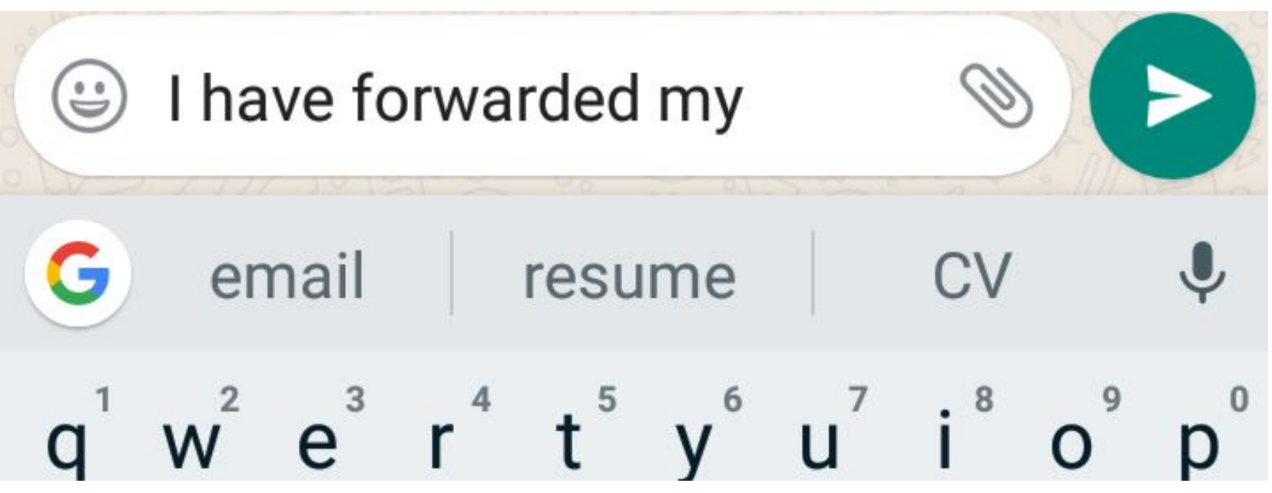
Your Data, Your Rules
Pretraining Federated Text Models

Arjun Singh* & Joel Stremmel*

Motivation

- While training on centralized data is the dominant paradigm in machine learning, there are a variety of limitations to centralized model training, such as compromised user privacy and maintenance of expensive compute resources.
- Federated learning provides a method to train models and conduct experiments on data grouped by individual clients rather than centrally aggregated.
- With mentorship from Google, we develop enhanced text models for federated NLP tasks, such as next word prediction (NWP).



I have forwarded my

Related Work

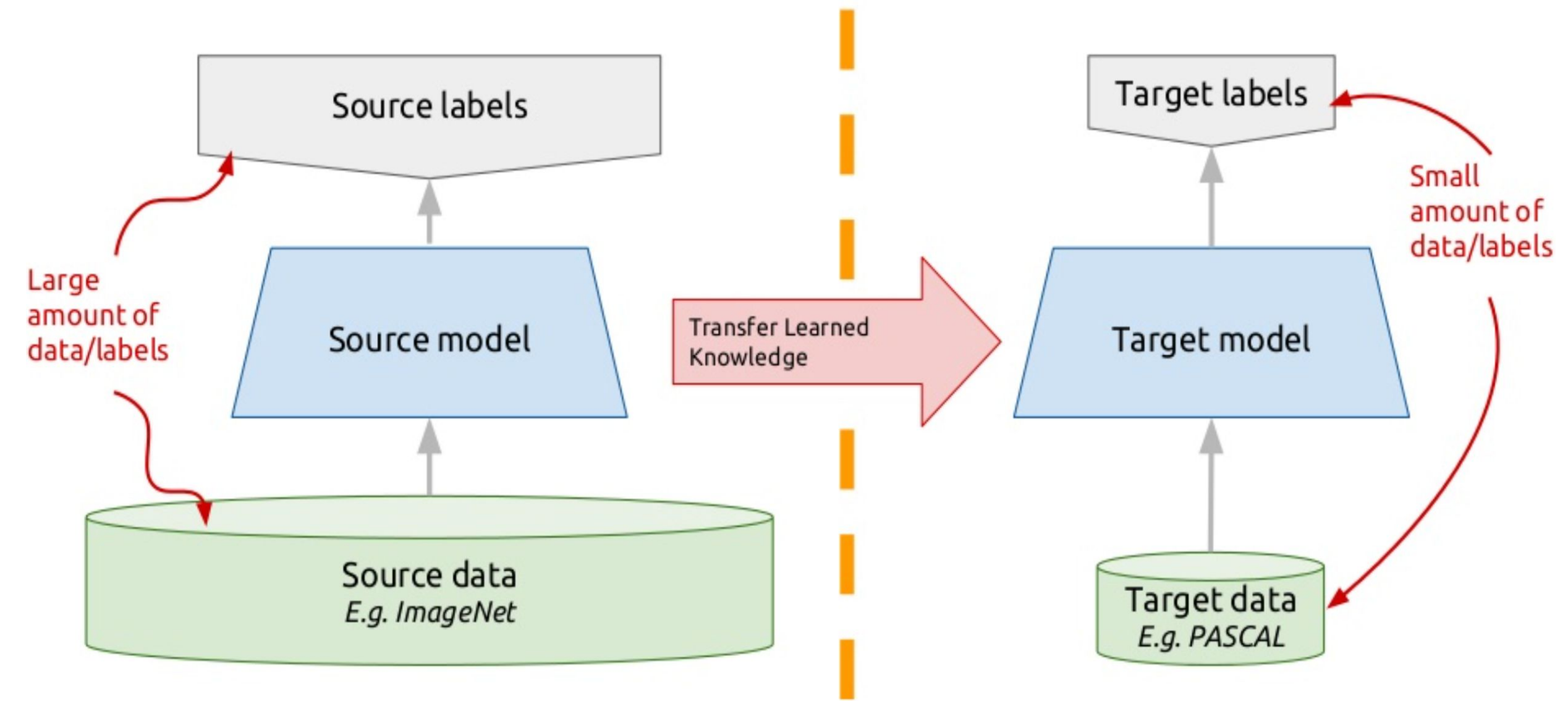
Federated Learning is a decentralized approach for training models on user devices, personalizing the process and ensuring privacy by summarizing local changes and only sending to the cloud a focused update from the local model.

