Sentiment detection with FedMD: Federated Learning via Model Distillation

Plamena Tsankova and Galina Momcheva

Varna Free University Chernorizets Hrabar, Yanko Slavchev 84, Chayka Resort,

Varna, Bulgaria
{182831005, galina.momcheva}@vfu.bg

Abstract. Federated learning is a distributed machine learning technique in which client devices train models locally without sharing any data, except for parameter changes, which get aggregated to a central model. This privacy-preserving approach has a huge potential for reconciling the need for large Deep Learning datasets with the increasing sensitivity of data ownership. Our paper takes the novel FedMD (Federated Learning via Model Distillation) algorithm and applies it for the first time to the field of Natural Language Processing. The results are promising with regards to solving the data heterogeneity and model personalization challenges by introducing client-specific models and collaborative learning realized through model distillation. The resulting small gap between the FedMD results and the non-FedMD implementation is compensated by the smaller amount of training data for the FedMD models and the successful preservation of privacy for locally available data.

Keywords: Federated Learning · FedMD · Sentiment classification.

1 Introduction

Federated learning (FL) is a machine learning technique (developed in 2016) in which training data remains private and does not leave the client device. In contrast to traditional Machine Learning (ML), where all data is centrally available, no data is shared in FL except for locally computed updates which are sent to the server by each device. These updates are aggregated by the server into a final global model. This approach is praised for its security-preserving nature. It builds on distributed machine learning principles but goes beyond them in terms of privacy and performance when dealing with the real-world challenges of heterogeneous data and devices.

There are several reasons why FL is becoming increasingly popular among users and companies. One reason is the potential of this technique for reducing data privacy issues. A second reason relates to the widespread availability of

powerful computing devices (e.g. mobile phones, tablets). A third reason comes from advances in Deep Learning (DL) [10], which can now be enjoyed by data-sensitive industries with the help of FL.

Despite the above-mentioned FL benefits, there are still numerous bottlenecks [6]. The main challenges relate to the presence of heterogeneity with regards to the data and the client devices. A very recent approach that tackles both topics was put forward by [7]. They challenge the existing setup with multiple uniform local models and introduce a client-specific model architecture. The resulting framework is named FedMD: Federated Learning with Model Distillation. FedMD builds upon learning from public and private data, followed by a collaborative knowledge exchange, all carried out within a FL setting with uniquely designed client models. Both transfer learning and knowledge distillation are embedded in the framework, as they ensure the knowledge transmission in and between the phases.

Inspired by [7], this paper answers their call for applying FedMD to the field of Natural Language Processing (NLP). For the experiment we chose a prominent NLP problem: sentiment classification of tweet messages on the Sentiment140 dataset. This challenge allows us to experiment not only with the learning setup, but also with different model architectures to benchmark model performance. This will enable a review of multiple variations of Long-Short-Term-Memory (LSTM) neural networks. which have a proven track record for sentiment classification [13, 3]. To summarize, the research goal of this paper is twofold: 1) to prove the feasibility of implementing FedMD on an NLP challenge and 2) to compare and analyze differences in the participating models' performance.

The choice of research focus is motivated by its multiple implications for both academics and practitioners. First, this paper adds to a body of research focused on developing algorithms with built-in privacy protections. Second, successful FedMD experiments help clients use DL when they are not in a position to share their data for legal or other reasons, or who have specific requirements such as data portfolio that requires tailored modelling. Third, the tweet sentiment classification results can support for example a physician who wishes to assess the mental health of patients through the sentiment detected in their social media messages, without being intrusive.

2 Literature review

The term FL was pioneered in 2016 in the research paper by [8]. The authors already point out the unbalanced and non-IID (independent and identically distributed) data, as well as the massive number of participating devices with varying trustworthiness and potentially high communication costs as the defining challenges of the emerging field [8]. Questions and concerns over varying

amounts of user data drawn from different distributions, differences in bandwidth and computational power, as well as communication cost and privacy risks have dominated the research field over the past years [6]. Solutions to these challenges are still in the making and require expertise from different fields and a good understanding of the complex nature of FL.

