A collaborative training approach for stress detection

Eleonora Ciceri[(1)], Marco Mosconi[(1)], Boris Rozenberg[(2)], Ron Shmelkin[(2)*]

(1) MediaClinics Italia
Via alla Cascata 56/C, Trento, Italy.
{e.ciceri, m.mosconi}@mediaclinics.it

(2) IBM Research - Haifa
Haifa Univ, Mount Carmel, Haifa, 3490002, Israel.
{borisr, ronsh}@il.ibm.com

Keywords: stress detection, neural network, collaborative training, data protection.

Abstract. Welfare solutions targeting workers are often focused on promoting healthy behaviors, with the goal of preventing pathological conditions caused by sedentary lifestyle. Specifically, *stress detection* is drawing more and more attention in these last years, as occupational stress is widespread among workers and can rapidly become chronic, bringing several complications with it. Nevertheless, existing tools for automatic stress recognition often have low accuracy (as training is generally done on small and unrealistic datasets collected in controlled environments) or low applicability (as they would require workers to wear many uncomfortable devices during their work). Building a tool that would work in real life would require to select very simple sensors (for instance, wearable ECG monitors mounted on T-shirts), collect data in real workplaces and use it to train a detection model. As data collection is expensive for companies, the optimal solution would be achieved by collecting data from several workplaces, so as to enlarge the dataset dimension and guarantee that such data comes from diversified sources. A central entity collecting all the data would then perform model training and send the trained model to each company. However, whenever datasets contain sensitive information (as in the case of stress detection), datasets sharing is not allowed, due to existing privacy regulations such as GDPR, ePrivacy, HIPAA etc. Thus, protection measures have to be put in place to enable data sharing in compliance with the aforementioned regulations. In this paper we present a collaborative training solution for stress detection that allows multiple parties to participate in the training of a large model with high performance. The proposed solution puts in place privacy-enhancing technologies to guarantee that data coming from a participant is not disclosed to other parties during the training process, guaranteeing workers' privacy.

1 Scientific Background

Stress is a reaction of the body that manifests itself when our brain recognizes some situation as either a challenge or a threat. A classical manifestation is *occupational stress*, that can be observed in workplaces as a reaction to different situations (for example, long working hours, lack of rewards). Workers of all kinds can be subjected to this condition, even at alarming levels when the pressure at work becomes too high. For instance, during the SARS-CoV-2 pandemic in 2020, several nurses and doctors experienced burnouts, as a result of emotional stress and increased number of responsibilities.

Occupational stress can become a chronic condition. For this reason, there has been a lot of attention on the topic over the last years, either to create tools that workers could use to reduce stress (for instance, collections of mindfulness exercises), or to recognize

[*]All authors have contributed equally

stress at early stages and avoid its chronicization. Tools for automatic stress recognition, which are often based on machine learning models, can monitor several types of physiological parameters to recognize stress statuses (for instance, electrocardiogram, electroencephalogram, galvanic skin response, eye movement) [1]. Nevertheless, the models found in literature are often trained on small datasets collected in controlled, unrealistic environments, and this results in a lack of accuracy when used in real-life situations and a lack of applicability, as some parameters used to train models are collected with uncomfortable devices that a worker would not wear daily. Therefore, companies that are interested in offering stress management solutions to their workers could decide to build better tools by collecting data in-house (using wearable sensors on their workers) and train their own models. This approach is unfortunately not valid for small enterprises, which have a limited number of workers and would collect insufficient data. A possible solution would then be to create a consortium of companies that are willing to build a *collaborative model*, and train the model using all their data. Nevertheless, although a collaborative solution would indeed guarantee a high performance due to the usage of a large and diversified dataset, personal data cannot be shared freely, as its processing is regulated by data protection laws (GDPR). Therefore, this solution is applicable only if direct data access is guaranteed exclusively to the data controller, and the model is trained on protected data.