FL trains language models on client devices without exporting sensitive user data to servers and has been successful at tasks like Next Word Prediction for mobile keyboards.

Models tasked to solve complex problems depend on copious data, but getting a ton of labelled data for supervised models can be difficult. Hence transfer learning has proven to be extremely helpful.

"After supervised learning — Transfer Learning will be the next driver of ML commercial success" - Andrew Ng

Transfer learning: idea



https://medium.com/the-official-integrate-ai-blog/transfer-learning-explained-7d275c1e34e2

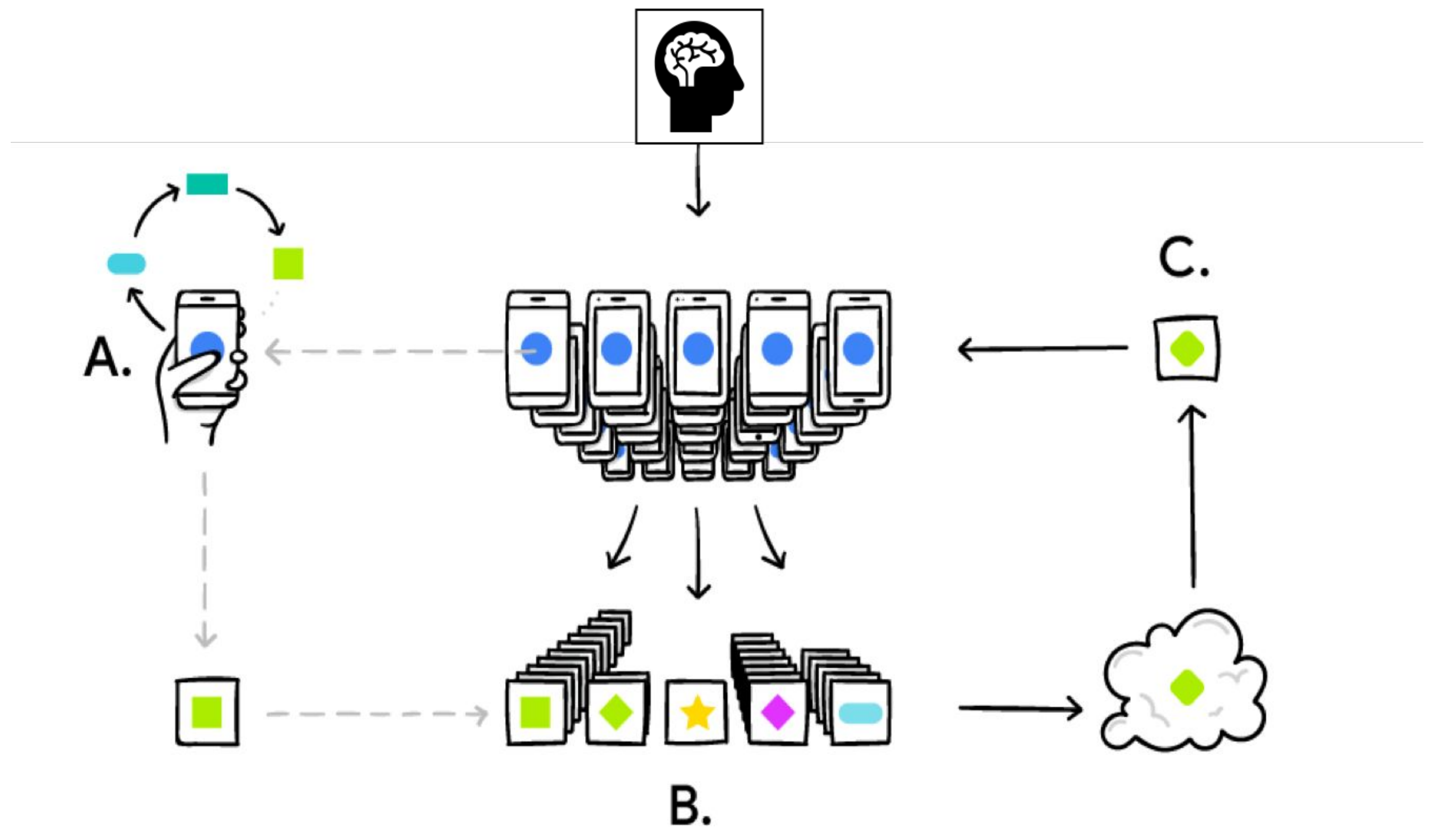
Problem Formulation

In this research we:

Combine the ideas of FL and Transfer Learning to produce pretrained federated models.

Develop and enhance baseline text models using LSTMs for the task of Next Word Prediction in the federated setting.

Introduce the idea of pretrained models and pretrained word embeddings to reduce required training rounds and increase the accuracy of federated text models.