It should be noted that FL bears resemblance to distributed machine learning, which stands for the practice of training a model on multiple devices. Similarly to FL, it covers numerous aspects such as the distributed storage of the training data, the distributed execution of the computing tasks and the distributed handling of the results aggregation [14]. A way to sum up the differences is that FL is " decentralized training over decentralized data" [12]. FL acknowledges the existing differences among participants so the challenge shifts away from having the most efficient distributed architecture to train a model to doing so while accounting for related data and system heterogeneity.

2.1 Standard Federated learning architecture

In the following the original FL setup suggested by [8] will be presented. This gives a reliable starting point for the FL exploration.

For starters, the presence of one centralized server and multiple edge devices or clients is required. In the first stage the central server selects a model type to be trained, which is uniform for all clients. A second decision narrows down the range of participating devices based on eligibility criteria such as the presence of strong wi-fi signal, sufficient battery levels and idle device state. This ensures that device owners will not be negatively impacted. The initial model is sent to all, or a selection of, participants and trained on their private data. During the training, updates with new knowledge are sent back to the server, where they are aggregated and incorporated into a global model. This stage has attracted a lot of research and there are various alternatives on how to securely and optimally integrate the local updates (e.g. [1]). The enhanced global model is subsequently transmitted to the user devices to replace the initial model and the entire cycle is repeated. This practice has sparked a debate on the tradeoff between maximizing the performance of the global model at the expense of diminishing personalization for the local models [6]. This debate also gave rise to alternative approaches and solutions based on meta-learning [5], multi-task learning [11] and FedMD [7].

2.2 The FedMD framework

As mentioned in the previous section, the default FL setting has only one model type for the clients and the server. This makes communication, results integration and model replacement less troublesome. In the following, the alternative setup called FedMD, as developed and implemented by [7], will be presented (see Fig. 1).

There are m clients in total. Each private dataset is expressed through $\mathcal{D}_k := \{(x_i^k, y_i)\}_{i=1}^{N_k}$ and may not come from the same global distribution. The public dataset is accessible to all devices and notated as follows: $\mathcal{D}_0 := \{(x_i^0, y_i^0)\}_{i=1}^{N_0}$. Each client has its own model with a unique architecture. Data, model design and hyper-parameters remain private and are not shared in any form during the learning process. Each model is trained initially on data $\mathcal{D}_0$ D0 and $\mathcal{D}_k$. The main purpose of FedMD is to improve the individual models' performance beyond training on local and public data through collaborative learning [7]. To prevent data leakage during collaboration, knowledge gains (updates) are transformed to a standard format. A central server computes a consensus from these updates and shares it with the clients. A translator adds a layer of standardization for the unique models' outputs and is implemented with the help of knowledge distillation by aggregating all models class scores [7].

Algorithm 1: The FedMD framework enabling federated learning for heterogeneous models.

Input: Public dataset $\mathcal{D}_0$, private datasets $\mathcal{D}_k$, independently designed model f_k, $k = 1 \dots m$,
Output: Trained model f_k
Transfer learning: Each party trains f_k to convergence on the public $\mathcal{D}_0$ and then on its private $\mathcal{D}_k$.
for j=1,2...P **do**
  Communicate: Each party computes the class scores $f_k(x_i^0)$ on the public dataset, and transmits the result to a central server.
  Aggregate: The server computes an updated consensus, which is an average $\tilde{f}(x_i^0) = \frac{1}{m} \sum_k f_k(x_i^0)$.
  Distribute: Each party downloads the updated consensus $\tilde{f}(x_i^0)$.
  Digest: Each party trains its model f_k to approach the consensus $\tilde{f}$ on the public dataset $\mathcal{D}_0$.
  Revisit: Each party trains its model f_k on its own private data for a few epochs.
end

Fig. 1. FedMD Architecture by [7]