In this paper, we present an utilization of privacy preserving collaborative training method to train a stress detection model using data coming from different sources. This solution makes use of privacy-enhancing technologies to protect data coming from each single source, so that the collective model is generated by merging several local models and avoiding direct data sharing among parties, thus ensuring workers' privacy.

2 Materials and Methods

To train a model that detects stress in workers, we decided to reproduce a method for stress detection that is manually performed by cardiologists, where heart rate variability features (henceforth, HRV) are observed to assess the stress level [1]. Such a method as a high applicability, as HRV features can be easily acquired using wearable devices, and the burden of tagging data can be distributed over several companies in a collaborative environment.

In the following sections, we present the dataset we used to perform training, the model to perform stress detection and the approach to collaborative training.

2.1 Dataset

For this study, we used a publicly available dataset called SWELL [2], that retraces the type of dataset we envisioned for this experiment. Indeed, the dataset (which in its original form contains several types of data) is available on Kaggle[6] in a reduced form comprising only HRV features. Each entry in the dataset is composed of 34 HRV features, and classified as either `interruption` (that is, a stress condition caused by the sudden interruption of a work activity), `time-pressure` (that is, a stress condition caused by the execution of a work activity in a very limited amount of time) or `no-stress`.

We reshaped the dataset by: i) removing three undocumented features (namely, `VLF_PCT`, `LF_PCT` and `HF_PCT`), so that the final feature vector contains 31 features; ii) reducing the problem to binary classification, by clustering the entries labeled as `interruption` and `time-pressure` under a unique class `stress`. The reduction to a binary classification problem was performed to accommodate a business requirement, that is, recognizing if a person is stressed or not, without analyzing the reasons why the person would be stressed. After the reduction, the dataset contains 222240 `no-stress` samples and 188082 `stress` samples, which has the positive side of

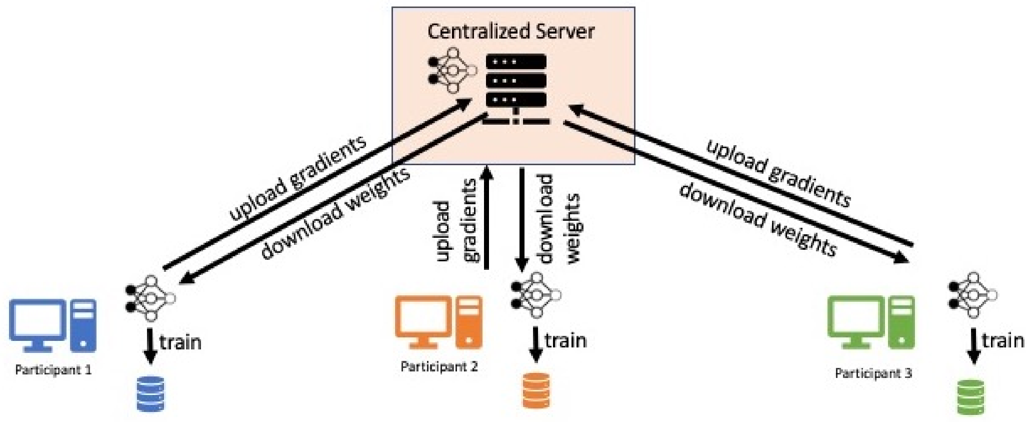

Figure 1: Collaborative training flow

being more balanced with respect to the original dataset.

2.2 Model definition

The work in [3] proposes a method to perform binary stress classification (stress/non-stress) by training a convolutional neural network (henceforth, CNN). In such a work, multiple HRV-based methods for performing stress classification were compared, showing that CNN outperformed the simpler methods (LDA, SVM). We thus take inspiration from this work and train a CNN on the dataset described in the previous section.

The model architecture is composed as follows:

- **Input layer**. The feature vector is composed of 31 HRV features (see Section 2.1).
- **Hidden layers**. The model includes two convolutional layers (first: 256 filters; second: 1024 filters; ReLU activation function) and one dense layer (256 nodes; ReLU activation function).
- **Output layer**. The output of the binary classification problem is derived from a dense layer with a single node and sigmoid activation function.