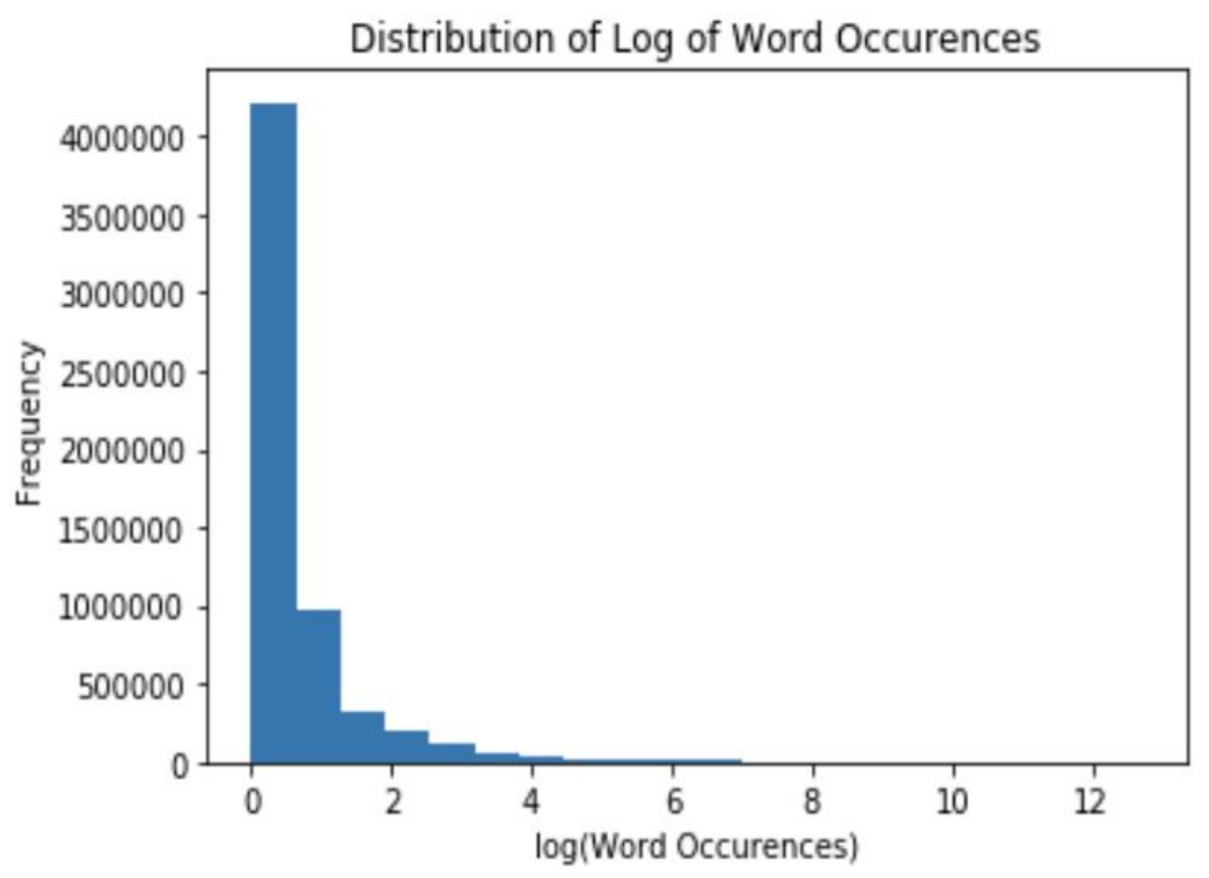
https://ai.googleblog.com/2017/04/federated-learning-collaborative.html

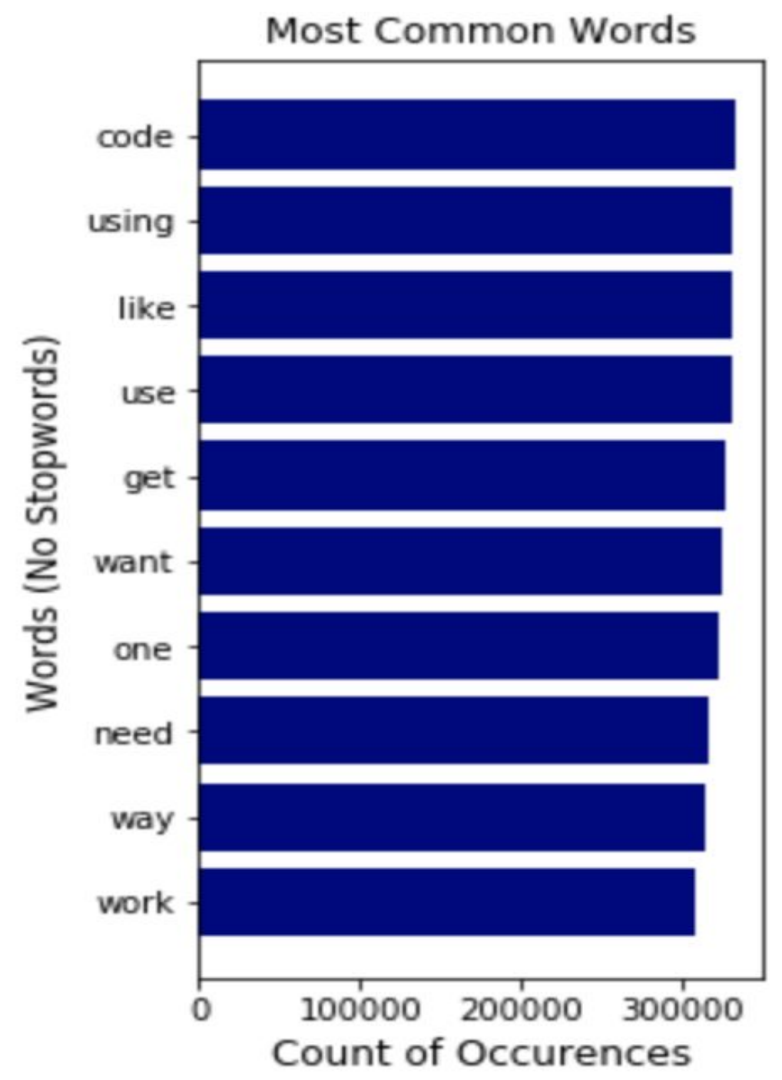
Data

The main dataset used is hosted by Kaggle and made available through the tff.simulation.datasets module in the Tensorflow Federated API. The Stack Overflow data contains the full body text of all Stack Overflow questions and answers along with metadata, and the API pointer is updated quarterly. The data is split into the following sets:

- Train: 342,477 distinct users and 135,818,730 examples.
- Validation: 38,758 distinct users and 16,491,230 examples.
- Test: 204,088 distinct users and 16,586,035 examples.