The FedMD framework consists of 3 identifiable stages. In the first one all clients are trained on the public data. This is a preventive measure to ensure that the resulting models are statistically solid and robust against large variations in their private data. Upon convergence, the models use transfer learning to train on their private data. The end of this stage sets a baseline for comparing performance. In contrast to the standard FL process, no updates or data are

sent to the central server during private data training. The final stage facilitates collaborative learning among all participating devices via model distillation. Here the independent models share their learned knowledge by sending raw prediction scores (i.e. the logits or class scores) to the server. In every iteration an alignment dataset is generated randomly from the public dataset, which serves as basis for communication between models. Upon receiving all predictions by all clients, the server averages them and sets them as the new training target. Subsequently, the models are trained to approach this consensus. To wrap up this stage, the models train on their own data for a few extra rounds. The results achieved after this stage form the second baseline for model performance. An optional experiment is included as well - by training the models on all available public and private data pooled together one can establish the theoretical upper boundary limit per model and compare it to the final FedMD outcomes.

[7] implemented FedMD on two datasets: MNIST and CIFAR 100 for image classification. They successfully boosted accuracy by on average 20% after the collaborative phase.

3 Research design

3.1 Dataset

The choice of dataset is an important decision with regards to the reproducibility of results. This motivated the selection of one of the five FL benchmark datasets, put forward by LEAF [2]. The choice of available datasets limited the choice of NLP challenge to binary sentiment classification. The Sentiment140 dataset by Stanford University contains over 1.6 million tweets from around 650 000 users. It comes close to a realistic FL scenario, as it is generated by multiple users.

Pre-processing. Working with tweets is conceptually different to other NLP tasks. The social nature of this medium and its short format encourage users to create as many tweets as possible, while grammar, spelling and style rules are not as strict as for larger body of texts like blog posts or articles. To cleanse the data, standard pre-processing techniques were applied such as stemming and the removal of stop-words, internet links, hashtags and references to other twitter users.

Private and public data split. The original intention was to derive the private data from single frequent users, each one of whom would be a client in FedMD. This approach was, however rejected upon engaging with the data. The user with the largest number of tweets has less than 550 tweets, and the 5th most frequent user has only around 280. These numbers are not enough to train a neural network in a meaningful way. FedMD is most useful when applied to big amounts of data, enough so that it reasonably justifies the existence of customized

models. Ideally, the dataset should have contained several thousand tweets from the same frequent twitter users.

The above-mentioned complication was overcome by creating an artificial split in private/public data. Private data profiles were generated by sorting the dataset by username and splitting the data into 20% public and 80% private data (or 8% for each of the 10 clients). This approach means, however, that a specific user's behaviour may not be well-learned due to the tiny number of tweets per user compared to the large amount of data from other users.

3.2 The FedMD framework

The main design of the FedMD framework was maintained for this experiment. The 4 training rounds present in the original model setup are also implemented in this experiment.

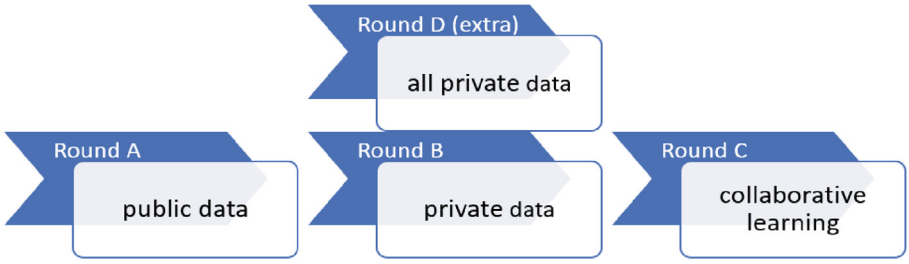

Fig. 2. FedMD implementation phases

3.3 Model architecture and parameters

Model heterogeneity is a defining feature of this empirical study. The experiment does not aim to showcase the best possible model architecture for detecting sentiment, but it is capable of comparing different neural network compositions through spot-checks. It was felt that expanding the original task, testing FedMD on NLP tasks, to testing different model architectures could lead to interesting practical insights.