The model was trained by subdividing the dataset into training set (369289 samples, 200082 non-stress and 169207 stress) and test set (41033 samples, 22158 non-stress and 18875 stress). The choice of hyperparameters (i.e., number of units in the two convolutional layers and number of nodes in the dense layer) was performed by exploring the hyperparameters space. The aforementioned model parameterization is the one that guaranteed the highest accuracy (99.24% on training set and 99.41% on test set).

In the next section, we present the approach used to train this model in a collaborative fashion.

2.3 Privacy Preserving Collaborative Training approach

When data is distributed among multiple parties, a model with high performance is obtained when each party uploads the data to a central entity, as by doing so the collected dataset is wide and diversified. The central entity will perform Neural Network (henceforth, NN) training on the entire dataset, and when the training is finished it will send the trained model to each of the parties. However, whenever datasets contain sensitive information, the parties are not allowed to share the datasets due to data sensitivity and existing privacy regulations such as GDPR, ePrivacy, HIPAA etc.

In order to allow multiple participants to perform collaborative training while guaranteeing the protection of training data of all parties, we employ and implement the method presented in [4]. Such a method (see Algorithm 1) allows multiple participants to perform CNN training collaboratively, making each participant perform independent local CNN model $\mathcal{M}$ training on their own dataset $\mathcal{D}$. During the training, each participant performs an iteration (epoch) on a local dataset $\mathcal{D}$ and calculates model's updates from the current epoch. Afterwards, the portion θ_{up} of gradients are chosen (in *choose_gradient* function) and participant may choose to add a Differential Privacy (DP) noise to these gradients. The gradients are chosen based on one of the two options: i) the most alternated gradients; or ii) randomly sampled gradients which are above a certain threshold. Each participant uploads the chosen gradients to the centralized server. The server component holds a joint model $\mathcal{M}_g$. On an upload call, the server updates the joint model with participants uploads. The server allows the participants to download any part of the joint model, based on participant's request parameter τ_{down}. Each participant downloads a part of the joint model weights and updates the local model accordingly. The training is performed until the desired accuracy is achieved. Such training achieves higher accuracy than training on a local and rather smaller set of data. Additionally, such training provides a private solution for distributed NN training due to following properties: i) each participant performs collaborative NN training while keeping all the training data on the participant's premises; ii) all participants learn the joint model and can use it privately and locally; iii) adding DP ensures that parameters' updates do not leak too much information about any individual point in the training dataset; iv) each participant fully controls which gradient to share/download and may decide not to share particularly sensitive ones.

Algorithm 1 Privacy Preserving Collaborative Training Algorithm

Client side

```
1: ppct_training($\mathcal{M}, \mathcal{D}, \tau_{down}, \tau_{up}$)
2: for $i \leftarrow 1, epochs$ do
3:     $\mathcal{M}_g \leftarrow$ download_weights($\tau_{down}$)
4:     $\mathcal{M} \leftarrow \mathcal{M}_g$
5:     train $\mathcal{M}$ on $\mathcal{D}$
6:     $\theta \leftarrow \mathcal{M}_i - \mathcal{M}_{i-1}$
7:     $\theta_{up} \leftarrow$ choose_gradients($\theta, \tau_{up}$)
8:     upload_gradients($\theta_{up}$)
9: end for
10: Return $\mathcal{M}$
```

Server Side

```
1: upload_gradients($\theta$)
2: $\mathcal{M}_g \leftarrow \mathcal{M}_g + \theta$
1: download_weights($\tau_{down}$)
2: Return $\tau_{down}$ percent of $\mathcal{M}_g$ weights
```

2.4 Attacks on Collaborative Training

Although during collaborative training participants' data remains on their own premises, in recent years, researchers presented attacks on neural networks that could target collaborative training. Attacks on Collaborative Training might be considered when an adversary participant succeeds to infer any sensitive information regarding other participant's training data or when an adversary succeeds to impact the utility of the training process or damage the accuracy of the NN model. Although in our settings all participants are considered as honest, sensitive information, such as data sample membership, can still be leaked when an adversary is allowed to send classification requests. Membership Inference attack [5] (MIA) allows the adversary to determine whether or not the data sample used in the model's training data. By definition DP is a defence against MIAs. However, adding DP noise to the NN model might significantly impact model utility. In Section 3.1 we evaluate and observe the results of Privacy Preserving Collaborative Training with DP.

