LOSS ADAPTED PLASTICITY: LEARNING FROM DATA WITH UNRELIABLE SOURCES

Anonymous authors
Paper under double-blind review

ABSTRACT

When data is streaming from multiple sources, conventional training methods update model weights often assuming the same level of reliability for each source; that is: a model does not consider data quality of a specific source during training. In many applications, sources can have varied levels of noise or corruption that can produce negative effects on the learning of a robust machine learning model. A key issue is that the quality of data or labels for individual sources is often not available to a model during training and could vary over time. A solution to this problem is to consider the mistakes made while training on data originating from sources and utilise this to create a perceived data quality for each source. This paper demonstrates a technique that can be applied to any gradient descent optimiser: Update model weights as a function of the perceived reliability of data sources within a wider data set. The algorithm controls the plasticity of a given model to weight updates based on the history of losses from individual data sources. We show that applying this technique can significantly improve model performance when trained on a mixture of reliable and unreliable data sources, and maintain performance when models are trained on data sources that are all considered reliable.

1 INTRODUCTION

In many machine learning applications, data is recorded by multiple sources, which are then merged together to form a broader dataset. For example, in healthcare monitoring applications, data from multiple machines or sites are often pooled together into a single dataset Baro et al. (2015); Tomar & Agarwal (2013). Within a dataset, data sources may not provide the same level of data quality and may include noisy observations, incorrect labelling or missing values. In general, in machine learning applications, models are trained using multi-source datasets, often without considering the reliability of underlying data sources. We propose a technique that can be implemented alongside any gradient descent optimisation algorithm to distinguish between reliable and unreliable data over time, mimicking the way children learn from multiple information sources. Our solution proposes that model weight plasticity can be a function of the perceived reliability of data sources. To achieve this, our technique maintains a distribution of weights over the set of data sources, which are used to control the back-propagation of gradient updates. Within this work, we focus on the data corruption of observations and labels, and do not consider missing data directly; although one of our experimental contexts could be considered a special case of learning from missing data.

2 RELATED WORK

Children are effective at learning from testimony and do not accept to learn from any speaker, but rather learn from a speaker that they consider trustworthy and knowledgeable (Scofield & Behrend, 2008). Children will then begin to give greater weight to trustworthy sources as a result of their experiences, learning less from perceived untrustworthy sources (Koenig & Echols, 2003). It is this natural dynamic weighting of source trust that we wish to capture in our proposed algorithm.

In machine learning, learning from multi-sourced data (multiple devices, deployment sites, or manual labellers) is often done without much regard for each source's data quality. However, when sources have reduced data quality, training methods should recognise when not to learn or with

greater caution to produce the most optimal model (Song et al., 2022). Frequent approaches to this challenge utilise methods that cope with noisy data, noisy labels, or both, as well as federated learning models. Song et al. (2022) provides a survey of techniques used to deal with noisy labels, in particular. A common method is to apply pre-processing or filtering techniques before training; during which labels or data suspected of being unreliable are removed or rectified. Several other methods consider changing the architectures of models to ensure they are robust to noise in labels (Sukhbaatar & Fergus, 2014; Goldberger & Ben-Reuven, 2016), limiting them to classification problems, and do not address the problem of noisy observations. Other work focuses on the task of dealing with noisy labels through the use of regularisation, robust loss design, and sample selection (Song et al., 2022). Robust loss design can involve loss re-weighting; in which loss values and data points are weighted based on, for example, meta-learning methods (Ren et al., 2018) or using exponential gradient re-weighting (Majidi et al., 2021). However, these works treat data points independently from the sources that they originate from. Majidi et al. (2021) specifically, store weights for each data point, limiting the method to offline training only. This differs from our proposed technique, in which weights are stored per source, meaning new data from existing sources can be weighted immediately, allowing for its use in continual learning.

