Bachelor thesis
Computer Science

Radboud University

Robustness of Federated Averaging for Non-IID Data

Author:
Stan van Lier
s4256166

First supervisor/assessor:
prof. dr. ir. A.P. de Vries
`a.devries@cs.ru.nl`

Second assessor:
dr. ir. E. Herder
`eelcoherder@cs.ru.nl`

August 21, 2018

Abstract

Machine learning requires access to all the data used for training. Recently, Google Research proposed Federated Learning as an alternative, where the training data is distributed over a federation of clients that each only access their own training data; the partially trained model is updated in a distributed fashion to maintain a situation where the data from all participating clients remains unknown.

In this research we construct different distributions of the DMOZ dataset over the clients in the network and compare the resulting performance of Federated Averaging when learning a classifier. We find that the difference in spread of topics for each client has a strong correlation with the performance of the Federated Averaging algorithm.

Contents

1 Introduction 2

2 Preliminaries 4

2.1 Federated Averaging 4

2.2 Non IID Data 5

2.3 DMOZ Dataset 5

2.4 Spanning Tree 6

2.5 Word Embedding 7

2.6 Multinomial Logistic Regression 8

2.7 Adam optimizer 8

3 Research 9

3.1 Method 9

3.1.1 Text classification problem 9

3.1.2 Preprocessing 9

3.1.3 Distribution 10

3.1.4 Train/test split 12

3.1.5 Distributing samples 12

3.1.6 FedAvg parameters 12

3.1.7 Model 12

3.2 Results 13

4 Conclusions 18

4.1 Future work 19

4.1.1 Grid search 19

4.1.2 Comparing distributions 19

4.1.3 Document embedding 19

Chapter 1

Introduction

Federated Learning is a new technique in machine learning where the training data stays on the device of the owner [7, 5, 4]. Where traditional machine learning first aggregates data to train a model on a central server, federated learning sends the model to the data and aggregates the local updates of the model. This technique makes it more practical to train a model using privacy sensitive data or data which is too big for being aggregated. This however results in implications for on what data a model is being trained. Since data will not be aggregated and the training only happens on the local data created mostly by individuals, this data will inevitably hold certain patterns resulting from their behaviour. The result is that data is not distributed independently and identically, we refer to this situation as non-IID data.

This results in non independent and identically distributed data, or non-IID data. This could be problematic when applying traditional machine learning algorithms in the federated setting since those assume independent and identically distributed data, or IID data.

Research suggests that using a naive way of repeatedly averaging locally updated models works relatively well [7]. They compare the proposed algorithm Federated Averaging (FedAvg) for the IID setting with an non-IID setting. The paper constructs one distribution of non-IID data and find equal performing models as with IID data after enough rounds. The problem with this outcome is it may and probably will not be possible to make such an assumption, since it is not possible to know all behavioural factors of individuals which influence the data. This often is the reason to use machine learning in the first place.

There is currently not much research of how the implicit assumptions of federated learning affect the performance of the learned model. In research where federated learning techniques are proposed there is often just one assumptions of how data is distributed over clients. In reality the exact distribution is unknown while it is possible to make assumptions of how data could be distributed. In this research we give more insight in how strong these assumptions must be to ensure well performing Federated Learning, and in particular Federated Averaging.

Chapter 2

Preliminaries

2.1 Federated Averaging

The FedAvg algorithm proposed in [7] is a naive, however claimed to be well performing, federated learning algorithm. It describes the communication between a central server and K clients and how the model updates done by the clients are aggregated on the central server. The algorithm consists of multiple rounds where for each round t the server randomly selects a fraction C of the K clients, resulting in a subset S_t of $m = \lceil C * K \rceil$ clients. The server sends the current model w_t to all clients in S_t, which then train the model using their local data and send the updated model w^k_{t+1} back to the server. The server then calculates a weighted average of all models according to the number of samples on the clients n_k.

The paper describes the aggregated update as $w_{t+1} \leftarrow \sum_k^K \frac{n_k}{n} w^k_{t+1}$ which is the weighted average over all local updated models w^k_{t+1} for all K clients, however, the clients who are not in S_t have not been updating a model, hence w^k_{t+1} is undefined for all $k \notin S_t$. There are three ways this ambiguity can be interpreted.

1. The weighted average must only be taken over the selected clients S_t, resulting in $w_{t+1} \leftarrow \sum_k^{S_t} \frac{n_k}{n_{S_t}} w^k_{t+1}$ where n_{S_t} is the total number of samples on all clients in S_t.

2. For clients k which are not in S_t: $w^k_{t+1} \leftarrow w^k_t$ implicitly, and all clients will be send the same initial model w^k_0.

3. Clients which are not in S_t will be sent the new model w_t but they will not do local updates (or update zero times) and therefore $w^k_{t+1} \leftarrow w_t$ implicitly.

After experimenting with all three interpretations we find the first one to give the best results, we will use this method in the rest of this research.

The second interpretation can cause clients which are not selected for some successive rounds to have negative effect on the model update. The third interpretation converges really slowly, especially with a small fraction C of clients, then most of the w_{t+1}^k will be equal to w^t causing the average weight to be closer to the old model. Having the difference between the old model w_t and the updated model w_{t+1} be heavily depended of C would be weird since we already have a learning rate parameter η to control this.