3 Results

We evaluated the implemented collaborative training for the stress detection task on the SWELL dataset [2] and CNN model as described in Sec. 2.2. We implemented the collaborative training stress detection in python using Keras [7] framework. The evaluation is performed for two main settings: with Differential Privacy noise and without. In this section we describe the achieved results of each approach and describe pros, cons, the discovered gaps and future work.

3.1 Setup for Stress Detection Collaborative Training

We evaluate the method for three participants. In order to emphasise the strength of collaborative approach we intentionally created an approximation error for each participant. Namely, we used only 10% of the dataset. The dataset was randomly sampled and equally split between three participants. Each participant obtained a local dataset of $\sim 3\% \ (12K)$ samples. The validation accuracy of each participants is evaluated on the same dataset (12% size, 44K data samples). It is worth mentioning that the stress detection CNN hyperparameters space was explored in non collaborative settings and the best hyperparameters were applied for collaborative training in order to achieve fair comparison. The CNN trained with Adam optimizer [8], batch size of 32 and learning rate 0.001. Additionally, we explored the collaborative training parameter space and the presented results are corresponding to the best parameters setup. All participants uploaded their gradients and downloaded the weights from the joint model on each training iteration (as described in Algorithm 1).

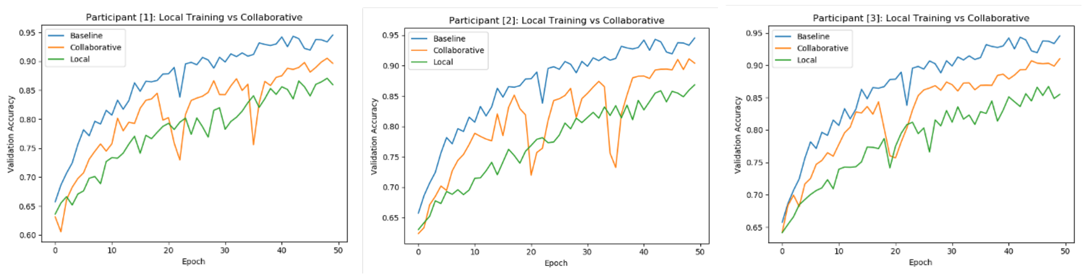

Figure 2: Validation Accuracy (without DP). $\tau_{up} = 0.5$, $\tau_{down} = 0.4$

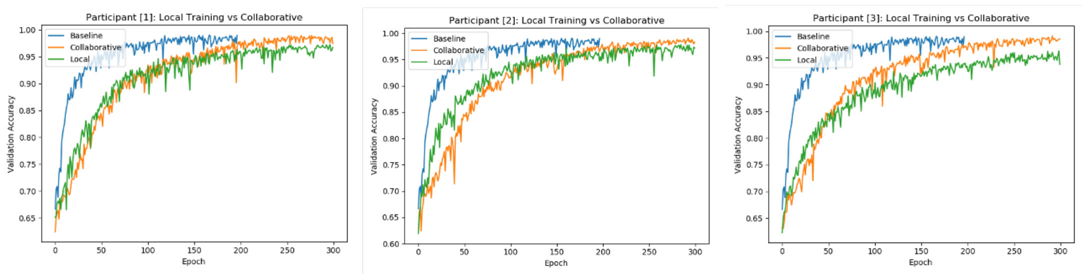

Figure 3: Validation Accuracy (with DP). $\tau_{up} = 0.1$, $\tau_{down} = 0.1$

Figures 2 and 3 present a comparison of validation accuracy for each participant between three setups: baseline, local and collaborative. Baseline result is defined when the stress detection model is trained on the centralized dataset (10%). Local result presents a setup when each participant performs training only on a local data. Collaborative result presents a setup when all three participants perform collaborative training.

