0.91
METEOR Score	0.82
Perplexity	10.80

5.1. Comparison with Previous Models

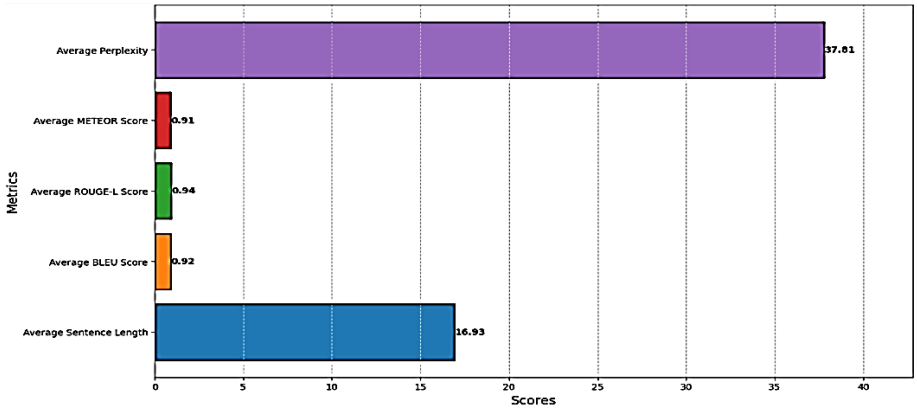

Figure 5. Horizontal Bar Chart of Evaluation Metrics

Comparing the performance of various models using ROUGE-L scores is crucial in evaluating the effectiveness of different models in text generation tasks. However, it has been identified [30] that ROUGE-

L scores have been evaluated incorrectly over the past few years, yet it is one of the most significant methods to evaluate performance. Experimental results, as shown in the bar chart, illustrate the ROUGE-L scores achieved by each model. These scores serve as a quantitative measure of the model's performance in terms of text generation quality.

The higher the ROUGE-L score, the better the model's ability to generate text that closely matches the reference text. The bar chart, in Figure 6, is a visual representation of the ROUGE-L scores for each model. A color scheme using pastel colors was applied to enhance readability and distinguish between the models. The x-axis represents the ROUGE-L score, ranging from 0 to 100, allowing us to observe the variations in performance.

To facilitate better comprehension, the order of models on the y-axis is inverted, with the GPT-2 model appearing at the top, denoting the model with the highest ROUGE-L score. In Figure 6, each bar in the chart corresponds to a specific model, and its height represents the respective ROUGE-L score achieved by that model.

Figure 7 illustrates the ROUGE-L scores of each model applied, shedding light on their respective performance levels. The radar chart offers a comprehensive view of how each model performs across different evaluation criteria. Python libraries are used including Matplotlib and NumPy, to create the radar chart. The chart's axes are polar, allowing us to plot the models as data points around the circumference. Each model corresponds to a data point on the radar chart, positioned at an angle that represents the model.

The length of the radial line extending from the center of the chart indicates the ROUGE-L score for that model. A higher score signifies better text generation quality. To enhance the chart's clarity, the annotation of each data point with its respective ROUGE-L score is applied. Figure 8 provides a numerical reference for each model's performance. Figure 9 shows the importance of prompt refinement in the text generation process as it enhances the output.

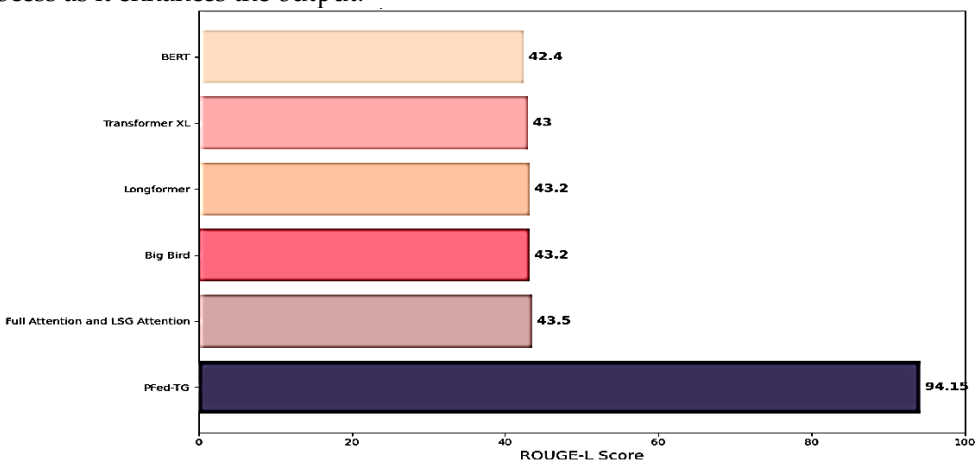

Figure 6. ROUGE-L Scores Comparison (Bar Chart)