2.2 Non IID Data

When each random variable in a collection of random variables has the same probability distribution, then the collection is independent and identically distributed, or IID. In all other cases the data is non independent and identically distributed, or non-IID. In the context of federated learning, data is IID when each sample is equally likely to occur on every client. In reality this can never be the case since data is produced by the client and therefore the client will influence the probability of containing a certain sample. This is inevitable with federated learning. Traditional machine learning techniques assume the IID of data. Research suggests a paradigm shift in the machine learning, from assuming IID data to assuming non-IID data [2].

2.3 DMOZ Dataset

DMOZ or the Open Directory Project is an open voluntary project of Mozilla which attempts to categorise all the best web pages of the internet. The data of the project was periodically made available through downloadable RDF-dumps. DMOZ closed on March 17, 2017. It was also know as the Open Directory Project (ODP). A successor version of the project is available at `curlie.org`. At the time of this research Curlie.org did not yet host their data in RDF dumps so we used the last dump form DMOZ, found on `curlz.org/dmoz_rdf/`.

We only use two files from the last available dump (from 12-03-2017), `content.rdf.u8` and `structure.rdf.u8`. The `content.rdf.u8` file contains information about 4 million pages, the topic, title, description and URL, some examples are shown in table 2.1. The `structure.rdf.u8` file contains information about topics, in particular the relations with other topics. There are three kinds of relations. The narrow relations are mostly between different hierarchies, where symbolic and related relations are mostly further away from each other. Further detail on what these different kind of relations implies is not relevant for this research. Some of these relations accruing in the dataset are shown in figure 2.1. Many topics occur only as relations or only as pages.

topic	Top/Shopping/Consumer_Electronics/H
title	Hello World Communications
description	New York based store offering audio, video, and communications rentals. Phone call required for ordering.
about	http://www.hwc.tv/
topic	Top/Society/Religion_and_Spirituality/Christianity/Denominations/Catholicism/Reference/Catholic_Encyclopedia/T
title	Tabbora
description	A titular see in Africa Proconsularis, suffragan of Carthage.
about	http://www.newadvent.org/cathen/14423d.htm
topic	Top/Recreation/Humor/Wordplay/Puns/Daffynitions
title	Improve Your Computer Vocabulary
description	Daffynitions from the Goleta Publisher.
about	http://www.troutcom.com/gdtpug/9605.html#vocabulary
topic	Top/Science/Biology/Flora_and_Fauna/Animalia/Arthropoda/Insecta/Coleoptera/Curculionidae
title	European Elm Flea Weevil
description	Photographs of Orchestes alni, description, life history and habits.
about	http://wiki.bugwood.org/HPIPM:European_Elm_Flea_Weevil

Table 2.1: Examples of pages in the dataset

Topics can be seen as hierarchical categories, some are more specific on a subject then others. There are 15 top level topics, all other topics are subtopics of one of those. The number of pages underneath subtopics, or in, the top-levels topics is shown in Figure 2.2.

2.4 Spanning Tree

In Section 3.1.3 use a spanning tree of topic relations to select a number of topics. A spanning tree is a sub graph for which hold that all nodes are at least indirectly connected. For example if the direction of the edges is ignored then Figure 2.1 is a spanning tree. This can be verified by checking if all nodes can be reached by only following the edges.