We prepared 5 unique model architectures (see Table 1) and varied the origin of the embedding weights to end up with 10 models in total. Each model in the chart below has two versions: one with word embedding initiated randomly and learned during training and a second where embedding weights come from a GloVe pre-trained model [9]. This experiment allows us to compare overall performance and test the assumed superior performance of the GloVe model.

Table 1. Neural network models architecture.

A (simple NN)	B (simple LSTM)	C (convoluted LSTM)	D (bidirctional LSTM)	E (Recurrent Dropout LSTM)
Embedding	Embedding	Embedding	Embedding	Embedding
GlobalAveragePooling	LSTM	Dropout	Bidirectional LSTM	LSTM
Dense	Dense	Conv1D	Dropout	Dense
		MaxPooling1D	Dense	
		LSTM		
		Dense		

All models start with an embedding layer and finish with a dense layer, activated with a softmax function in order to output the class scores of each sentiment category. Model categories B-E present different variations to the LSTM neural network, whereas category A is a simple neural network implementation. The choice for LSTM was motivated by its ability to handle sequential data (i.e. tweets as sequence of words). LSTMs introduce 'memory' in the neural network, which allows to save the context of words and capture long- and short-term dependencies between words [4].

4 Results

4.1 Phrase A results

As a reminder, public data comprised 20% of the initial dataset. The model validation accuracy achieved after training with the public data is summarized in Fig. 3.

The models' performance is very similar with overall accuracy between 74% and 76%. This is surprising considering the different model architectures and parameters used. It suggests that further performance gains are unlikely to be achieved through different models, but rather with better data or data preprocessing.

The models with randomly initiated embedding weights vs. pre-trained GloVe word embeddings are also very close. It is possible that the GloVe embeddings provide little additional value due to the nature of tweets - they are short, have little context and contain misspelled and shortened words, all of which stands in general contrast to the GloVe training data and logic, which was based on structured texts coming from e.g. Wikipedia or Common Crawl [9].

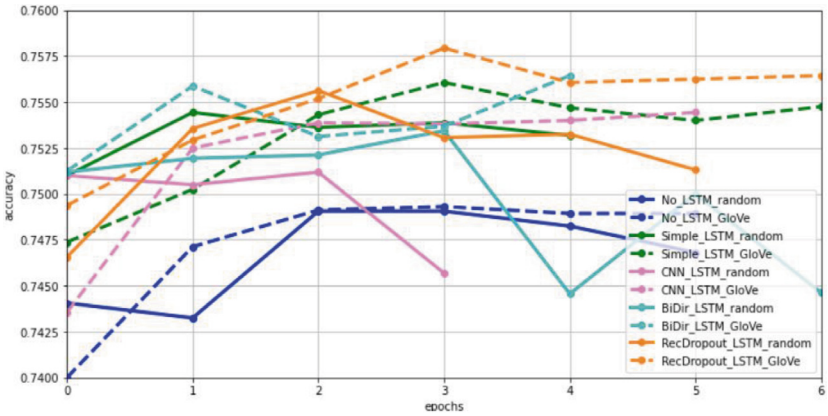

Fig. 3. Model accuracy on the public data set

4.2 Phase B & C results

Once the models were trained on the public dataset, they were fine-tuned with their own private data, which comprised around 8% of the original dataset.

The summary results in Fig. 4 reveal that the training rounds on unseen private data gave a boost of around 1% for each model. This was not a lot, yet it could be anticipated. Due to the low number of tweets per user, the private datasets contain multiple users' tweets. Effectively, each private dataset is comparable to a sample from the population and is therefore most likely similarly distributed to the remaining private datasets. If we otherwise had used private data that is unbalanced and carries a lot of specific user traits, then this phase would have been more impactful and provided better results.

