

# Flagging Hate Speech and Cyber-Bullying in Private Chats

Using distributed machine learning and federated learning techniques to preserve privacy when learning from personal data

Aditya Srivastava (CSCI 6502-001)  
Harsh Gupta (CSCI 6502-001)



# Problem Space

- Hate speech and cyber-bullying are significant problems in online interactions
- Most methods of policing objectionable online content only work on open and public data
  - Vetting by human moderators
  - Automated moderation tools

**How to moderate private interactions between users without leaking their data or violating their privacy?**



# Proposed Work

Use **federated learning** for **privacy preserved** machine learning

- Send model to clients instead of fetching data from clients
- Does not require users to share their data
- Distributed **edge computing** for model training
- Learn from **large amounts of data**
- Use **MapReduce** to parallelize model aggregation



# Prior Work

- Federated learning first introduced in 2017 by McMahan et al. <sup>[1]</sup>
- ML paradigm has since been **applied to many different domains**;
  - **Healthcare** (eg. personal IoT devices) <sup>[2]</sup>
  - **Mobile computing** (eg. autocorrect, next word prediction) <sup>[3]</sup>
  - **Finance** (eg. automated credit scoring) <sup>[4]</sup>
- Work on **productionizing FL at scale** has been undertaken by Google and Microsoft <sup>[5, 6]</sup>



# Data

- Text classification problem
  - **Input:** English text
  - **Output:** Binary class (objectionable or not)
- Combine multiple datasets for testing

Dataset	Offensive	Inoffensive	Sample Size
<i>Hate Speech Dataset</i> <sup>[2]</sup>	599	2399	2998
<i>Cyberbullying dataset</i> <sup>[3]</sup>	57581	390552	448133
<i>ConvAbuse</i> <sup>[4]</sup>	758	2515	3283
<i>Hate Speech and Offensive Language</i> <sup>[5]</sup>	20620	4163	24783
<b>Total After Removing Duplicates</b>	<b>49683</b>	<b>213165</b>	<b>262848</b>



# Data (Continued)

- Splits created
  - Full Model
    - ***Train-Eval-Test Ratio:*** 75 : 15 : 15
    - ***Samples:*** 183,993 : 39,427 : 39,428
  - Federated Clients
    - ***Identical and Independently Distributed (IID)***
    - ***Train:*** (# Total training samples)/(# Clients)
    - ***Eval and Test:*** No change
- Stress testing: 7.8 million sentences from Wikipedia.<sup>[12]</sup>



# Infrastructure

- **Google Cloud Platform (GCP)**
  - Server
    - ***Dataproc Cluster with Google Compute Engine (GCE) instances***
      - Master node with **TPU**
    - ***Spark backend for MapReduce operations***
  - Clients
    - ***GCE instances with GPUs***
- **Google Colab** for ML development and testing