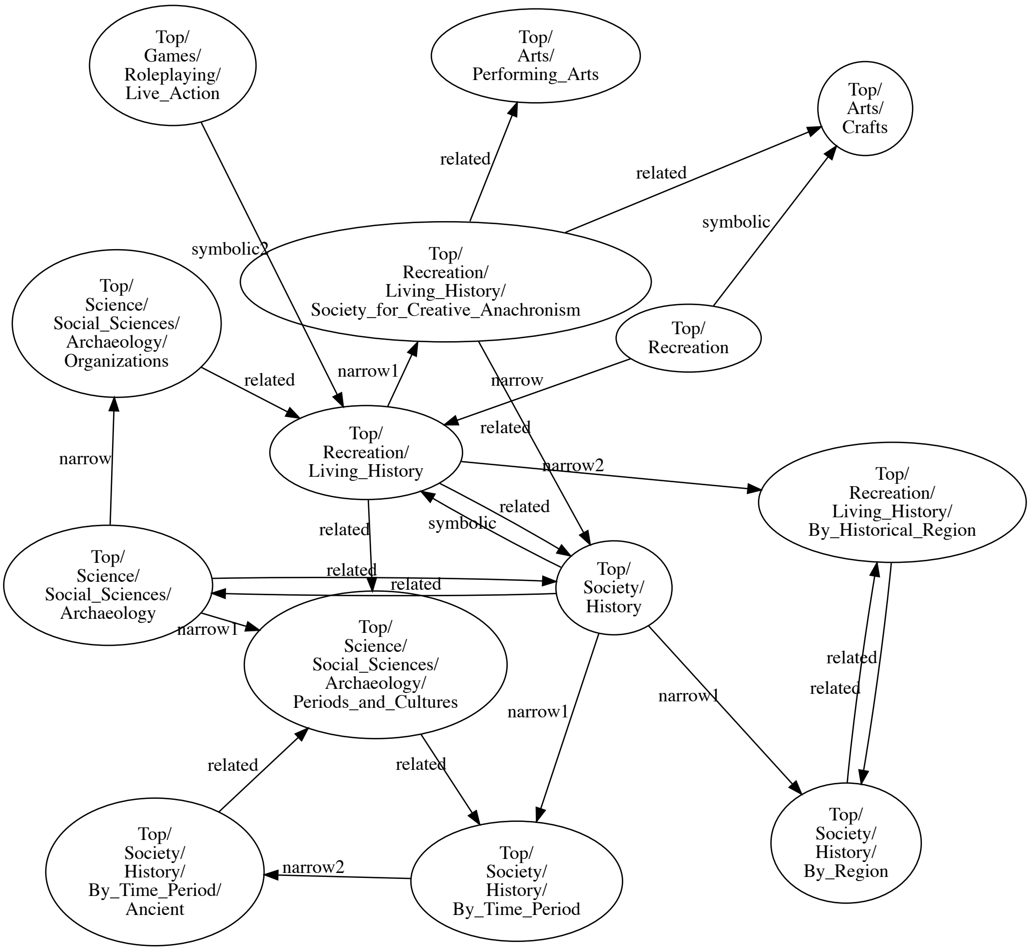

Figure 2.1: Example of relations between topics in the DMOZ dataset

2.5 Word Embedding

We use a word embedding for the transformation of words into a point in vector space to feed into the model. A sequence of words will then be represented as a sequence of vectors. We use the GloVe.6B pre-trained word embedding [8], which maps 400.000 words to 50 dimensional vectors. A property of this embedding is that the meaning of words is represented by the point where the word is mapped to, such that the euclidean distance between words having similar meaning is smaller than between words with different meanings.

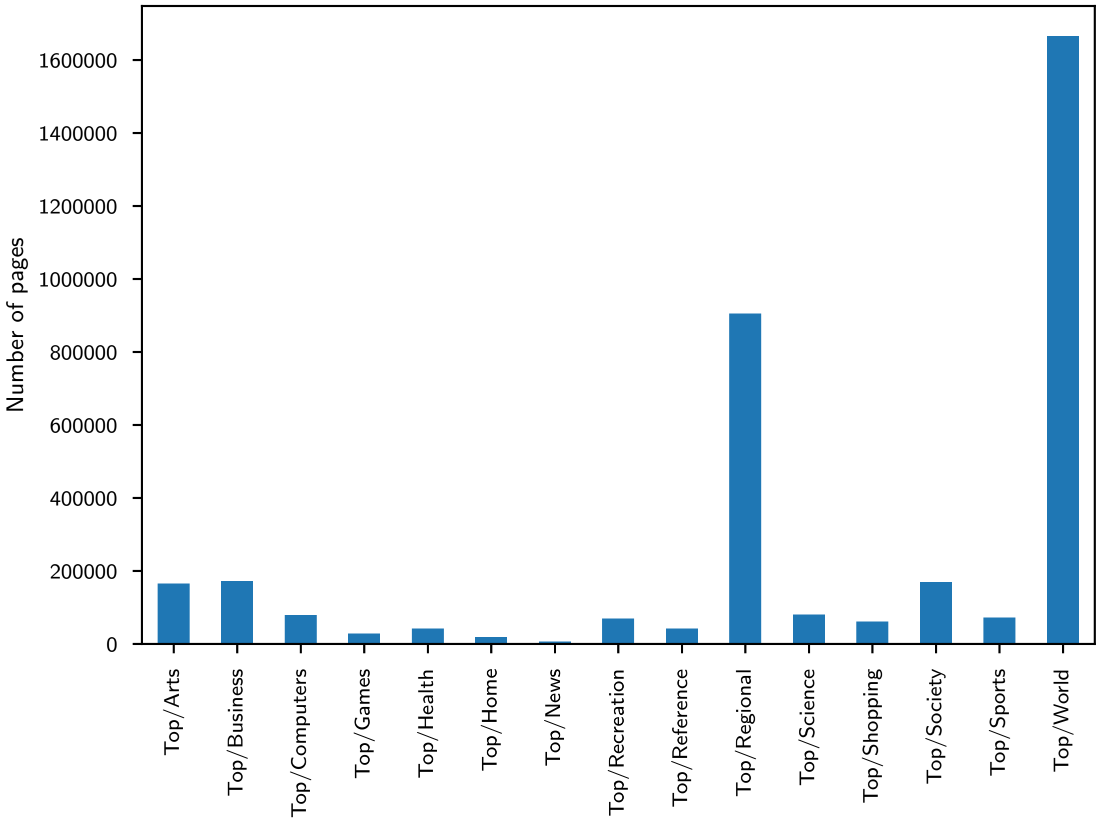

Figure 2.2: Number of pages for each top-level topic.

2.6 Multinomial Logistic Regression

The original logistic regression model can solve binary classification problems. This can be generalised to multinomial logistic regression which can predict probabilities of more than two outcomes [1]. The model is made up of connections from all input dimensions to all output dimensions which correspond the probability of every outcome. These connections have weights and biases which are learned using gradient descent.

2.7 Adam optimizer

Adam is a Stochastic Gradient Descent extension which uses a separate adaptive learning rate for every parameter in the model [3]. This causes the model to converge much quicker and reduces the running time of the experiments.