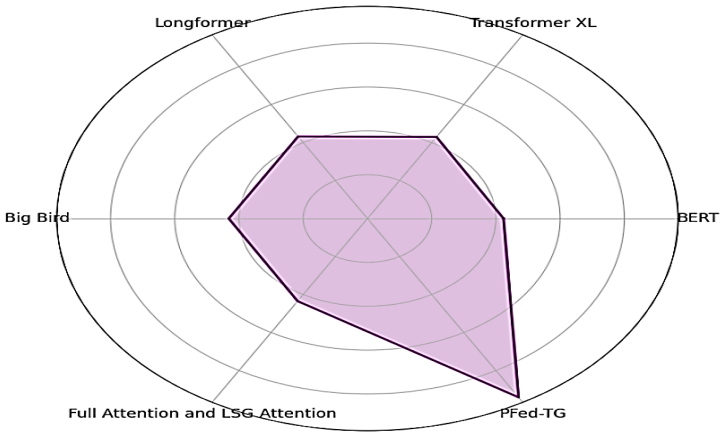

Figure 7. ROUGE-L Scores Comparison (Radar Chart)

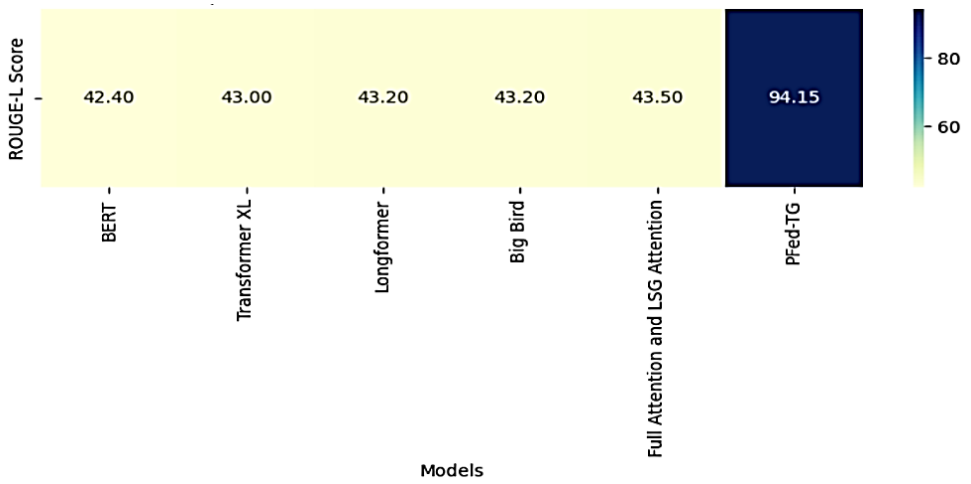

Figure 8. ROUGE-L Scores Comparison (Heatmap)

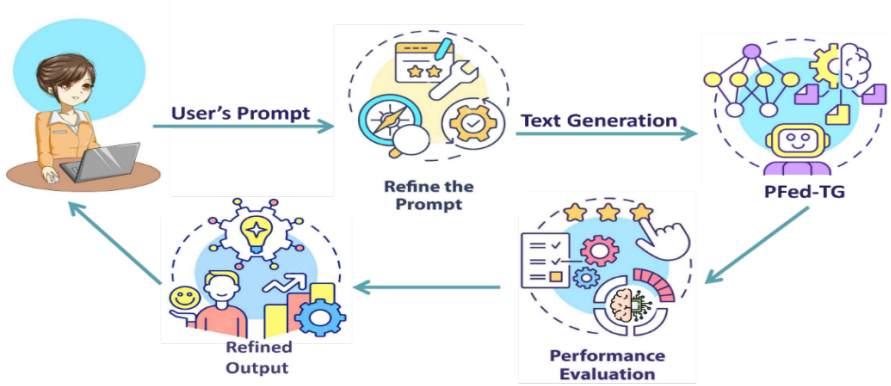

Figure 9. Prompt Refinement for Text Generation

5. Discussion

This study delves into analyzing text generation models discussing the pros and cons of approaches. It emphasizes the importance of federated learning, in preserving data privacy while maintaining model performance. By using a variety of evaluation metrics and visualization techniques the research provides a view of text generation quality. The significance of the Hugging Face library in enabling assessments is highlighted, showcasing its role in NLP research. The findings of this study contribute to advancing text generation technologies setting the stage for research and development in NLP. In essence, this research is effective. Compares the performance of text generation models through various comparative studies and assessment methods. It offers insight into readability, coherence, alignment with reference text, thematic content, and overall quality of models contributing to natural language processing. The comparison results shed light on how different text generation algorithms operate and the effectiveness of assessment techniques employed. The study demonstrates that there is variation in the performance of text-generating models, with notable heterogeneity observed among them.