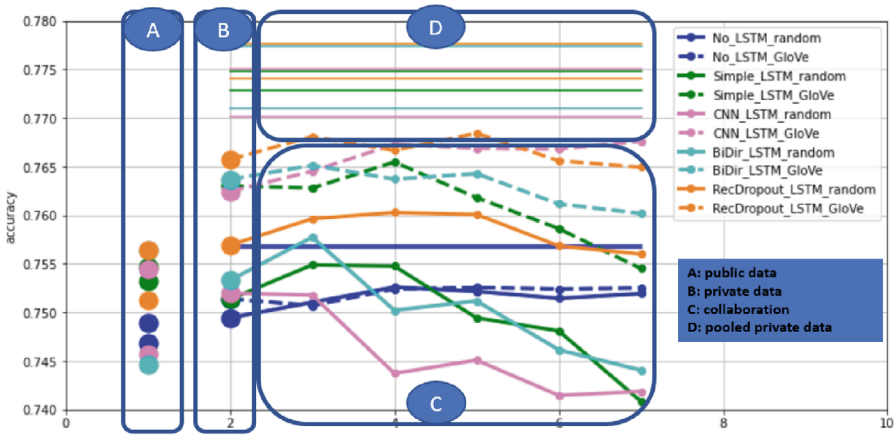

Fig. 4. Final results summary

The next phase C tested the model's capacity for collaborative learning. Each model shares their logits for an alignment dataset, randomly sampled from the public data. The validation accuracy reveals that the models did not benefit from this phase; even worse - their performance went back a bit, as demonstrated in Fig. 5.

	Train_acc	Train_loss	Validation_acc	Validation_loss
No_LSTM_random	0.766	0.497	0.752	0.519
No_LSTM_GloVe	0.764	0.504	0.753	0.519
Simple_LSTM_random	0.839	0.352	0.741	0.589
Simple_LSTM_GloVe	0.822	0.382	0.755	0.523
CNN_LSTM_random	0.854	0.332	0.742	0.605
CNN_LSTM_GloVe	0.794	0.437	0.768	0.487
BiDir_LSTM_random	0.841	0.350	0.744	0.602
BiDir_LSTM_GloVe	0.806	0.417	0.760	0.512
RecDropout_LSTM_random	0.813	0.398	0.756	0.519
RecDropout_LSTM_GloVe	0.804	0.417	0.765	0.494

Fig. 5. Results after collaborative learning

The likely reason behind the witnessed development is over-fitting. Training the models on the alignment datasets with a target determined by averaging all models' predictions was unlikely to introduce new knowledge, given that the models were already performing within 2% difference. Unsurprisingly, the models that did best - LSTM with recurrent dropout: 76.5% and a CNN with a LSTM layer: 76.8% - had extra protection from over-fitting in the form of dropout layers.

In the original FedMD setup, [7] boosted the accuracy by about 20% in the collaboration round. A main distinction is that their public and private data came from different sources: MNIST and EMNIST. This allowed for data heterogeneity and provided opportunities to transfer knowledge from one dataset to the other. In addition, some of the experiments included training the models on only selected data classes, which made the collaborative learning phase more vital for obtaining knowledge on unseen classes. These extreme circumstances can occur in the real world, yet they could not be simulated with the Sentiment140 dataset. This conclusion stresses the need for real-world FL datasets.

4.3 Results comparison to non-FL benchmark

The results from the final supplementary D phase brings some good news. Here, the phase A models were trained on all pooled private data. The setup mimics the traditional case where all models have access to all data and are trained on it. As can be seen in Fig.6, the achieved accuracy is extremely close to the FedMD results, which strengthens the case for FedMD.

4.4 Results comparison to LEAF benchmark

As noted earlier, the selected dataset is one of LEAF's FL benchmark datasets. In their setup, [2] used the standard FL architecture with a uniform model across all clients.