Chapter 3

Research

3.1 Method

3.1.1 Text classification problem

To test the performance of federated learning we first define a machine learning problem. This will be the classification of top-level topic of pages in the DMOZ dataset. There are 15 of such topics, as can be seen in Figure 2.2 the samples are highly unbalanced. 75% of the pages are part of "Top/Regional" and "Top/World" and almost non are in "Top/News". If we would use "Regional" and "World" as labels in our classification problem the model could learn to predict only those classes and still gain an accuracy score of 0.75. On the contrary, the model will gain almost nothing for predicting the label "News" right so it will probably not learn to do so. To simplify our experiment by not needing to take these unbalances into account we drop all pages in or underneath these three topics. We use the remaining 1 million pages as samples and we label each using the remaining 12 top-level topics. Figure 3.1 shows the number of samples for each label. Note that because we used part of the topic information for the class labelling we must not use this in the input to the model. The sample input data will consist of the page title concatenated with the page description with a special token in between.

3.1.2 Preprocessing

To speed up the training process we use the pre-trained Glove.6B embedding [8]. The vocabulary consists of 400.000 words, which are embedded in 50 dimensions. All words are in lower-case and most do not contain special characters, although special characters are in the vocabulary as single character. Therefore we transform every word to lower-case and detach any special characters from words. Then we map each word and special character to their embedding. We add an embedding for unknown words to handle

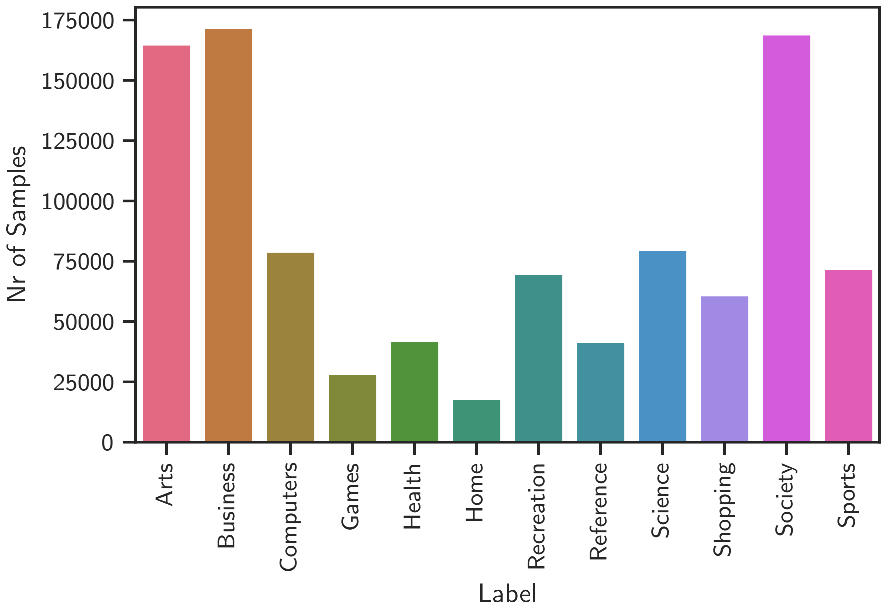

Figure 3.1: Number of samples for each label.

words which are not in the vocabulary and we add an embedding for the special token which marks the concatenation of the title and the description. This results in a sequence of vectors for each sample. These sequences have varying size since the page title and description have varying size. This is problematic since we need one input dimension for our model. To equal all sample dimensions we take the maximum sequence length and pad all samples which are shorter with zero vectors until the length is equal to the maximum sequence length.

3.1.3 Distribution

To test the robustness of federated averaging we try to keep as much information as possible in the way samples are distributed over clients, while at the same time making it slightly easier for the model to train with each next distribution d_{i+1}. We also distribute samples randomly over clients in d_{iid} which we hypothesise produce the best training results.

Technique

Using the DMOZ dataset we distribute the samples over clients using the topic information. Since there are many topics which all are hierarchical we can choose the number of clients relatively freely. To distribute samples over K clients we assign one unique topic to each client. We place all samples on the client for which the assigned topic is equal to the sample topic. If there does not exists such a client than we reduce the sample topic by going up one hierarchical level until there is a client assigned to the topic, and we place the sample with that client. Using this technique we distributed all samples over the K clients and thus ensuring a degree of preservation in topic information with the distribution. This distribution d_0 has the minimum entropy used in our experiment, we captured as much information as possible in the way we distributed the samples over K clients using the chosen topics.

To make other distributions having increasing entropy we look at the relations according to client topics. We consider all differed kind of relations and we ignore the direction of relations. We only use relations for which both topics are assigned to clients. Then we place each sample with equal probability on one of the clients which have a direct relation with the client where the sample currently is placed on, and to lower the entropy increment per distribution iteration we place the sample on the current client with twice the probability. We can iterate this process multiple times while every iteration increases the entropy of the resulting distribution. The goal is that after enough iterations the entropy of the resulting distribution is close to the entropy of a IID distribution.

Client topics

As noted in section 2.3 not all topics of samples have relations. If we would assign such topics to clients and using the described distribution technique, than the same samples will be placed on such clients with every distribution iteration d_i. This will prevent the entropy to increase since the distribution is not changing for all samples placed on such clients. Another cause of this problem is when some client topics are having relations only with each other, resulting in samples which placement is only possible on one of those clients. We prevent these issues by building multiple spanning trees using all relations. We then only use the topics occurring in the largest spanning tree.