For the task of pre-training we use the Shakespeare dataset: http://www.gutenberg.org/files/100/old/1994-01-100.zip





Learning Process

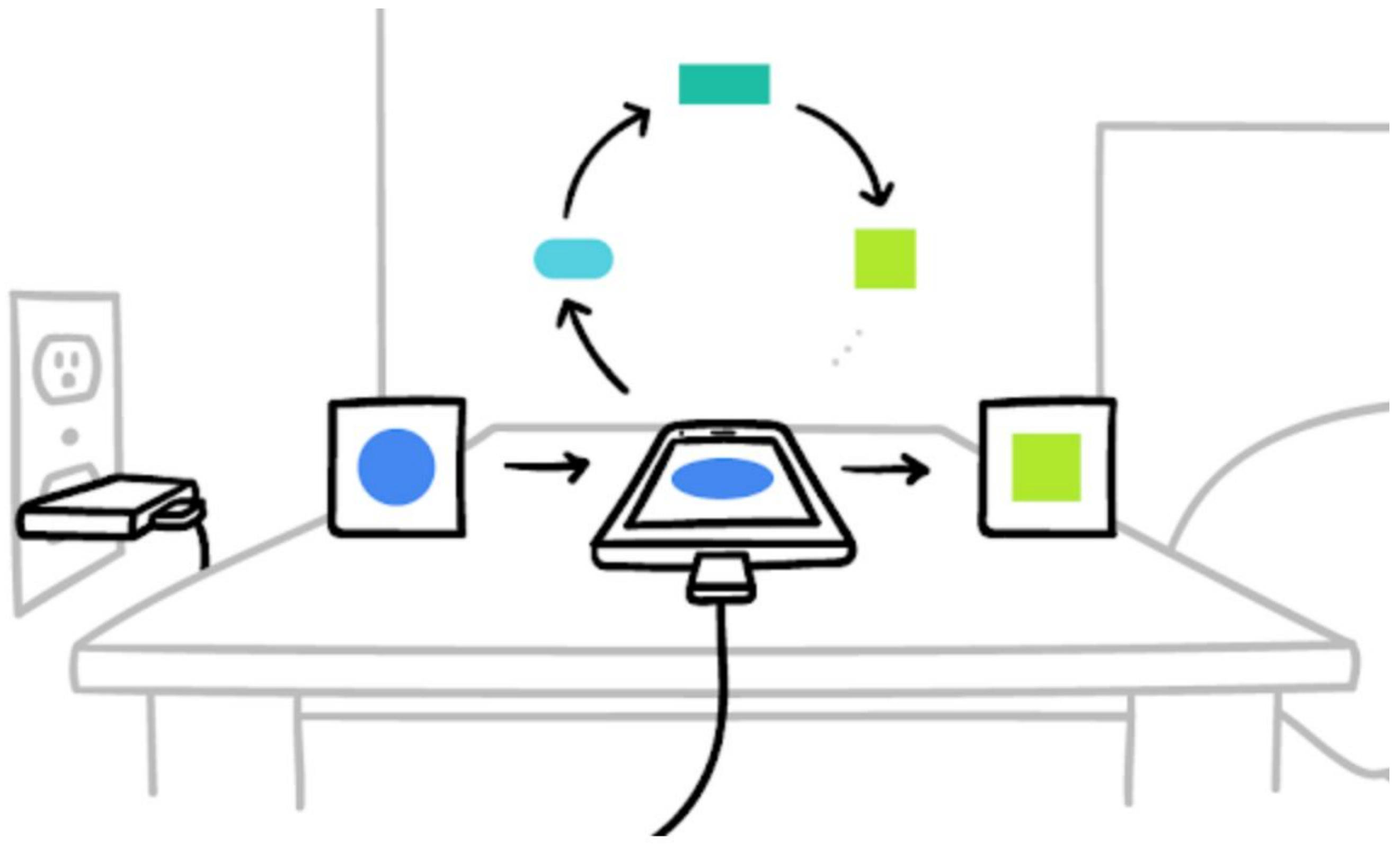
We apply three enhancements to federated training for next word prediction, demonstrating increased accuracy with fewer required training rounds. Our enhancements include:

1. Centrally pretraining then fine tuning in the federated setting.
2. Incorporating pre-trained word embeddings and fine tuning these embeddings in the federated setting.
3. Combining centralized pretraining and pretrained word embeddings with federated fine tuning.

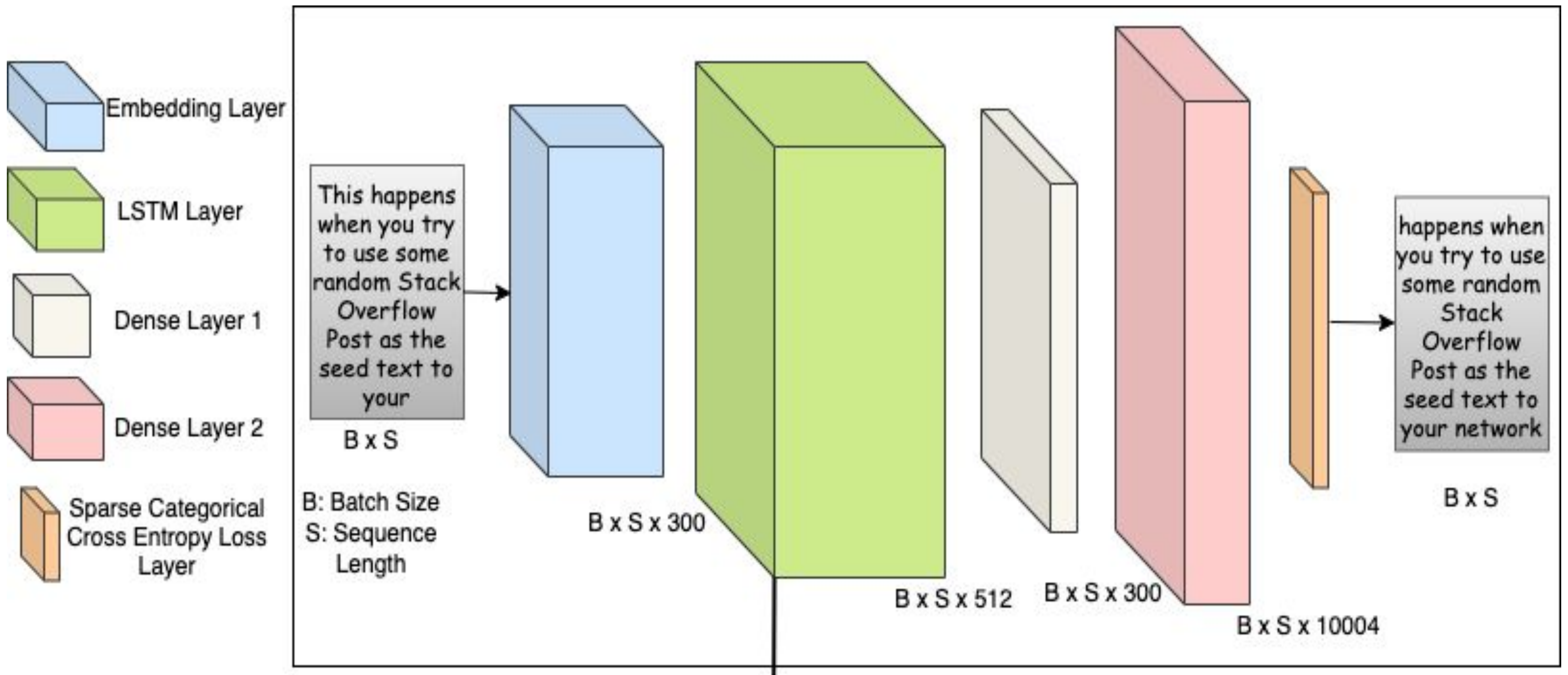
In 2 and 3, we test a variety of pretrained word embeddings and apply PCA and PCA with post-processing (denoted "PP" in the results table) from Raunak's "Simple and Effective Dimensionality Reduction for Word Embeddings."