Federated Learning allows for training models on data from multiple sources whilst respecting each source's privacy. The server trains a global model based on local updates of models on each source device, eliminating the requirement for a centralised server to store and handle data (Konečnỳ et al., 2016). When data sources are unreliable (i.e., client data is corrupted or potentially malicious), federated learning algorithms may fail (Li et al., 2020), as local model updates can have negative effects on the global model. Jebreel et al. (2022) and Stripelis et al. (2022) discuss solutions to situations in which unreliable data is malicious (i.e., in adversarial attacks), but do not consider noisy observations from benevolent sources. To tackle the issue of varied data quality in client data, Li et al. (2021) proposed a novel federated learning approach called Auto-weighted Robust Federated Learning (ARFL) which learns the global model and the weights of local updates simultaneously. We will refer to this model as FED ARFL to emphasise its grounding in Federated Learning. The weighted sum of empirical risk of clients' loss is minimised with a regularisation parameter, where the weights are distributed by comparing each client's empirical risk to the average empirical risk of the best k clients. The contribution to weight updates of the clients with higher losses are minimised in respect to the updates from other clients; allowing the global model to learn more from clients that achieve lower losses. FED ARFL will serve as a baseline model in our study as it is the closest research in the literature to the setting that we investigate. However, in our testing, we allow for the transfer of data to the server.

3 Preliminaries and Motivation

When designing predictive models for analysing multi-source/multi-site data, it is common to treat data sources as having equal reliability (Baro et al., 2015; Tomar & Agarwal, 2013). Predictive models trained on unreliable data could lead to reduced performance and potentially, unforeseen consequences when deployed in production.

Integrating multi-source data into single datasets before performing model training and testing could result in the reduction of model performance on reliable data. Figure 1 shows the decision boundaries of a logistic regression model trained on data from a reliable source, an unreliable source, and both. In this experiment, 60% and 40% of the data is chosen to be reliable and unreliable respectively. We see that unreliable data sources can contaminate the training data and significantly reduce a model's performance.

These challenges motivate us to design a technique that can be applied to gradient descent optimisers. The objective is to control the influence of unreliable data sources on the weight updates of a deep neural network and satisfy the following criteria: **1:** Identify and reduce the level of trust of unreliable data sources, without compromising learning on reliable data. **2:** Update the level of previously unreliable data sources, if data from these sources becomes reliable again (by measuring the error from these sources). **3:** Be easily implementable alongside existing models.And **4:** Require limited additional compute power. These requirements would make the technique suitable in diverse contexts, and applicable to a variety of deep learning architectures.

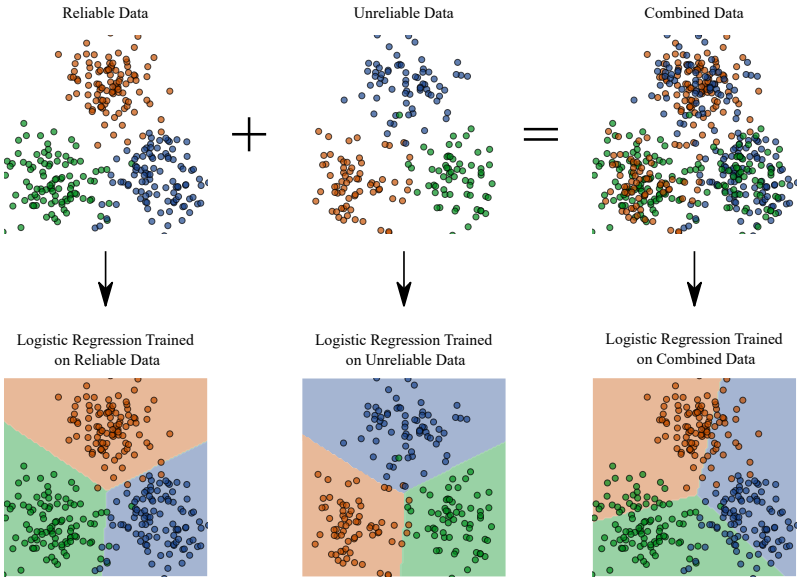


Figure 1: Example of distribution and logistic regression model applied to a database made up of 3 classes. The three graphs at the bottom show the areas of the plane for which a Logistic Regression model predicted the class with their corresponding colour. The bottom left one shows the boundaries after being trained on the reliable data sources only; the one in the centre shows the boundaries when trained on the unreliable sources; and the one on the right shows the boundaries for a combination of the two, overlaid on the reliable data.