Another criterion for clients is that it needs to contain enough samples in order to have a meaningful contribution in federated learning. We set this threshold on a minimum of 200 samples per client. This is realised by first looking at each of the lowest hierarchical level of used topics and counting how many samples would be placed on a client having the topic. If this is above 200, we pick this topic to be used as a client topic. Then we go up one hierarchical level, we again count the samples which would be placed on each of the topics in this level by taking the earlier picked topics into account, and again only pick topics for which a client would have more than 200 samples. We repeat this until all levels are checked, then we check if all picked topics still produce a spanning tree, if this is not the case we again pick only the topics in the biggest spanning tree and repeat the process.

This is repeated until we end with just one spanning tree. In our case we end up with 1984 topics, hence the number of clients $K = 1984$.

Using this technique we ensured the possibility for every sample to be placed on a every client after at least K distribution iterations. This increases the possible entropy using the distribution technique.

3.1.4 Train/test split

To test the performance of the model we sample a test set from the data. The test set will be 30% of the samples selected using a stratified selection on the picked topics. This preserves equal percentages in amounts of samples with each client topic. The other 70% will be our training set. We do not use a cross validation technique since it is not the purpose of this research to find a realistic prediction score but only to compare scores obtained from different distributions of the data.

3.1.5 Distributing samples

Using the distribution technique described in Section 3.1.3 we construct the initial distribution d_0 using the training set and the 1984 clients. We do 100 distribution iterations to construct all distributions from d_1 until d_{100}. We also make one random distribution d_{iid} by assigning a random client to each sample. Figure 3.2 shows for some of these distributions how the labels of samples are distributed over all clients. Samples occurring on clients which have the same top-level client topic have the same colour. Notice how only the distribution of labels across clients becomes more random, while the total number of samples per client stays unbalanced. Figure 3.3 shows the entropy increment for all distributions.

3.1.6 FedAvg parameters

In the paper which proposes FedAvg [7] many combinations of parameters have been tested. After some experimentation with the parameters which gave performance in that paper we choose just one set of parameters for our experiment. These are: mini batch size $B = 64$, fraction of clients $C = 0.1$ and client epochs $E = 5$. The only variable in our experiment is the number of distribution iterations d_i.

3.1.7 Model

The label predictions are done by a multinomial logistic regression classifier [1]. The model consists of sequence length × embedding dimensions × classes parameters. We use as sequence length the maximum in our data and pad all others with zero vectors to have equal size. In our case this is 165 which results in a total of $165 \times 50 \times 12 = 99000$ parameters. We use the

Adam optimizer with a learning rate of 0.001. This speeds up the training process a lot compared to Stochastic Gradient Decent by using an adaptive learning rate for every parameter [3].

3.2 Results

We run the setting for eight different distributions, d_0, d_5, d_{10}, d_{25}, d_{50}, d_{75}, d_{100}, d_{iid} and for the central setting (none distribution). For every run we use the same random seed to make sure that the performance cannot be influenced by the randomly picked clients nor by the initial model weights. As shown in Figure 3.4 there is a strong correlation between the score of the model and the used distribution. The IID distribution d_{iid} has the highest score after enough rounds, although before round 40 d_{75} and d_{100} have a slightly better score. When compared to no distribution, federated learning scores better on the accuracy scale with distributions d_{50}, d_{75}, d_{100} and d_{iid}, on the ROC AUC scale standard machine learning has the optimal score.

As shown in Figure 3.5 from round 150 until round 250 there is little to no improvement in score for each of the distributions.