- **800 training rounds**
- **10 training client datasets per round**
- **Max of 5,000 non-IID text samples**
- **20,000 Validation Set examples**
- **1 million Test Set examples**

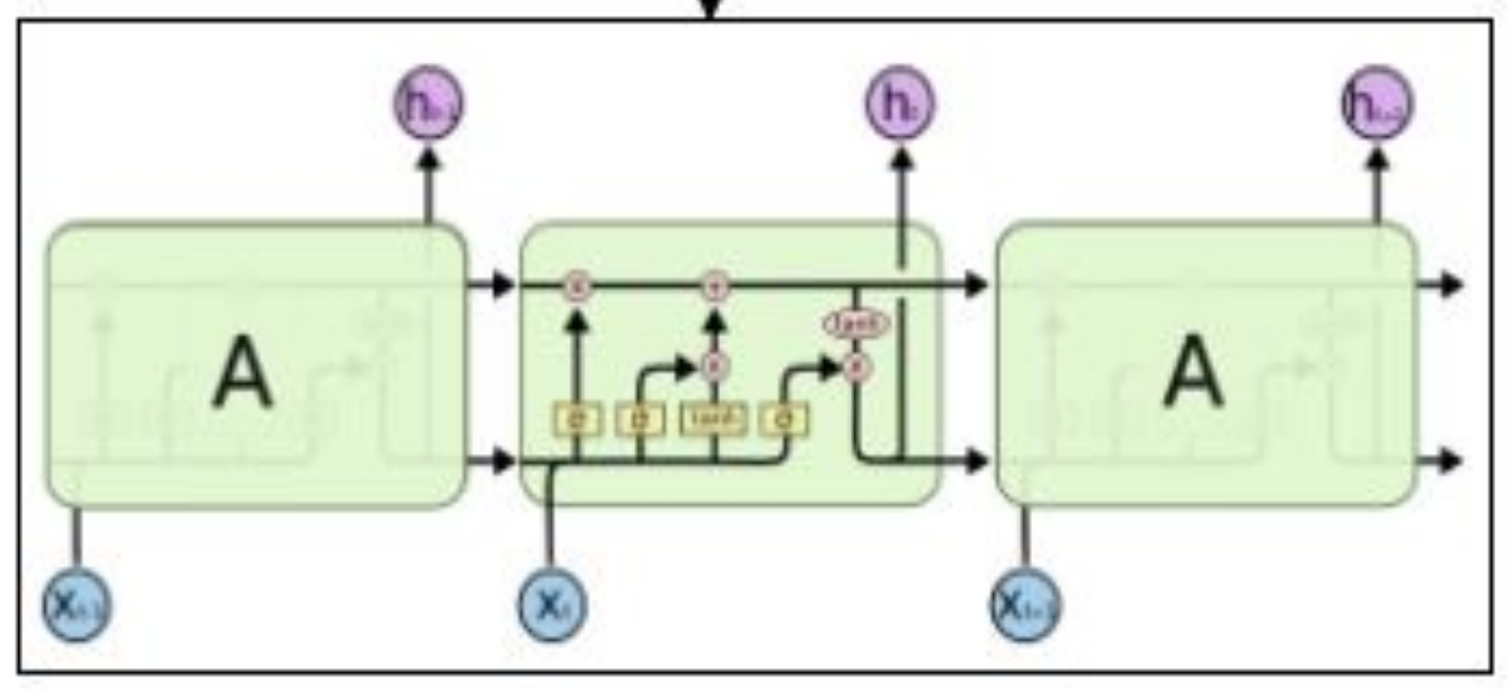
https://github.com/tensorflow/federated/blob/master/tensorflow_federated/python/common_libs/anony



Model



Embedding Layer
LSTM Layer
Dense Layer 1
Dense Layer 2
Sparse Categorical Cross Entropy Loss Layer
This happens when you try to use some random Stack Overflow Post as the seed text to your
B x S
B: Batch Size
S: Sequence Length
B x S x 300
B x S x 512
B x S x 300
B x S x 10004
happens when you try to use some random Stack Overflow Post as the seed text to your network
B x S