4 Methods and Evaluation

Within deep learning, gradient descent algorithms are one of the most common techniques to optimise a model's weights for a task. Therefore, the proposed technique exploits the wide use of gradient descent optimisers through an addition that can calculate which sources to consider as reliable and unreliable.

4.1 Unreliable Data Sources

As discussed in Section 3, unreliable data sources can be introduced in a multitude of ways, affecting both the independent and dependent variables of data. To create unreliable data for controlled and verifiable experiments in this paper, we apply synthetic corruption techniques summarised in Table 1. In line with Li et al. (2021), we apply label flipping, label shuffling and the addition of Gaussian noise to inputs. Additionally, we choose random labels, replace data with noise and shuffle chunks of the input data to extend the results.

To implement the corruption, data is split into mutually exclusive sources of data points, before a number of these are chosen to be unreliable and corrupted. For each experiment, a single corruption technique is used for all unreliable sources.

4.2 Loss Adapted Plasticity

To learn from data sources with mixed reliability, we propose a technique able to discern the reliable from the unreliable sources, based on the history of losses incurred by a model during training. When a model frequently produces a loss on a given source significantly larger than the losses on other sources, the model becomes less plastic when updating its weights to that source.

We start by assuming that losses generated by sources over a relatively small number of steps follow a normal distribution, which is discussed in Appendix A.4, alongside experimental results. Given a set of sources S, a history of losses for each source $\{L_s^t\}_{s\in S}^{\tau-h+1,\tau}$ (with h denoting the history length and τ denoting the current step), a trust level for each source $\{R_s\}_{s\in S}$ (in which $R_\zeta \in [0, +\infty)$, and

Table 1: Methods used to produce unreliable data sources. *Techniques employed in Li et al. (2021).

Corruption Type	Description of Corruption Applied
No Corruption	No corruption is applied to the data.
Chunk Shuffle	Split the data into distinct chunks before shuffling. This is only done on the first, or first and second axis of a given input.
Random Label	Randomly assign a new label to an input, from the same domain.
Batch Label Shuffle*	For a given batch of input data, randomly shuffle the labels.
Batch Label Flip*	For a given batch of input data, assign all inputs in this batch a label randomly chosen from the labels in the batch.
Added Gaussian Noise*	Add Gaussian noise to a given input, sampled from the normal distribution, with mean $= 0$ and standard deviation $= 1$.
Replace with Gaussian Noise	Replace a given input with Gaussian noise, sampled from the normal distribution, with mean $= 0$ and standard deviation $= 1$. This can be considered a special case of learning from sources with missing data.

a larger value corresponds to a lower level of trust in ζ), the updated trust at time step τ on source ζ is calculated using the following process:

We calculate the weighted average, μ and standard deviation, σ of the loss history over all sources and time-steps cached, $[\tau - h + 1, \tau]$, excluding the source ζ currently being updated, and with weights $= \{1/(1 + R_s)\}_{s \neq \zeta \in S}$. This weighting forces sources that are trusted less, to have less of an influence on the mean and standard deviation of source losses. The formulas are as follows:

$$\mu = \frac{\sum_{s \neq \zeta} \sum_{t=\tau-h+1}^{\tau} \frac{1}{1+R_s} L_s^t}{\sum_{s \neq \zeta} \frac{1}{1+R_s}}, \quad \sigma = \sqrt{\frac{\sum_{s \neq \zeta} \sum_{t=\tau-h+1}^{\tau} \frac{1}{1+R_s} \times (L_s^t - \mu)^2}{\sum_{s \neq \zeta} \frac{1}{1+R_s}}} \tag{1}$$

Then, we calculate the mean loss from the history of the current source, μ_ζ:

$$\mu_\zeta = \frac{\sum_{t=\tau-h+1}^{\tau} L_\zeta^t}{h} \tag{2}$$

And use μ_ζ, μ, and σ in the following to update the current source's trust level, R_ζ:

$$R_\zeta = \begin{cases} -1 & \mu_\zeta < \mu + \lambda\sigma \\ +1 & otherwise \end{cases} \tag{3}$$