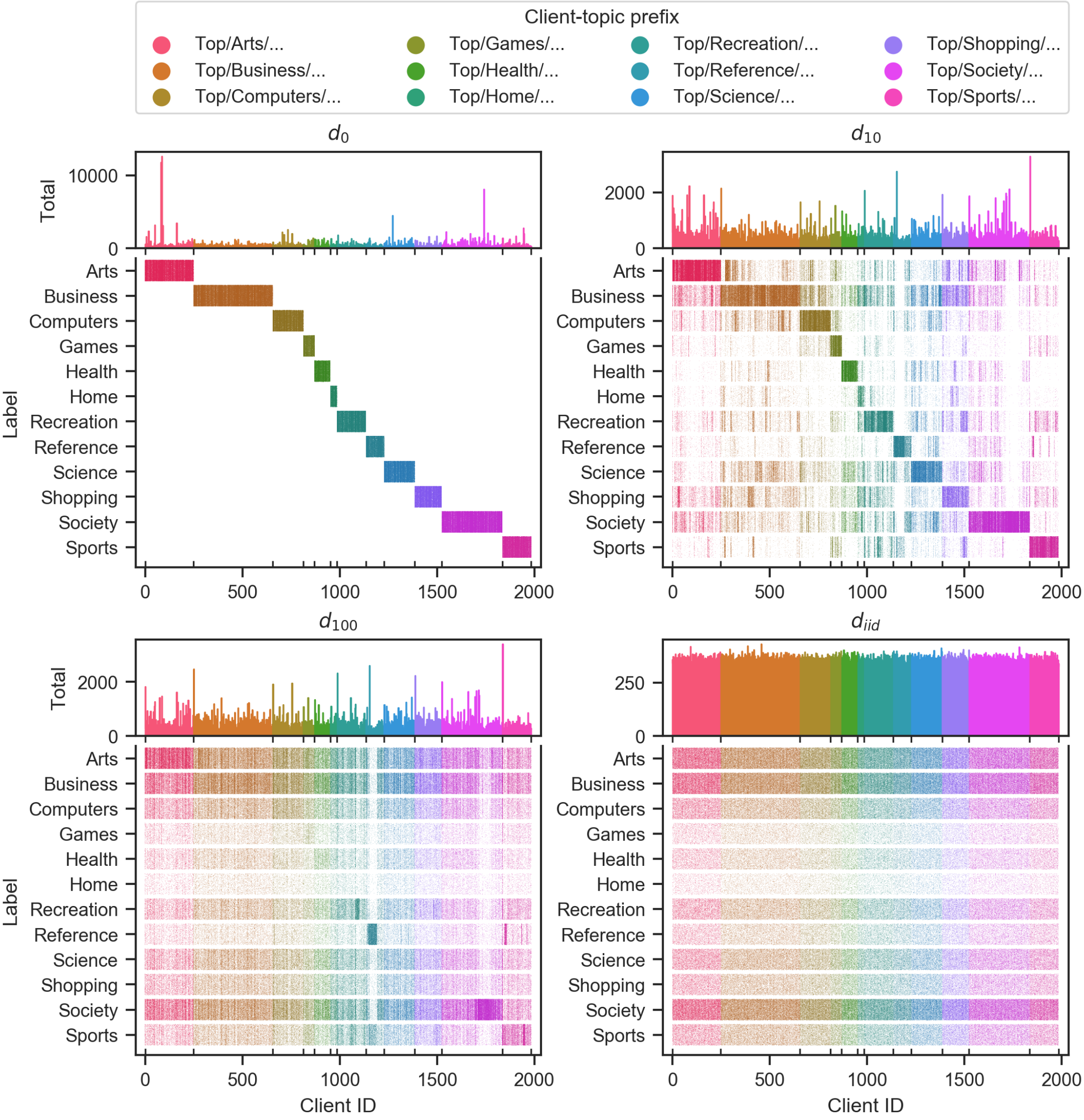

Figure 3.2: Four distributions of training data over the clients. Samples are coloured corresponding to the client they occur on. Clients which have the same top-level topic will have the same colour. The separation between colours is also shown with the x-ticks on both sides

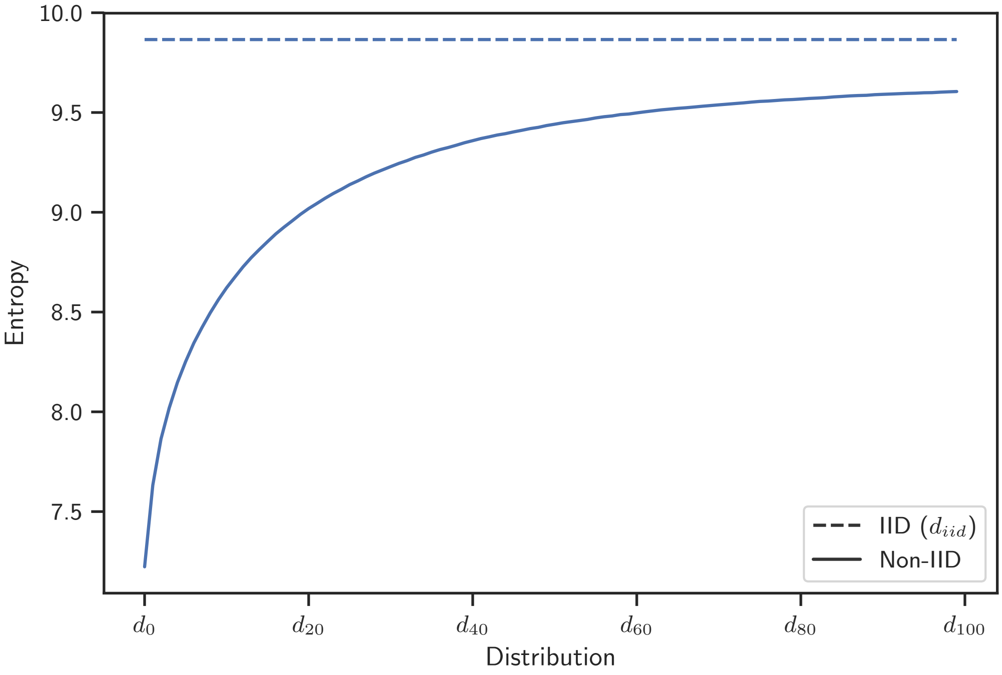

Figure 3.3: Entropy for labels occurring on clients for distribution iterations compared to the entropy of the IID distribution. The entropy of each distribution calculated by $-\sum_{k,y} \Pr(k, y) * \log\big(\Pr(k, y)\big)$ where k ranges over all clients, and y ranges over all labels, hence $\Pr(k, y)$ stands for the probability for client k to contain samples with label y.