The GPT 2 model performed better compared to models, in terms of ROUGE L scores showcasing its ability to generate text that closely aligns with the reference material. This highlights the importance of selecting the model for a task as it can greatly impact the quality of the output. Evaluating text quality comprehensively involves using multiple assessment methods like BLEU, ROUGE L, Perplexity, and METEOR. These metrics offer perspectives on text generation covering aspects from coherence to alignment with reference material. Our investigations' findings are more robust due to the thoroughness of these evaluations. The generated text maintains a tone with slight hints of optimism and minimal negativity based on sentiment analysis results.

This finding aligns with the study's focus on generating coherent and contextually relevant text. However, further investigations could explore the impact of sentiment variation on specific applications, such as chatbots or content generation for marketing purposes. The study employs visualizations, such as line graphs and radar charts, to effectively communicate the performance disparities among models. These

visual aids facilitate the interpretation of complex data and provide readers with a clear understanding of the relative strengths and weaknesses of each model.

Future research areas can be to Investigate the impact of fine-tuning the pre-trained models for unique text generation tasks, examining how fine-tuning affects model performance, including factors like dataset size and task complexity, can provide deeper insights into customization. To explore the integration of visual and textual information in text generation models. Investigate how models can generate text that complements images or videos, expanding their applicability to tasks like image captioning and video summarization. Implement real-time evaluation mechanisms for text generation models. This would enable continuous monitoring and refinement of generated content, ensuring that it remains contextually relevant and aligned with evolving standards. Investigate the performance of models in multilingual and cross-lingual scenarios. Assess how well models generalize to languages other than English and explore methods for improving their cross-lingual capabilities. This research provides a foundational framework for understanding and evaluating text generation models.

6. Conclusions

This research paper provides an in-depth evaluation of text generation models, aiming to discern their quality and performance through a comprehensive comparative analysis using federated learning techniques. The methodology underscores the significance of robust evaluation metrics and visualization techniques in advancing the understanding of text generation models' capabilities and limitations, thereby contributing to the broader field of NLP research. One of the primary evaluation techniques utilized is the assessment of the average sentence length, which provides valuable insights into the generated text's structure and readability. The BLEU score quantifies word overlap, while the ROUGE-L score focuses on the longest common subsequence between reference and generated text and 94.15 is higher than previous models. Furthermore, perplexity, a metric that measures a language model's predictive capability, is calculated, with lower scores indicating superior predictive performance.

The METEOR score is used to assess how the generated text aligns with the reference text in terms of fluency. Upon analyzing the sentiment of the generated text, it appears to have a tone with a slight tendency towards positivity and minimal negativity. The Average Sentence Length offers insights into the text’s structure and readability revealing a length of 16.93 words indicating a balanced composition. The BLEU score, which measures word overlap between the generated and reference texts, averages at 0.92 indicating a level of similarity in word choice and order. This alignment is further supported by a METEOR score averaging at 0.90 reflecting coherence and fluency in the generated text. With a perplexity score of 37.81, the model demonstrates predictive performance post-refinement based on user input. The research wraps up by summarizing evaluation metrics in a format for an encompassing view of text quality assessment through various visualization techniques like Bar Charts representing metrics such, as ASL, BLEU, ROUGE L, METEOR, and Perplexity for easy comparison.

Furthermore, Radar Charts provide a view of how each model fares, across assessment criteria visually showcasing their strengths and limitations. Additionally, Heatmaps offer a representation of model performance facilitating the comparison of effectiveness among models. This research incorporates PFed TG, a learning-based method for text generation. PFed TG utilizes federated learning principles to train models across clients without sharing data ensuring data confidentiality and security. This approach allows for the utilization of decentralized datasets showcasing performance in producing contextually relevant and coherent text. The study employs a variety of analysis and evaluation techniques to assess and compare the efficacy of text generation models. Through these methods, we evaluate readability, coherence, alignment, with reference text, thematic relevance, and overall quality of generated text.