Where λ is defined as the strictness parameter; used to control the probability that a source producing reliable data is considered less trustworthy. R_ζ can be interpreted as how reliable a source is considered; the larger the value, the less reliable a source is considered, and the less plastic the model would be to updates based on source ζ. If $R_\zeta < 0$ after an update, its value is modified to 0, to ensure that sources that are deemed reliable at the start can be considered unreliable without delay. This plasticity is controlled through the scaling of gradients in the gradient descent optimiser. Each back-propagation gradient g is scaled using:

$$\hat{g} = d_\zeta g, \quad d_\zeta = 1 - \tanh^2(\delta R_\zeta) \tag{4}$$

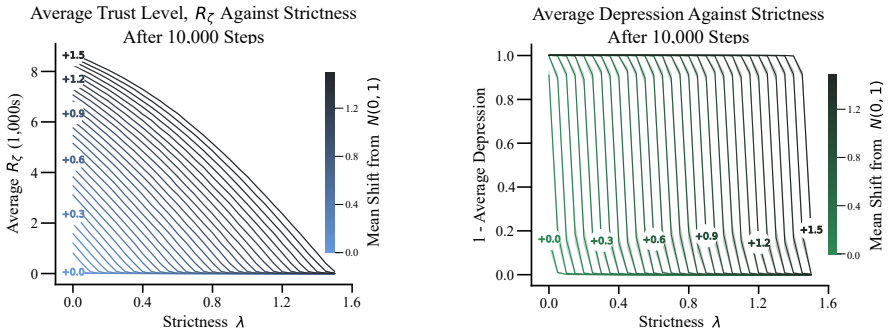
Here, $\hat{g}$ represents the updated gradients after the gradients g are depressed by the scalar d_ζ, in which δ corresponds to the discrete step size (referred to as the depression strength) used to control the rate of change in depression applied. For two sources that are reliable and unreliable respectively, say ζ_r and ζ_u, we expect that $d_{\zeta_r} > d_{\zeta_u}$. This would correspond to less plasticity in the weight updates for a source that is unreliable compared to a source that is producing reliable data. Since $0 < d_\zeta \leq 1$, the gradients are only ever depressed by this process.

Within this technique, we introduce two main hyper-parameters, namely the strictness (λ) and the depression strength (δ), which control the criteria for increasing R_ζ and the rate of change of the depression respectively.

Figure 2 shows how the trust level and depression can be affected by the strictness parameter, assuming that losses generated by sources form a normal distribution at any given training step. Here, the loss distribution on reliable data is given mean $= 0$ and standard deviation $= 1$. Each line represents a source that is producing losses with a mean given by the hue, and a standard deviation $= 1$. The

values are then calculated by taking the average of 100 walkers, moving under Equation 3, sampling values from the same distribution for $10,000$ steps, given by the hue. The larger the shift in mean, the more unreliable a source is considered (given by larger values in R_ζ - Figure 2a) and the less plastic the weights will be to updates on this source (given by a lower value in $1 -$ Average Depression - Figure 2b). The strictness value allows us to control the mean value of loss distributions for which the technique should depress gradient updates. It also allows us to calculate the probability of a source producing reliable data having its level of trust reduced (R_ζ increased).

The depression strength parameter, δ, controls the rate of change of the depression. Too small a δ value and the unreliable sources may have a significant negative effect on the training of a model, however, too large a value and gradient updates from reliable sources may be incorrectly depressed. From our experiments, we found it helpful to split the depression parameter into $\delta_s = \delta \times 0.005$. Using δ_s allowed for greater control over the parameter and meant that when R_ζ is on the order of 1000, the value of d_ζ is close to 0.



(a) (b)

Figure 2: a: 100 random walkers produced values sampled from the Normal distribution with mean equal to the hue of the figures and standard deviation 1. Each time that value was more than the strictness, λ given by the x axis, the walker increased R_ζ by $+1$, otherwise, it decreased R_ζ by -1 as in Equation 3. This step was completed $10,000$ times, to produce the figure. A larger R_ζ represents a lower level of trust. b: The average depression calculated using R_ζ from each of the walkers in Figure a. A larger value on the y axis ($1 -$ Average Depression) represents a lower level of weight plasticity ($1 - d_\zeta$).