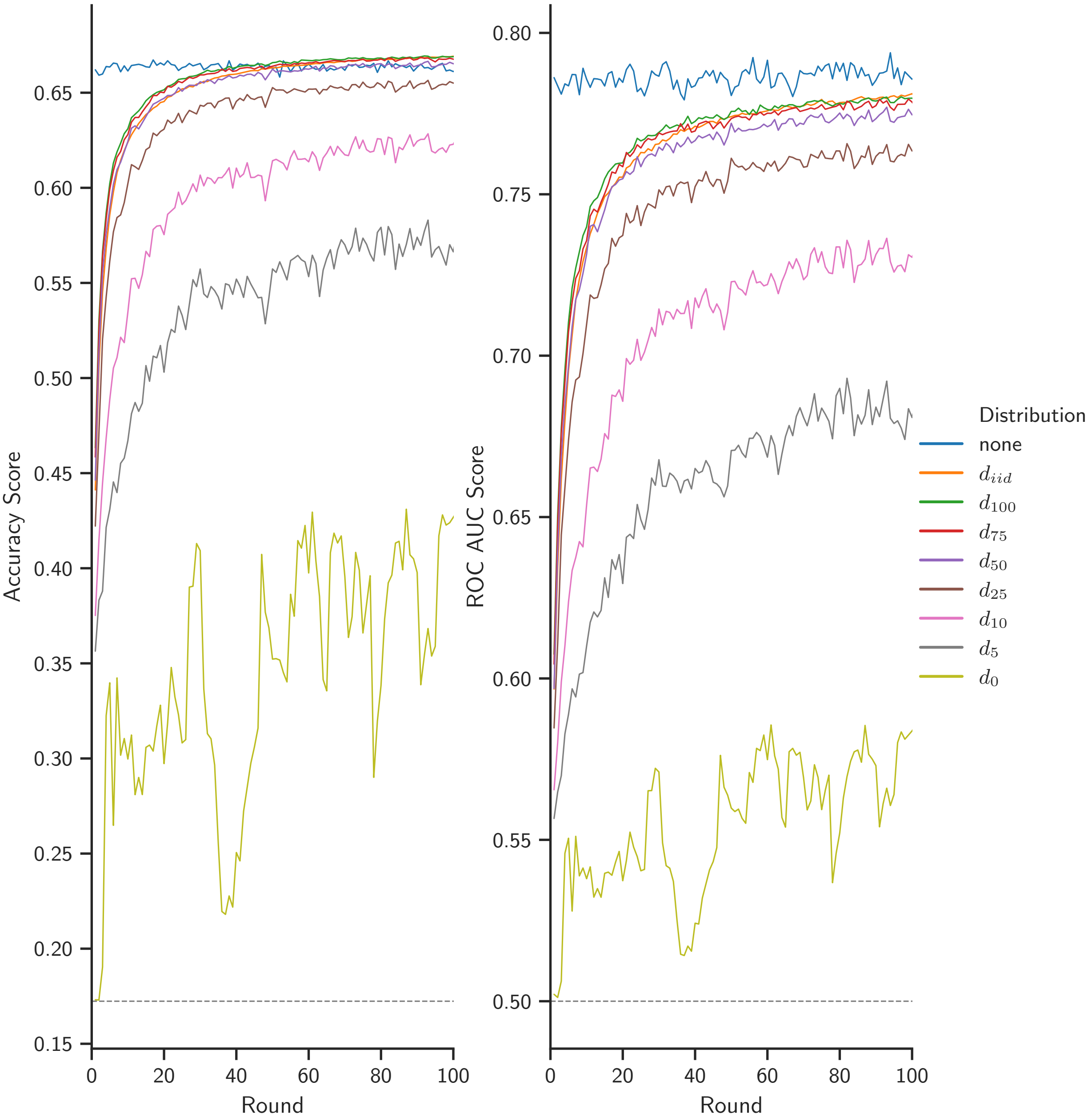

Figure 3.4: Accuracy and ROC AUC score on the test set for the first 100 rounds for eight of the distributions and for standard machine learning (Distribution = none). The dashed line indicates the score for the base classifier. All runs use the same random seed for the client selections S_t.