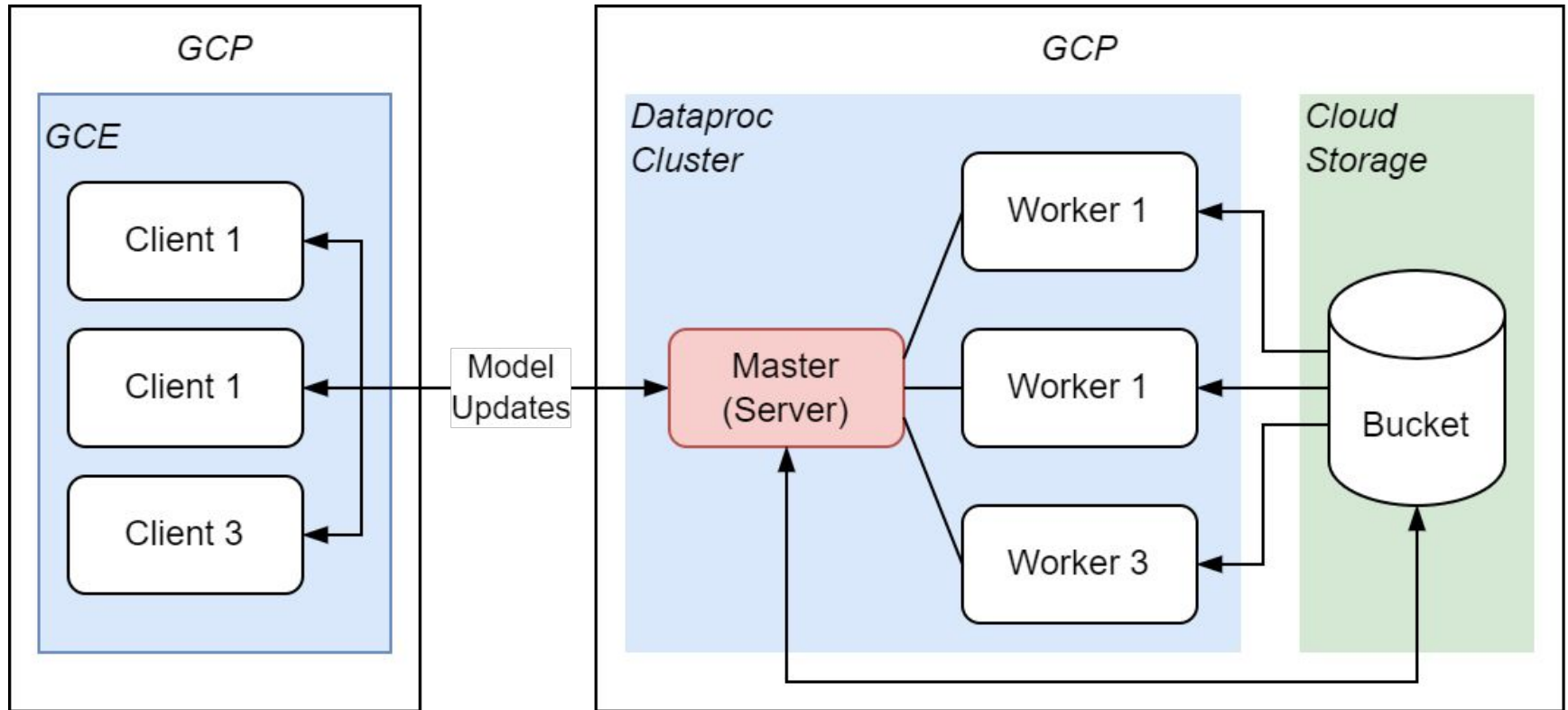
# Tools

- **Linux OS**
- **Python**
  - **Flask** server and client applications
  - **Google Cloud APIs**
  - **Pytorch** for Machine Learning
  - 🤗 **HuggingFace** for pretrained language models
  - **PySpark** for MapReduce functionality