Each user was translated into a client and depending on the setting, a minimum of 3, 10, 30 or 100 tweets were required to participate. This setup is very different from the FedMD approach. Regardless of the scenario, the LEAF models' median accuracy did not exceed 68%, as shown in Fig. 6. This is well below the worst performing FedMD model (simple LSTM neural network with 74.1% accuracy). It is, however, difficult to judge which learning setting performed better due to differences in the amount of training data. Nevertheless, the LEAF results indicate that the FedMD performance is on par with and not inferior to the benchmark FL setting. Even better, FedMD facilitates model personalization, as well as collaborative learning, which should prevent big fluctuations in performance, as witnessed in the LEAF models results. Furthermore, the FedMD model architecture was selected with a diversity intention in mind and not only performance, leaving space for further improvements in model design and variations.

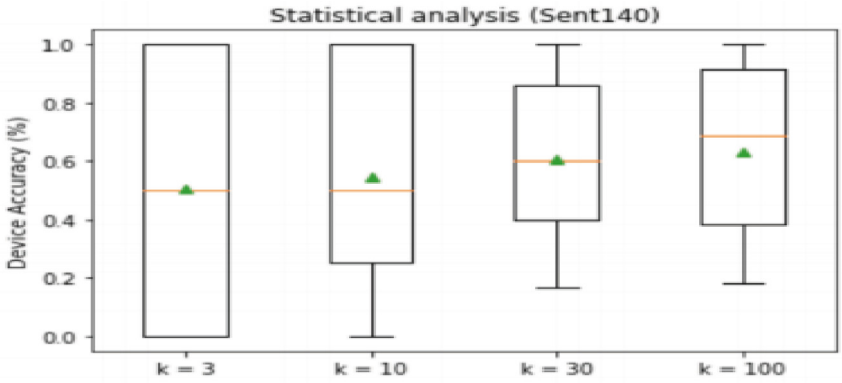

Fig. 5. LEAF's SENTIMENT140 results by [2]

4.5 Results and Discussion

Overall, the results confirm the feasibility of FedMD applied on NLP problems such as sentiment classification. Decentralized learning over decentralized data was successfully carried out in a privacy-compliant way. The individual learning phases resulted in a well-rounded learning setup, in which knowledge was transmitted through transfer learning and knowledge distillation. During the collaborative learning phase, the unique models could benefit from each other's knowledge without sacrificing its ability to deliver personalized predictions. This outcome can prove useful in counteracting the non-IID data scenario, which is highly probable in real-world circumstances. It should be further noted that the final models performed consistently well, which is an indication for the robustness of the setup. Despite limitations imposed by the Sentiment140 dataset, the final results were comparable to the LEAF benchmark and did not lag much behind the alternative non-FL implementations. The small loss in accuracy vs. the theoretical limit was compensated by the fact that 80% of the data remained local.

This experiment should be repeated using a dataset that satisfies the data availability requirements for each client, in order to better showcase the worth of FedMD and avoid over-fitting. For this reason, FedMD appears to be a better fit to companies or institutions as clients, rather than small private users.

5 Conclusion

This paper demonstrates that FL opens a world of opportunities for privacy-preserving machine learning. It is a development that tries to facilitate the ongoing AI revolution, while satisfying the data privacy demands with in-built safeguards. The main contribution of this research paper is the successful adaptation and implementation of FedMD framework to an NLP problem. Another achievement is the small gap between the FedMD results and the alternative non-FL implementation, even though the FedMD models trained on less data and kept their locally available data private.

In order to solidify these outcomes, experiments with multi-class classification challenges are recommended. Even better, a real-world FL dataset could be an ideal testing ground for FedMD without artificial partitioning. Another worthwhile research direction is combining model- and system- heterogeneity - i.e. scenarios where clients experience limitations in bandwidth or drop out during the collaborative learning phase. Further research avenues include the expansion of FedMD to challenges other than classification.

Overall, experiments like the one presented in this paper prove the case behind FL and demonstrate the potential of privacy-preserving and performant frameworks such as FedMD.