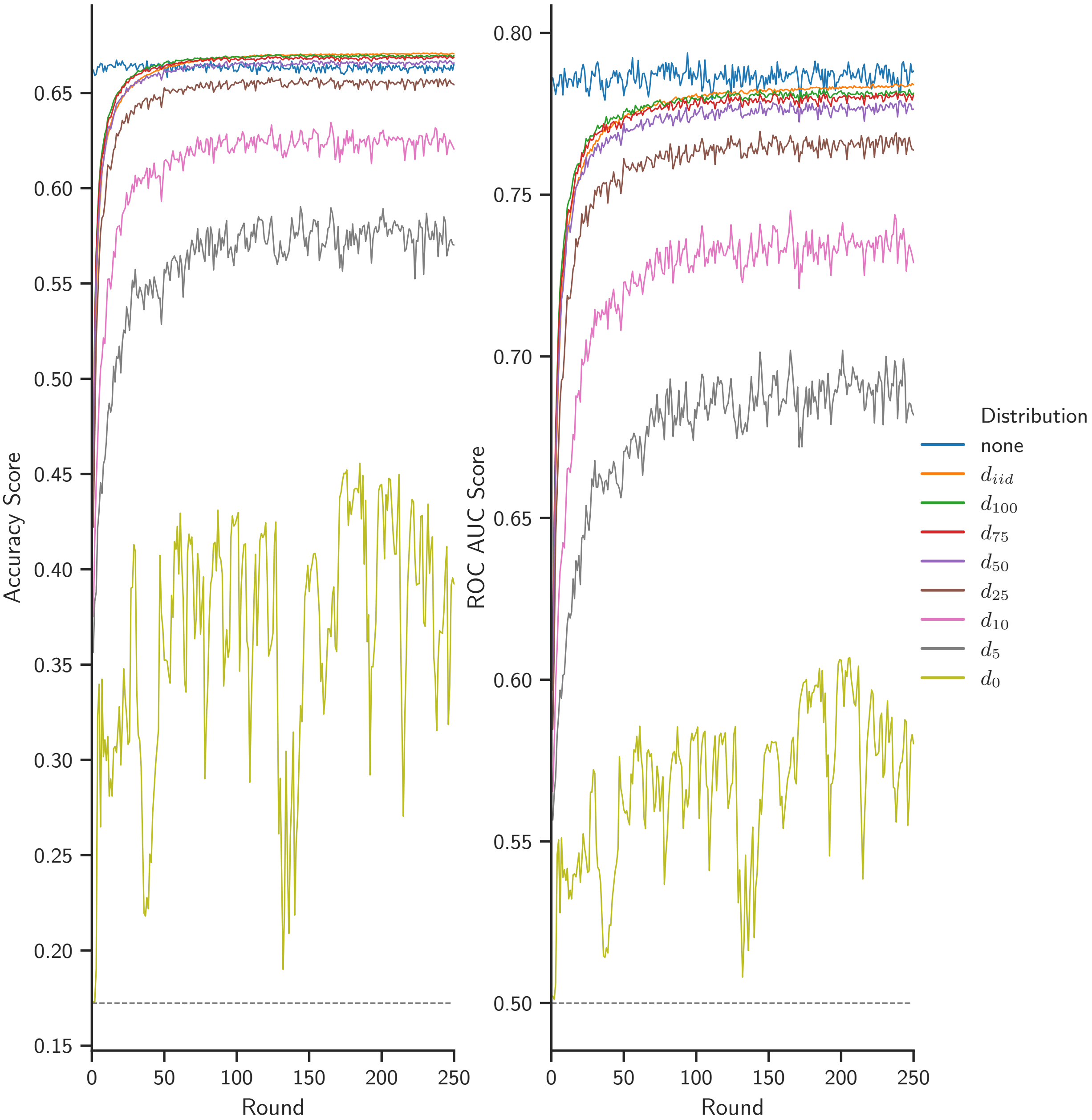

Figure 3.5: Same as Figure 3.4 but for the first 250 rounds.

Chapter 4

Conclusions

As shown in the results, there is a clear correlation between our constructed distributions and the score of the model trained in a federated setting by using these distributions, and by using the same FedAvg parameters. We also see that after enough distribution iterations (more then d_{75}) there is almost no difference in performance. Sometimes the distribution can have a slight advantage over an IID distribution as we see with d_{75} and d_{100} until round 40. This can be due to that selected clients have many more samples than the mean samples per client, causing the model to being trained on more data than the model with the IID distribution where all clients have a number of samples around the mean. This same result has been found by [7]

With distributions having high entropy FedAvg can score better on the accuracy scale than standard machine learning, while standard machine learning scores better on the ROC AUC score. This means that FadAvg more often correctly classifies samples belonging to the majority classes while standard machine learning is better to correctly classify samples from minority classes.

When trained for 250 rounds a clear maximum for each distribution is reached. It is improbable that the model will score better then this when trained for many more rounds.

We conclude that by applying FedAvg the distribution of samples over clients according to their classes have a direct impact on the performance of the learned model when used the same FedAvg algorithm parameters for each distribution. Although even with a hard separation of labels across clients (d_0) a better model can be learned than the base classifier which always predicts the class with the highest number of samples.

All code to reproduce the experiment is available at `https://github.com/stanvanlier/FedAvg_DMOZdistributions`.

4.1 Future work

4.1.1 Grid search

There is the possibility that by tuning the FedAvg parameters for each distribution independently will produce higher scores. This is due to the possibility for specific parameter combinations taking advantage of the characteristics of certain distributions. This can be researched by performing a grid search on some values for all FedAvg parameters.

4.1.2 Comparing distributions

This paper only shows the possibility of constructing assumptions, further research can be done in how exactly these assumptions can be constructed when a distribution producing a desired score has been found.

4.1.3 Document embedding

We use in this paper a sequence of word embeddings to represent pages. Since all sequences need to be equal sized this results in many sequences to have a long zero padding. To get a better representation of a page in the available dimensions a Doc2Vec [6] embedding could be used.

Bibliography

[1] Dankmar Böhning. Multinomial logistic regression algorithm. *Annals of the Institute of Statistical Mathematics*, 44(1):197–200, Mar 1992.

[2] Longbing Cao. Non-iid recommender systems: A review and framework of recommendation paradigm shifting. *Engineering*, 2(2):212 – 224, 2016.

[3] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

[4] Jakub Konečný, H. Brendan McMahan, Daniel Ramage, and Peter Richtarik. Federated optimization: Distributed machine learning for on-device intelligence, 2016.

[5] Jakub Konečný, H. Brendan McMahan, Felix X. Yu, Peter Richtarik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. In *NIPS Workshop on Private Multi-Party Machine Learning*, 2016.

[6] Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In *International Conference on Machine Learning*, pages 1188–1196, 2014.

[7] H. Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2017.

[8] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014.