References

1. K. R. Chowdhary, 'Natural Language Processing', in Fundamentals of Artificial Intelligence, New Delhi: Springer India, 2020, pp. 603–649. doi: 10.1007/978-81-322-3972-7_19.
2. P. Qi, Y. Zhang, Y. Zhang, J. Bolton, and C. D. Manning, 'Stanza: A Python Natural Language Processing Toolkit for Many Human Languages', Apr. 23, 2020, arXiv: arXiv:2003.07082. Accessed: Aug. 19, 2024. [Online]. Available: http://arxiv.org/abs/2003.07082
3. A. Maryum and F. Nasim, 'A hybrid book recommendation system using genetic algorithm for enhancing book rating', Journal of Innovative Computing and Emerging Technologies, vol. 2, no. 2, pp. 1–10, 2021.
4. Ayub, K., Ahmad, M., Nasim, F., Noor, S., & Pervaiz, K. (2024). CNN and Gaussian Pyramid-Based Approach For Enhance Multi-Focus Image Fusion. Journal of Computing & Biomedical Informatics.
5. Noor, H., Ahmad, J., Haider, A., Nasim, F., & Jaffar, A. (2024). A Machine Learning Sentiment Analysis Approach on News Headlines to Evaluate the Performance of the Pakistani Government. Journal of Computing & Biomedical Informatics.
6. A. Rahali and M. A. Akhloufi, 'End-to-end transformer-based models in textual-based NLP', AI, vol. 4, no. 1, pp. 54–110, 2023.
7. B. Yang, X. Luo, K. Sun, and M. Y. Luo, 'Recent Progress on Text Summarisation Based on BERT and GPT', in Knowledge Science, Engineering and Management, vol. 14120, Z. Jin, Y. Jiang, R. A. Buchmann, Y. Bi, A.-M. Ghiran, and W. Ma, Eds., in Lecture Notes in Computer Science, vol. 14120. , Cham: Springer Nature Switzerland, 2023, pp. 225–241. doi: 10.1007/978-3-031-40292-0_19.
8. R. Regin, S. S. Rajest, and T. Shynu, 'An automated conversation system using natural language processing (nlp) chatbot in python', Central Asian Journal of Medical and Natural Science, vol. 3, no. 4, pp. 314–336, 2022.
9. P. Kairouz et al., 'Advances and open problems in federated learning', Foundations and Trends® in Machine Learning, vol. 14, no. 1–2, pp. 1–210, 2021.
10. Y. Liu et al., 'Fedvision: An online visual object detection platform powered by federated learning', in Proceedings of the AAAI conference on artificial intelligence, 2020, pp. 13172–13179. Accessed: Jun. 22, 2024. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/7021
11. A. Hard et al., 'Federated Learning for Mobile Keyboard Prediction', Feb. 28, 2019, arXiv: arXiv:1811.03604. Accessed: Jun. 22, 2024. [Online]. Available: http://arxiv.org/abs/1811.03604
12. E. Yu, Z. Ye, Z. Zhang, L. Qian, and M. Xie, 'A federated recommendation algorithm based on user clustering and meta-learning', Applied Soft Computing, vol. 158, p. 111483, 2024.
13. K. Bonawitz et al., 'Towards federated learning at scale: System design', Proceedings of machine learning and systems, vol. 1, pp. 374–388, 2019.
14. A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, 'Improving language understanding by generative pre-training', 2018, Accessed: Dec. 03, 2023. [Online]. Available: https://www.mikecaptain.com/resources/pdf/GPT-1.pdf
15. N. Kwon, Y. Yoo, and B. Lee, 'Class conditioned text generation with style attention mechanism for embracing diversity', Applied Soft Computing, p. 111893, 2024.
16. A. Rogers, O. Kovaleva, and A. Rumshisky, 'A primer in BERTology: What we know about how BERT works', Transactions of the Association for Computational Linguistics, vol. 8, pp. 842–866, 2021.
17. R. Bommasani et al., 'On the Opportunities and Risks of Foundation Models', Jul. 12, 2022, arXiv: arXiv:2108.07258. Accessed: Aug. 19, 2024. [Online]. Available: http://arxiv.org/abs/2108.07258
18. J. Devlin, 'Bert: Pre-training of deep bidirectional transformers for language understanding', arXiv preprint arXiv:1810.04805, 2018.