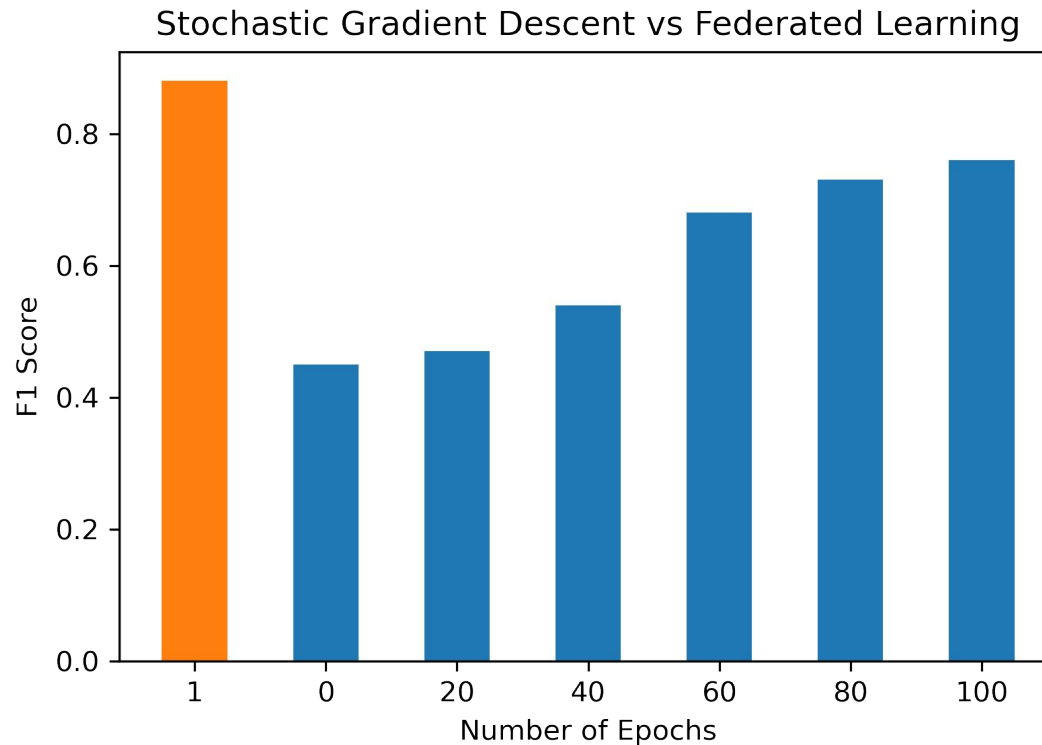
# Architecture



# Evaluations



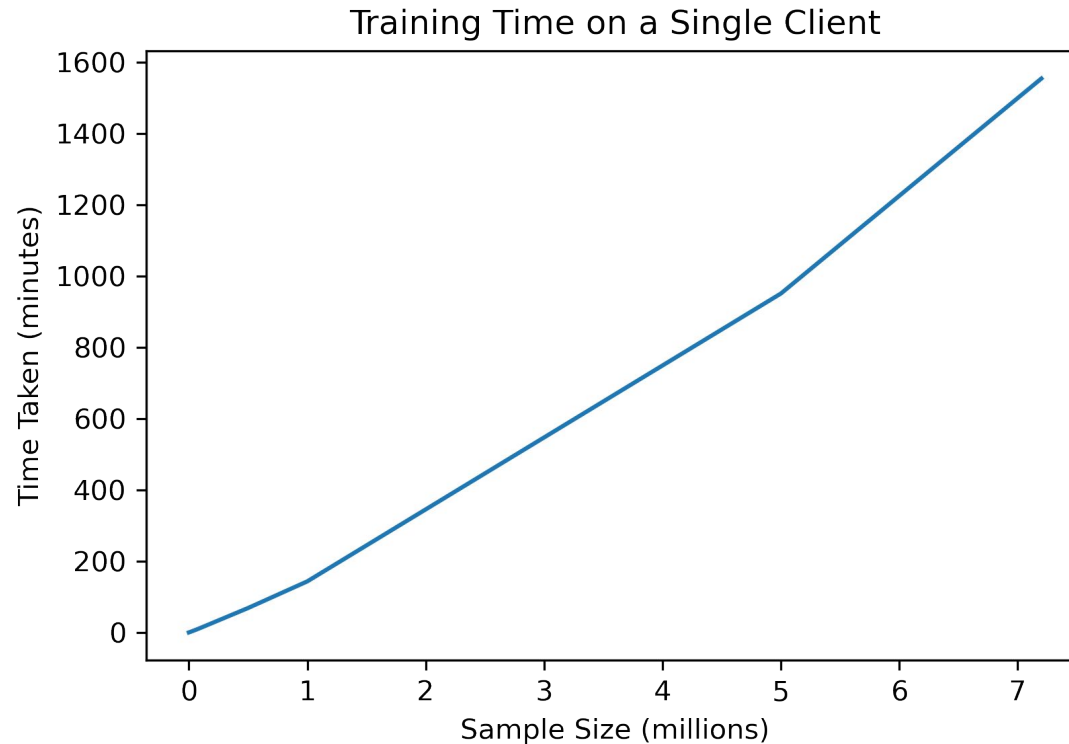
# SGD vs Federated Averaging



- Training performed on **16 clients**
- **Best: 0.88** F1-score for SGD compared to **0.76** for FL