We refer to this technique as Loss Adapted Plasticity (or LAP). The code[1] was implemented in Pytorch 1.10 (Paszke et al., 2019) for both Stochastic Gradient Descent and the Adam algorithm (Kingma & Ba, 2014). For further notes on the practical use of LAP, please see Section 6.

5 Experiments and Results

To investigate the performance of the Loss Adapted Plasticity (LAP) algorithm, we employ image datasets and synthetically produce unreliable sources before training and evaluating models using LAP and no LAP. In addition, we compare our technique against FED ARFL; the training algorithm detailed in Li et al. (2021). We will evaluate all models on reliable, uncorrupted data. We choose to focus on synthetic experiments to ensure that our experiments are reproducible and accessible. Additionally, we test our algorithm on real-world data, which is presented in Appendix A.3.1.

5.1 Datasets

In order to evaluate the proposed techniques, three datasets are employed: CIFAR-10 and CIFAR-100 (Krizhevsky et al., 2009), and Fashion-MNIST (F-MNIST) (Xiao et al., 2017). CIFAR-10 and CIFAR-100 are datasets made up of $60,000$ RGB images of size 32×32 divided into 10 and 100 classes, with $6,000$ and 600 images per class respectively. F-MNIST is made up of $70,000$

[1]Github link will be added upon completion of the review process. Code is also available in the supplemental material.

greyscale images of clothes of size 28×28 divided into 10 classes with $7,000$ images per class, which was flattened into datapoints with 784 features. All data used is managed under the MIT License. These datasets are chosen as they are three of the most notable datasets that are employed for object classification. Moreover, Li et al. (2021) uses CIFAR-10 to benchmark their model, aligning our evaluation with the Federated Learning literature.

To implement the notion of reliable and unreliable data sources, data was split into 10 distinct groups, with datapoints randomly assigned to each. For CIFAR-10 and CIFAR-100, 4 and 2 of these sources were chosen to be unreliable respectively, whilst for F-MNIST 6 were chosen to be unreliable. We chose different levels of data corruption for testing, to understand whether LAP would perform well in different environments, and in line with the perceived difficulty of the datasets. Unreliable data was corrupted using the techniques discussed in Section 4.1, and remained corrupted in the same way for all of the training. Data from each source is batched in sizes of 128 for CIFAR-10 and sizes of 200 for F-MNIST. Each batch contains data from a single source to ensure that weight updates corresponded to single sources, allowing us to control plasticity of weights determined by the data source. Each dataset was split further into a training and validation set with ratio $3 : 1$ to tune hyper-parameters.

5.2 MODELS

After the data is loaded into batches, and reliable and unreliable data sources are chosen, the training process begins. Multiple models and training procedures are used to evaluate the performance of Loss Adapted Plasticity (LAP), which are detailed below:

1. **CIFAR-10** and **CIFAR-100**: A convolutional network is chosen consisting of the following architecture: A convolutional layer with 32 channels and a kernel size of $(3, 3)$ with a ReLU activation function and a max pooling layer with kernel size $(2, 2)$; a convolutional layer with 64 channels and a kernel size of $(3, 3)$ with a ReLU activation function and a max pooling layer with kernel size $(2, 2)$; a convolutional layer with 64 channels and a kernel size of $(3, 3)$ with a ReLU activation function and a max pooling layer with kernel size $(2, 2)$; a fully connected layer with size $(1024, 64)$ and a ReLU activation layer; and another fully connected layer with size $(64, 10)$ for CIFAR-10 and $(64, 100)$ for CIFAR-100. This model is used for all experiments, including those using LAP and for FED ARFL (Li et al., 2021). All these models are trained for 25 epochs, with the Adam algorithm (Kingma & Ba, 2014), and a learning rate of 0.001. For FED ARFL global updates were performed 25 times, in which all clients are trained on data for a single epoch, to keep training consistent across techniques.
2. **F-MNIST**: A fully connected model is chosen consisting of the following architecture: A fully connected layer of size $(784, 16)$ with dropout and a ReLU activation function; a fully connected layer of size $(16, 16)$ with dropout and a ReLU activation function; a fully connected layer of size $(16, 10)$ with a softmax activation function. The models are trained for 40 epochs using the Adam algorithm (Kingma & Ba, 2014), and a learning rate of 0.001. For FED ARFL global updates were performed 40 times, in which all clients are trained on data for a single epoch.