In Figure 2 we can observe that the collaborative approach exceeds the results of training on the local data only, although being less accurate than the baseline. The advantages of the collaborative approach can be seen starting from the very first epochs. By sharing only part of the gradients and leaving the participants' data on their premises, some privacy has been achieved. However, there is no way to quantify the privacy guarantee.

As the second setup, we evaluated the method with differential private noise based on the approach in [4]. In addition we added averaging functionality to the server. The server aggregates updates from each participant and averages the value of the updates based on the number of participants.

In Figure 3 we can observe that the collaborative approach exceeds the results of training on the local data only, and almost as accurate as the baseline. However, the convergence time is increased due to added noise and updates averaging performed by the server. We chose to share and download only a small part of the gradients/weights in order to reduce the global DP ε.This approach provides a good privacy guarantee per models parameter (per gradient). However, the total privacy guarantee is $\varepsilon = 7 * 10^5$, which by definition has no privacy guarantee with this value. We are not aware of any practical membership inference attack even for such high DP epsilon value. Thus, the efficiency of the attack needs to be checked empirically.

4 Conclusion

In this paper we presented a solution for performing collaborative training of a stress detection model, employing privacy-enhancing technologies to preserve workers' privacy. The conducted work proves the advantages of combining collaborative learning techniques with privacy-enhancing technologies in healthcare-related contexts, where a limited amount of data may be available (for instance due to data controllers unwilling to release data to third parties or patients unwilling to disclose their health data) and low-performance models would be created out of small training datasets. Our work showed that in the practical scenario of stress detection, where data subjects are rarely interested in releasing data for research purposes, collaborative training provides better results with respect to training on local data only. The proposed solution enables the creation of consortia of data sources that can contribute to the construction of high-performing models while preserving the privacy of the involved data subjects.

As a future work, we could use the stress detection model presented in this paper on a real population of users, to assess its performance in a real-life scenario. Moreover, we could adapt the collaborative training approach to other scenarios in the healthcare sector, such as fatigue assessment in elderly patients and sleep quality classification.

Acknowledgments

This work was partly supported by the PAPAYA project funded by the European Union's Horizon 2020 Research and Innovation Programme, under Grant Agreement no. 786767.

References

[1] Sharma and Gedeon, "Objective measures, sensors and computational techniques for stress recognition and classification: A survey". Computer methods and programs in biomedicine, vol.108, no.3, pp. 1287–1301, 2012.

[2] Koldijk et al., "The SWELL knowledge work dataset for stress and user modeling research". Proceedings of the 16th international conference on multimodal interaction, pp. 291–298, 2014.

[3] Jiayuan et al., "Real-time detection of acute cognitive stress using a convolutional neural network from electrocardiographic signal". IEEE Access, vol.7, pp. 42710–42717, 2019.

[4] Shokri et al., "Privacy-preserving deep learning". Proceedings of the 22Nd ACM SIGSAC Conference on Computer and Communications Security, pp. 1310–1321, 2015.

[5] Shokri et al., "Membership inference attacks against machine learning models". 2017 IEEE Symposium on Security and Privacy (SP), pp. 3-18, 2017.

[6] The SWELL heart rate variability (HRV) dataset for research on stress and user modeling: https://www.kaggle.com/qiriro/swell-heart-rate-variability-hrv

[7] Chollet et al., (2015). Keras. GitHub. Retrieved from https://github.com/fchollet/keras

[8] @Diederik et al., "Adam: A Method for stochastic optimization".3rd International Conference on Learning Representations 2015