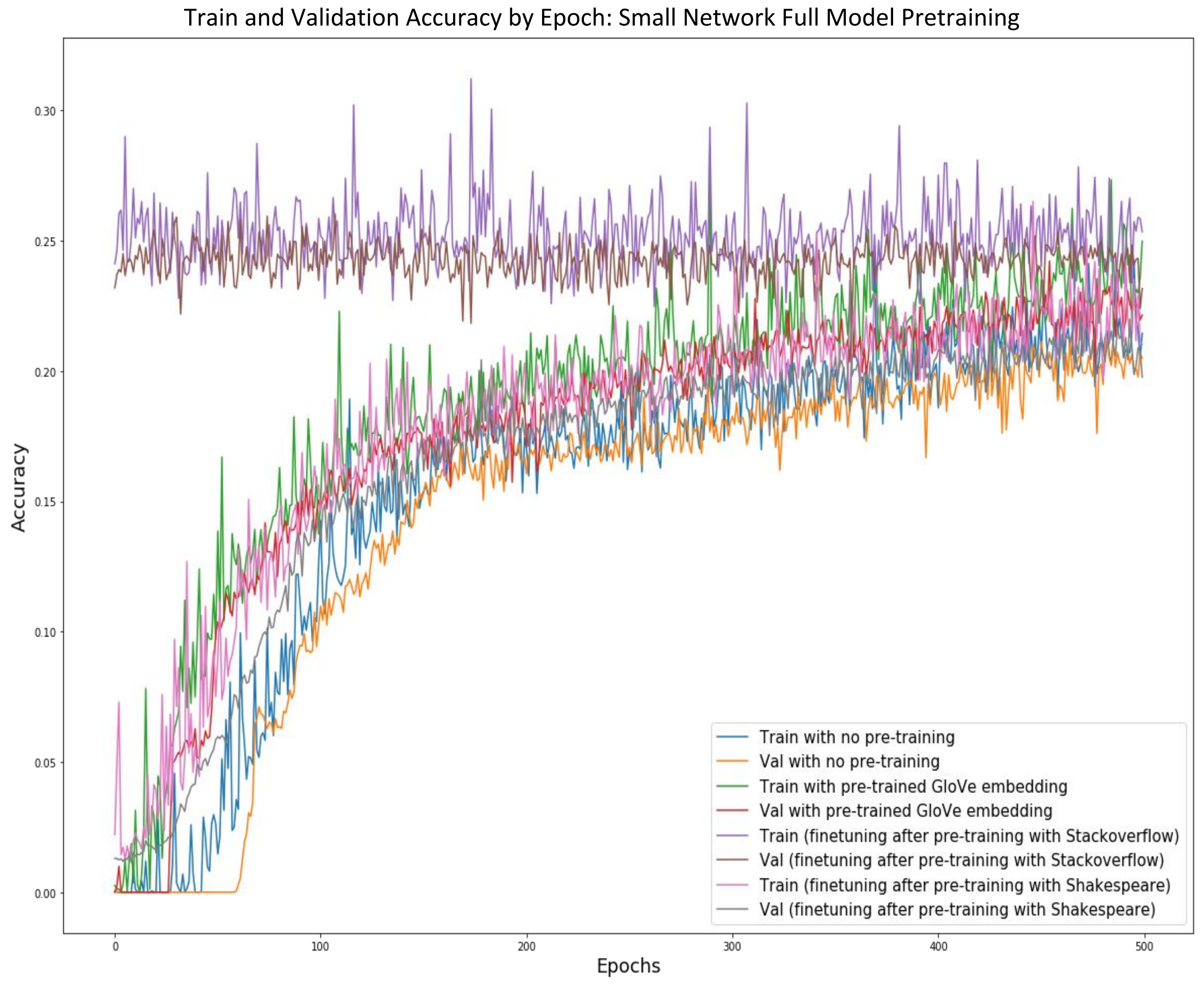
Repeating modules in the LSTM

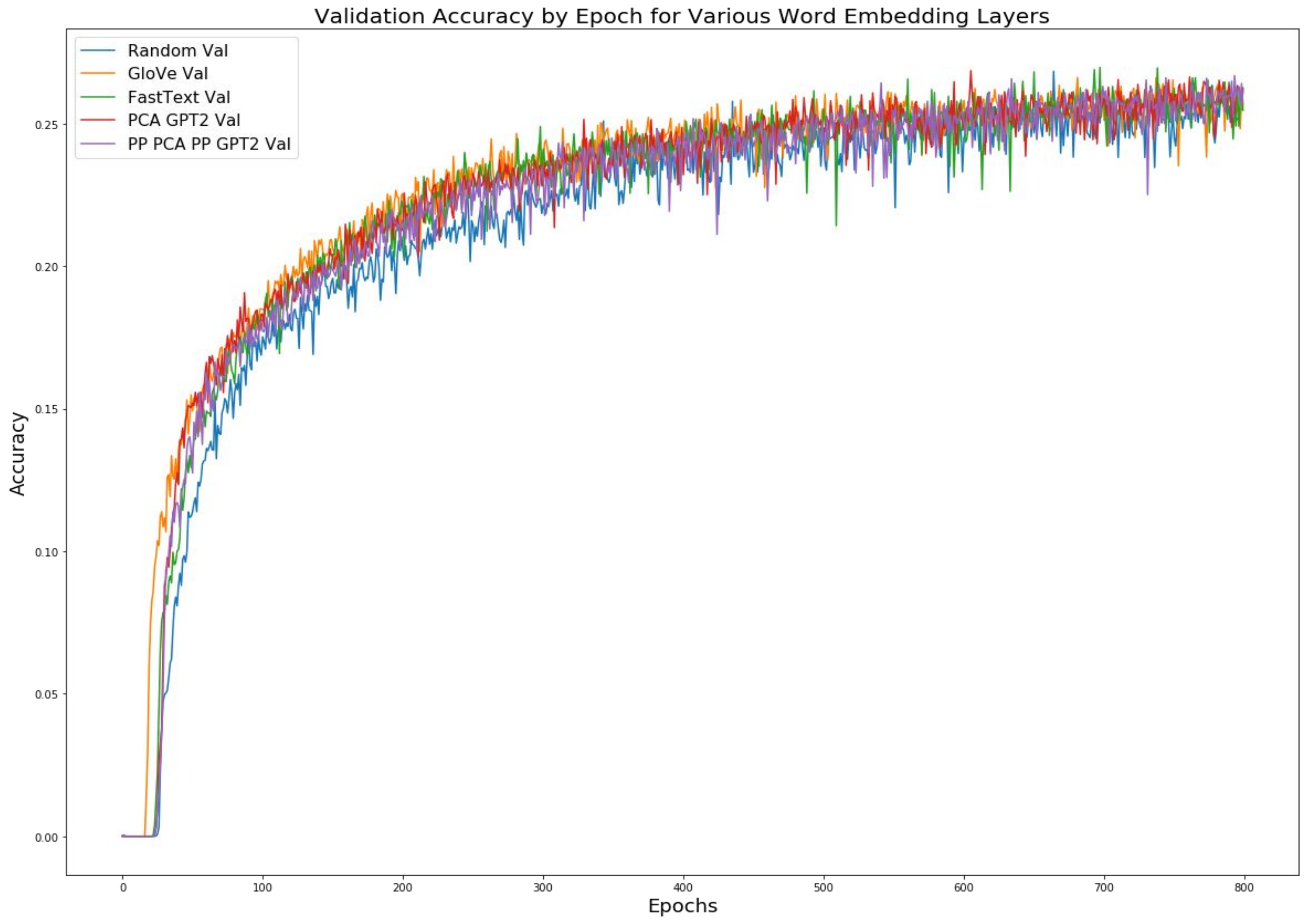
https://colah.github.io/posts/2015-08-Understanding-LSTMs/