References

1. Bonawitz, K., Ivanov, V., Kreuter, B., Marcedone, A., Brendan McMahan, H., Patel, S., Ramage, D., Segal, A., Seth, K.: Practical Secure Aggregation for Federated Learning on User-Held Data. arXiv e-prints arXiv:1611.04482 (2016)
2. Caldas, S., Meher Karthik Duddu, S., Wu, P., Li, T., Kone˘cn´y, J., McMahan, H.B., Smith, V., Talwalkar, A.: LEAF: A Benchmark for Federated Settings. arXiv eprints arXiv:1812.01097 (2018)
3. Chen, N., Wang, P.: Advanced combined lstm-cnn model for twitter sentiment analysis pp. 684–687 (2018). https://doi.org/10.1109/CCIS.2018.8691381
4. Goldberg, Y.: A Primer on Neural Network Models for Natural Language Processing. arXiv e-prints arXiv:1510.00726 (2015)
5. Jiang, Y., Kone˘cn´y, J., Rush, K., Kannan, S.: Improving Federated Learning Personalization via Model Agnostic Meta Learning. arXiv e-prints arXiv:1909.12488 (2019)
6. Kairouz, P., McMahan, H.B., Avent, B., Bellet, A., Bennis, M., Nitin Bhagoji, A., Bonawitz, K., Charles, Z., Cormode, G., Cummings, R., D'Oliveira, R.G.L., El Rouayheb, S., Evans, D., Gardner, J., Garrett, Z., Gasc´on, A., Ghazi, B., Gibbons, P.B., Gruteser, M., Harchaoui, Z., He, C., He, L., Huo, Z., Hutchinson, B., Hsu, J., Jaggi, M., Javidi, T., Joshi, G., Khodak, M., Kone˘cn´y, J., Korolova, A., Koushanfar, F., Koyejo, S., Lepoint, T., Liu, Y., Mittal, P., Mohri, M., Nock, R., Ozg¨ur, A., Pagh, R., Raykova, M., Qi, H., Ramage, D., Raskar, R., Song, D., Song, ¨ W., Stich, S.U., Sun, Z., Theertha Suresh, A., Tram`er, F., Vepakomma, P., Wang, J., Xiong, L., Xu, Z., Yang, Q., Yu, F.X., Yu, H., Zhao, S.: Advances and Open Problems in Federated Learning. arXiv e-prints arXiv:1912.04977 (2019)
7. Li, D., Wang, J.: FedMD: Heterogenous Federated Learning via Model Distillation.arXiv e-prints arXiv:1910.03581 (2019)
8. McMahan, H., Moore, E., Ramage, D., Hampson, S., Ag¨uera y Arcas, B.: Communication-Efficient Learning of Deep Networks from Decentralized Data.arXiv e-prints arXiv:1602.05629 (2016)
9. Pennington, J., Socher, R., Manning, C.D.: Glove: Global vectors for word representation. In: EMNLP. vol. 14, pp. 1532–1543 (2014)
10. Rahman Minar, M., Naher, J.: Recent Advances in Deep Learning: An Overview. arXiv e-prints arXiv:1807.08169 (2018)
11. Smith, V., Chiang, C.K., Sanjabi, M., Talwalkar, A.: Federated Multi-Task Learning. arXiv e-prints arXiv:1705.10467 (2017)
12. Tang, H., Lian, X., Yan, M., Zhang, C., Liu, J.: D2: Decentralized Training over Decentralized Data. arXiv e-prints arXiv:1803.07068 (2018)
13. Wang, X., Liu, Y., Sun, C., Wang, B., Wang, X.: Predicting polarities of tweets by composing word embeddings with long short-term memory. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics. pp.1343–1353. Association for Computational Linguistics, Beijing, China (2015)
14. Yang, Q., Liu, Y., Chen, T., Tong, Y.: Federated Machine Learning: Concept and Applications. arXiv e-prints arXiv:1902.04885 (2019)