# Impact of Data Size on Training



- Approximately linear curve
- Model loading time adds minimum overhead



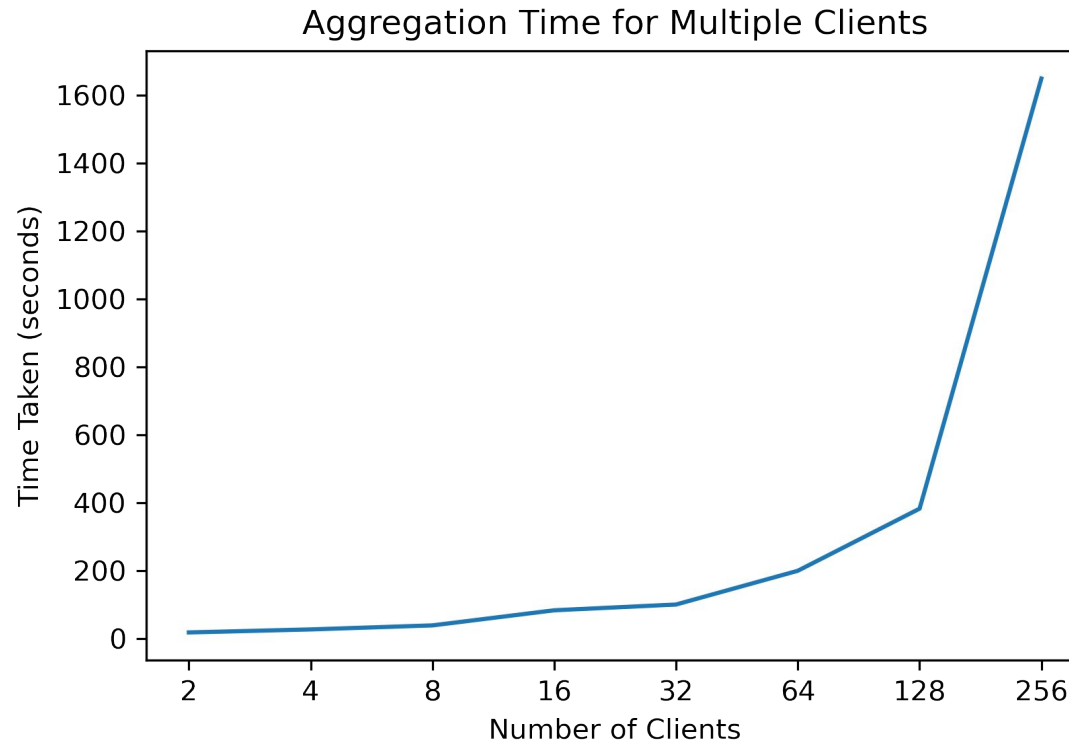
# Impact of #Clients on Training



- #Samples remain constant as #Clients increase
- Exponential decrease as data is split between the clients