Your Data, Your Rules
Pretraining Federated Text Models

Arjun Singh* & Joel Stremmel*

Experiments & Results

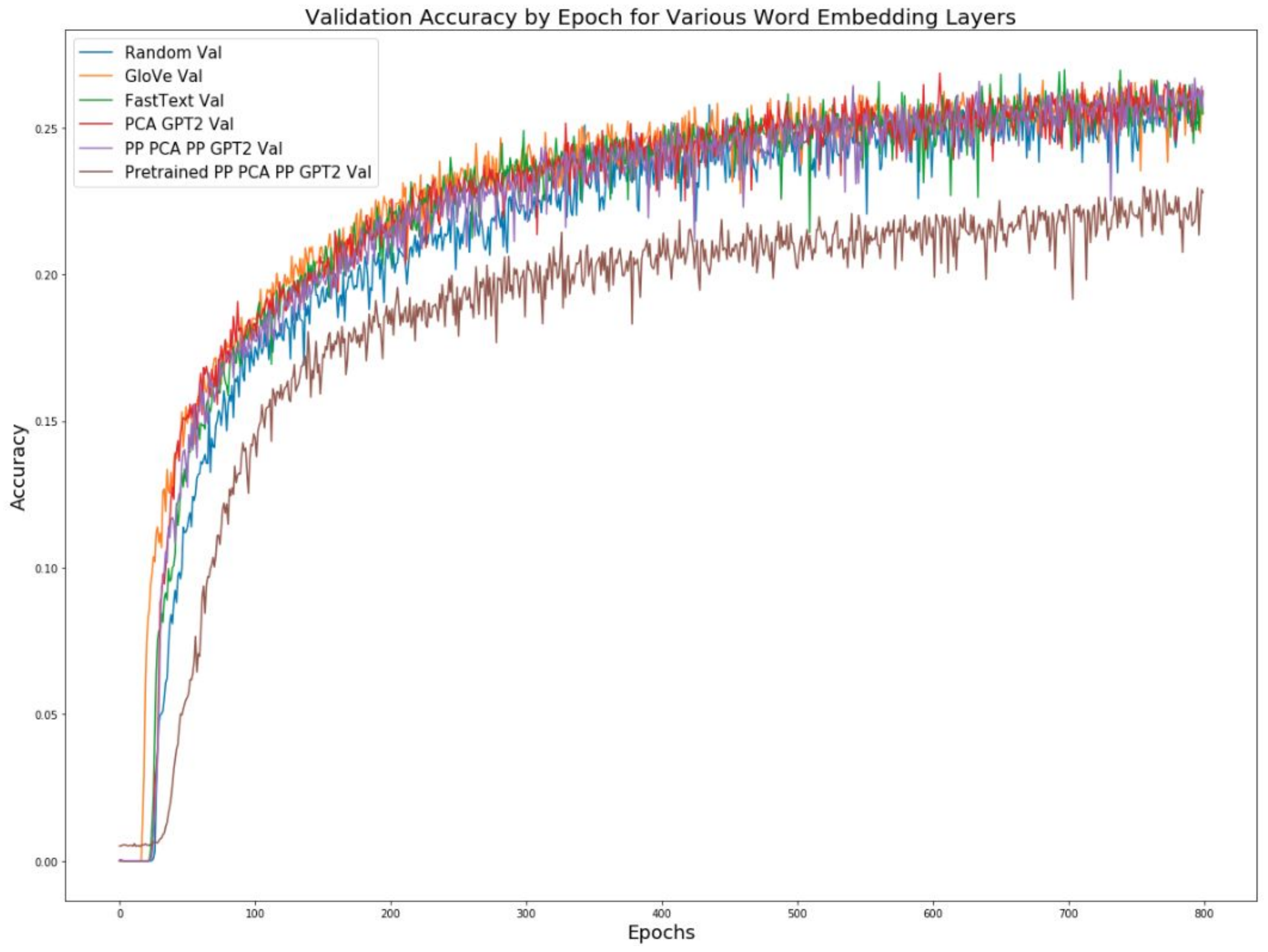




Model	Accuracy	Accuracy No OOV No EOS
Small Random*	**0.2246**	**0.1821**
Small GloVe	0.2269	0.1838
Small PCA FastText	0.2250	0.1823
Small PCA FastText + PP	0.2285	0.1852
Small PCA GPT2	0.2293	0.1859
Small PCA GPT2 + PP	0.2262	0.1834
Large Random*	**0.2485**	**0.2086**
Large GloVe	0.2557	0.2162
Large FastText	0.2548	0.2137
Large PCA GPT2	0.2522	0.2118
Large PCA GPT2 + PP**	**0.2569**	**0.2169**

Model	Parameters	Weights Size (MB)
Small	2.4M	9.6
Large	7.8M	31.3

***Baseline approaches**
****Best performing model**



PRETRAINED MODELS:

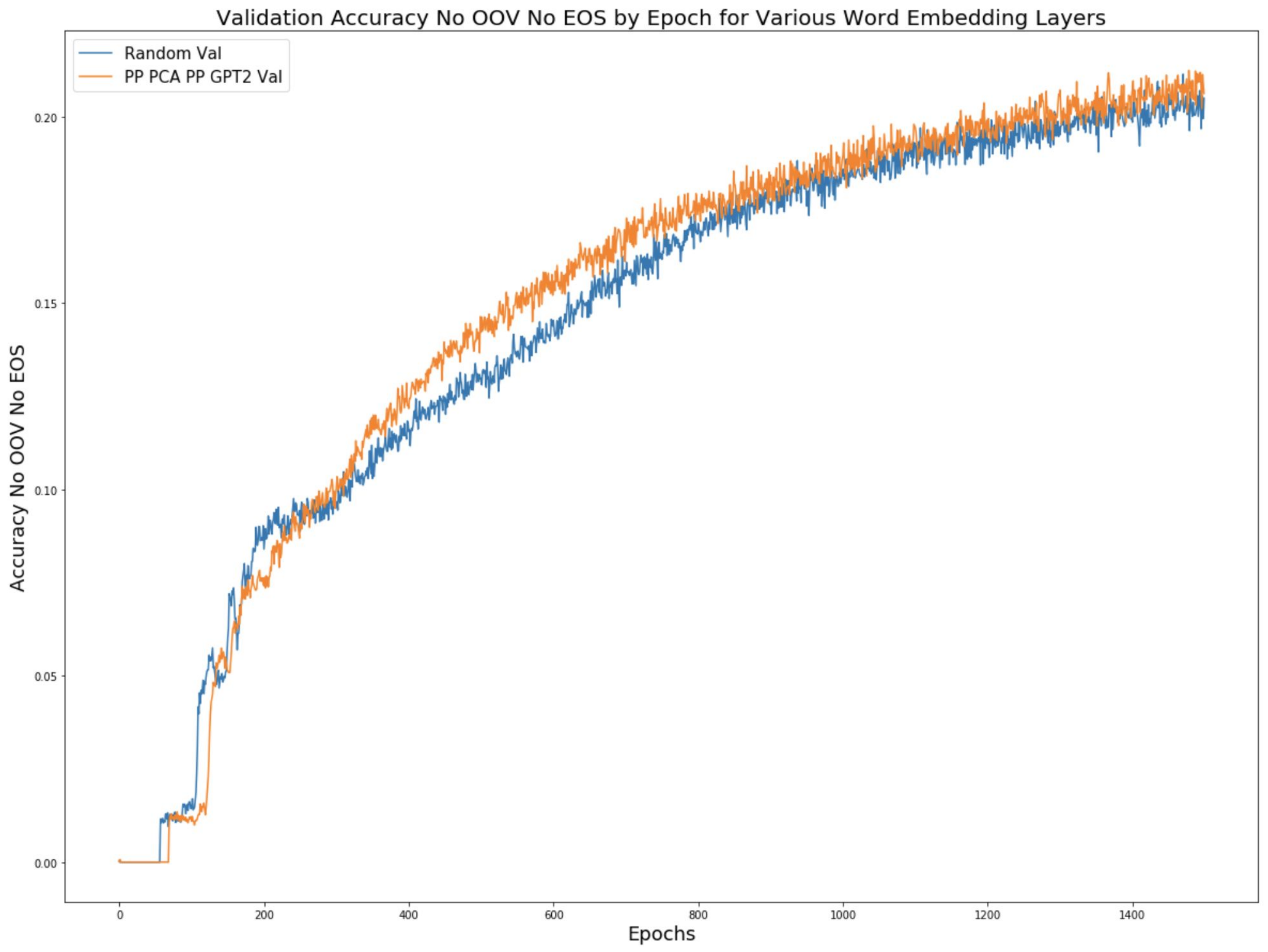
- Model centrally pretrained using (a) Shakespeare Data and (b) Stack Overflow data
- Pretrained model, fine-tuned in federated style using Stack Overflow data (distinct samples)
- Central pretraining outperforms no pretraining for both cases; Central pretraining using SO outperforms SP
- Downstream fine-tuning adds value for SP based pretraining, but not for SO based fine-tuning

PRETRAINED WORD EMBEDDINGS:

- Pretrained embeddings achieve the same level of accuracy in fewer training rounds than random embeddings for the large network as shown in the learning curves above.
- Pretrained embeddings slightly outperform random embeddings in later training rounds, though random embeddings start to “catch up” on accuracy.
- Pretrained embeddings outperform random embeddings for the larger network architecture by over a half percent on our test set of 1m text samples, as shown in the table above.
- Smaller networks demonstrate little to no increase compared to random embeddings.

PRETRAINED MODELS + PRETRAINED WORD EMBEDDINGS:

- Our experiments demonstrate how to pretrain a model centrally and fine tune it in the federated setting, however…
- Pretraining the large network on Shakespeare and fine tuning on Stack Overflow with the best word embedding method performed worse than federated training with random and pretrained embeddings



PRETRAINED WORD EMBEDDINGS COMAPRED TO GOOGLE’S MODEL

We compare our best embedding approach to randomly initialized word embeddings using the client sampling strategy and model architecture from ”Adaptive Federated Optimization.”

We use the Adam defaults for optimization.

Future Work

- We suspect the following could yield better results for full model pretraining and aim to achieve that as an immediate future work item:
    - A dataset more like Stack Overflow than Shakespeare
    - Learning rate optimization
    - Federated instead of central pretraining
- Current research for NWP is limited to the Stack Overflow federated dataset, because the other federated text dataset, Shakespeare, currently only deals with character level tasks.
- We aim to evaluate federated pretraining and federated fine-tuning, in addition to the current experiments using central pretraining and federated fine-tuning.
- While small models seem to limit accuracy, further experimentation with small model architectures may yield improved results.

References

- H. Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, Blaise Aguera y Arcas. “Communication-Efficient Learning of Deep Networks."
- Keith Bonawitz, Hubert Eichner, Wolfgang Grieskamp, Dzmitry Huba, Alex Ingerman, Vladimir Ivanov, Chloe Kiddon, Jakub Konecny, Stefano Mazzocchi, H. Brendan McMahan, Timon Van Overveldt, David Petrou, Daniel Ramage, Jason Roselander. “Towards Federated Learning at Scale: System Design.”
- Andrew Hard, Kanishka Rao, Rajiv Mathews, Swaroop Ramaswamy, Francoise Beaufays Sean Augenstein, Hubert Eichner, Chloe Kiddon, Daniel Ramage. “Federated Learning for Mobile Keyboard Prediction.”
- Vikas Raunak, Vivek Gupta, Florian Metze. "Effective Dimensionality Reduction for Word Embeddings."
- Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever. "Language Models are Unsupervised Multitask Learners."

Acknowledgements

Shoutout to our wonderful mentors, Keith Rush and Peter Kairouz, for their insights and helpful tips throughout the project, as well as to Brendan McMahan, for this amazing opportunity to collaborate with the Google FL team!

1 & 1 make 11