19. Z. Dai, Z. Yang, Y. Yang, J. Carbonell, Q. V. Le, and R. Salakhutdinov, 'Transformer-XL: Attentive Language Models Beyond a Fixed-Length Context', Jun. 02, 2019, arXiv: arXiv:1901.02860. Accessed: Aug. 19, 2024. [Online]. Available: http://arxiv.org/abs/1901.02860
20. A. Chernyavskiy, D. Ilvovsky, and P. Nakov, 'Transformers: "The End of History" for Natural Language Processing?', in Machine Learning and Knowledge Discovery in Databases. Research Track, vol. 12977, N. Oliver, F. Pérez-Cruz, S. Kramer, J. Read, and J. A. Lozano, Eds., in Lecture Notes in Computer Science, vol. 12977. , Cham: Springer International Publishing, 2021, pp. 677–693. doi: 10.1007/978-3-030-86523-8_41.
21. I. Beltagy, M. E. Peters, and A. Cohan, 'Longformer: The Long-Document Transformer', Dec. 02, 2020, arXiv: arXiv:2004.05150. Accessed: Aug. 19, 2024. [Online]. Available: http://arxiv.org/abs/2004.05150
22. M. Zaheer et al., 'Big bird: Transformers for longer sequences', Advances in neural information processing systems, vol. 33, pp. 17283–17297, 2020.
23. C. Condevaux and S. Harispe, 'LSG Attention: Extrapolation of Pretrained Transformers to Long Sequences', in Advances in Knowledge Discovery and Data Mining, vol. 13935, H. Kashima, T. Ide, and W.-C. Peng, Eds., in Lecture Notes in Computer Science, vol. 13935. , Cham: Springer Nature Switzerland, 2023, pp. 443–454. doi: 10.1007/978-3-031-33374-3_35.
24. C. He et al., 'FedML: A Research Library and Benchmark for Federated Machine Learning', Nov. 08, 2020, arXiv: arXiv:2007.13518. Accessed: Feb. 10, 2024. [Online]. Available: http://arxiv.org/abs/2007.13518
25. H. Wang, M. Yurochkin, Y. Sun, D. Papailiopoulos, and Y. Khazaeni, 'Federated Learning with Matched Averaging', Feb. 15, 2020, arXiv: arXiv:2002.06440. Accessed: Jun. 22, 2024. [Online]. Available: http://arxiv.org/abs/2002.06440
26. Q. Yang, Y. Liu, T. Chen, and Y. Tong, 'Federated Machine Learning: Concept and Applications', ACM Trans. Intell. Syst. Technol., vol. 10, no. 2, pp. 1–19, Mar. 2019, doi: 10.1145/3298981.
27. T. Wolf et al., 'HuggingFace's Transformers: State-of-the-art Natural Language Processing', Jul. 13, 2020, arXiv: arXiv:1910.03771. Accessed: Aug. 19, 2024. [Online]. Available: http://arxiv.org/abs/1910.03771
28. S. Willium, 'Shakespeare Dataset'. Accessed: May 25, 2024. [Online]. Available: https://storage.googleapis.com/download.tensorflow.org/data/shakespeare.txt
29. X. Yang, Y. Li, X. Zhang, H. Chen, and W. Cheng, 'Exploring the Limits of ChatGPT for Query or Aspect-based Text Summarization', Feb. 15, 2023, arXiv: arXiv:2302.08081. Accessed: Aug. 19, 2024. [Online]. Available: http://arxiv.org/abs/2302.08081
30. A. Rogers, J. Boyd-Graber, and N. Okazaki, 'Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)', in Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), 2023. Accessed: Aug. 19, 2024. [Online]. Available: https://aclanthology.org/2023.acl-long.0.pdf
31. A. Imtiaz, D. Shehzad, F. Nasim, M. Afzaal, M. Rehman and A. Imran, "Analysis of Cybersecurity Measures for Detection, Prevention, and Misbehaviour of Social Systems," 2023 Tenth International Conference on Social Networks Analysis, Management and Security (SNAMS), Abu Dhabi, United Arab Emirates, 2023, pp. 1-7, doi: 10.1109/SNAMS60348.2023.10375405.
32. Shah, A. M., Aljubayri, M., Khan, M. F., Alqahtani, J., Sulaiman, A., & Shaikh, A. (2023). ILSM: Incorporated Lightweight Security Model for Improving QOS in WSN. Computer Systems Science & Engineering, 46(2).
33. Khan, U., An, Z. Y., & Imran, A. (2020). A blockchain ethereum technology-enabled digital content: Development of trading and sharing economy data. IEEE Access, 8, 217045-217056. IEEE.
34. Mahmood, T., Li, J., Pei, Y., Akhtar, F., Imran, A., & Yaqub, M. (2021). An automatic detection and localization of mammographic microcalcifications ROI with multi-scale features using the radiomics analysis approach. Cancers, 13(23), 5916. MDPI.

35. Iftikhar, A., Elmagzoub, M. A., Shah, A. M., Al Salem, H. A., ul Hassan, M., Alqahtani, J., & Shaikh, A. (2023). Efficient Energy and Delay Reduction Model for Wireless Sensor Networks. Comput. Syst. Sci. Eng., 46(1), 1153-1168.