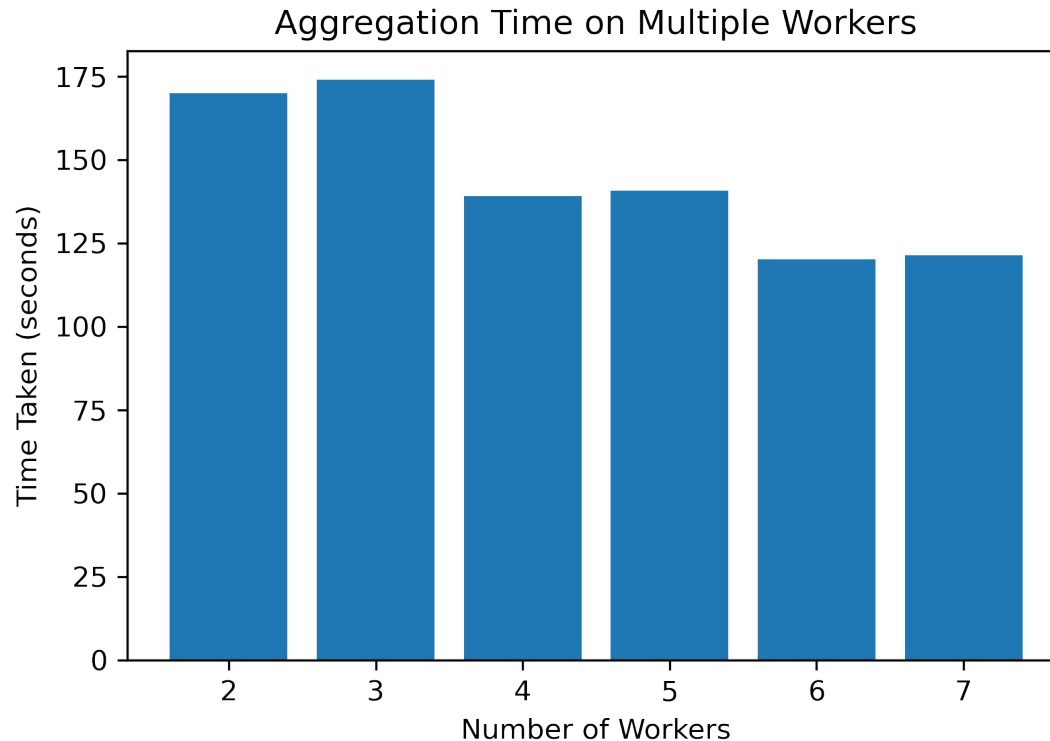
# Impact of #Clients on Model Agg.



- Sharp rise in training time as #Clients increase
- Storage and Memory IO may take increasingly longer time



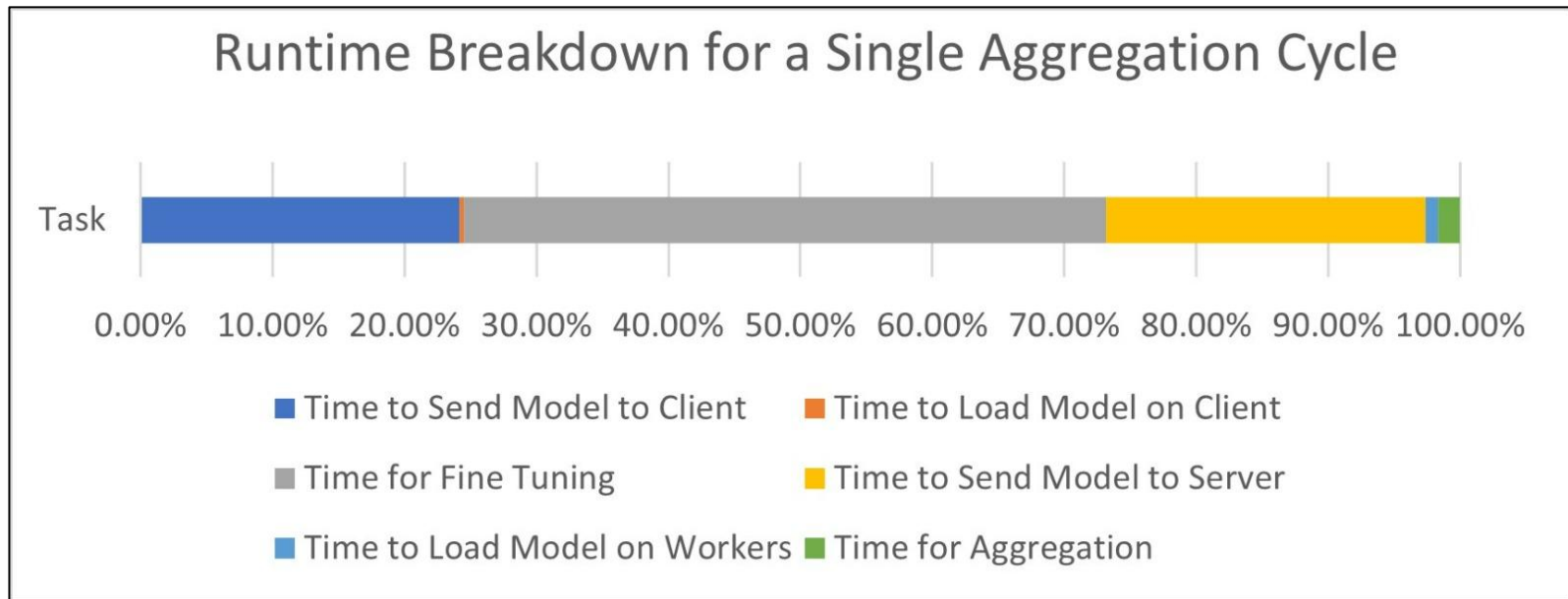
# Impact of #Workers on Model Agg.



- Aggregation time decreases as the #Workers increase
- Odd #Workers do not contribute significantly more



# Runtime Breakdown



- Majority time consumed during finetuning (48%), followed by model transmission and reception (24% each)





# Other Statistics

- **Average latency:** 28.215 ms (b/w server and client)
- **Average throughput:** 1360 samples/second (calculated over 5M samples, 16 clients, 4 workers)



# Key Learnings

- FL Pros
  - High throughput (can handle more data and faster)
  - Data privacy
- FL Cons
  - Infrastructure expenses (compute, storage, etc)
  - High latency (transmitting models to clients and back)
  - Federated Averaging < Gradient Descent



# Challenges and Bottlenecks

- **Challenges**

- Limited compute (\$\$\$) - Colab and GCP
- Synchronising model versions on the client and server
- Models have a large memory footprint
- Needs GPUs/TPUs for efficient training and evaluation

- **Bottlenecks**

- Limited number of workers for MapReduce operations
- Storage IO
- Data transmission between server and clients



# Up Next...

- Evaluating the pipeline on Non-IID data
- Evaluating weighted averaging
- Performing model evaluations using MapReduce routines



# References

1. McMahan et al., *Communication-Efficient Learning of Deep Networks from Decentralized Data* ([link](#))
2. Yuan et al., *A Federated Learning Framework for Healthcare IoT Devices* ([link](#))
3. Hard et al., *Federated Learning for Mobile Keyboard Prediction* ([link](#))
4. OpenMined, *Federated Learning for Credit Scoring* ([link](#))
5. Google, *Towards Federated Learning at Scale: System Design* ([link](#))
6. Microsoft Research, *FLUTE: A Scalable, Extensible Framework for High-Performance Federated Learning Simulations* ([link](#))
7. Shane Cooke, *Labelled Hate Speech Dataset* ([link](#))
8. Saurabh Shahane, *Cyberbullying Dataset* ([link](#))
9. Curry et al., *ConvAbuse: Data, Analysis, and Benchmarks for Nuanced Abuse Detection in Conversational AI* ([link](#))
10. Davidson et al., *Automated Hate Speech Detection and the Problem of Offensive Language* ([link](#))
11. Andrii Samoshyn, *Hate Speech and Offensive Language Dataset* ([link](#))
12. Wikipedia Dataset ([link](#))