Code for FED ARFL is made available by the authors Li et al. (2021), with an MIT Licence.

5.2.1 TRAINING

To evaluate the performance of LAP against the baseline models, we test four different training setups:

1. **Standard Model**: Trained with no knowledge of the sources and whether they are reliable or not.
2. **Corruption Oracle**: Trained with the knowledge of which sources are reliable and unreliable. This model only makes weight updates when the data batch consists of reliable data. This was done to give context to the results.
3. **LAP Model**: Trained with the knowledge of the source label, but not whether it is reliable or not. Here, we use the LAP extension to Adam with a loss history length of 25 for

CIFAR-10 and CIFAR-100, and 50 for F-MNIST, a depression strength of $\delta_s = 1.0$, and a strictness of $\lambda = 0.8$. The longer history length for F-MNIST was chosen because the model was trained for a larger number of epochs. λ and δ_s were chosen using the validation dataset and the analysis in Section 4.2. When training on CIFAR-100, we started applying LAP after 250 gradient update steps, for reasons discussed in Section 6.4.

4. **FED ARFL** (Li et al., 2021): The model as described in Section 5.2, however, each source (or client in the context of Federated Learning) corresponds to a single instance of the corresponding network described in Section 5.2. All other parameters were kept as default in the code made available by the authors (Li et al., 2021).

Each experiment is performed 10 times with different random seeds to ensure reproducibility.

6 Results and Discussion

We present the results of the experiment and the accuracy of each model, trained on unreliable and reliable data sources, and evaluated on reliable data to show that Loss Adapted Plasticity (LAP) provides a solution to training with data sources of mixed reliability. For experiments on real-world data, please see Appendix A.3.1.

Figure 3 shows the results of the models described in Section 5.2. LAP training out-performs both the standard training method, and FED ARFL. LAP also allows the model to achieve results often comparable to the model with knowledge of the reliability of a data batch (Corruption Oracle model).

Figures 3a and 3b show that LAP helped the models significantly out-perform both the standard training method and FED ARFL for the CIFAR-10 and CIFAR-100 datasets in which the labels are used to corrupt the source data (random label, label shuffling and batch flip). Here, LAP training also outperforms both of the baseline models when the input data is corrupted, although less significantly. Moreover, when trained on reliable data, both the standard training and LAP training performed similarly, suggesting that LAP is able to boost performance when trained on combinations of reliable and unreliable data, without reducing the performance of models trained on reliable data.

On F-MNIST, the LAP trained model outperforms the standard trained model, and in some cases, the corruption oracle too. Furthermore, Figure 3c shows that the LAP and standard trained model performed equally when trained on data that was reliable.

Appendix A.1 shows the test results on CIFAR-10 with batch label flipping and the same experimental set-up as in Figure 3a for varying values in the depression strength, the strictness, and history length.

6.1 Is Lost Adapted Plasticity Identifying the Correct Sources?

Figure 4a shows the depression applied to the gradient updates (d_ζ in Equation 4, Section 4.2), for the 10 data sources in training, on the CIFAR-10 dataset. Here, 4 of the data sources are unreliable, and the LAP technique is able to correctly identify these, and reduce their gradients.

6.2 Does Lost Adapted Plasticity Detect When Unreliable Sources Become Reliable?

Figure 4b, shows the depression applied to the gradients in updates on each of the sources, split by their reliability. Initially, 4 sources were unreliable, and the LAP technique depressed the gradient updates on these batches, reducing weight plasticity for the corresponding sources. However, once these sources produced reliable data, the technique reduced the amount of depression applied to the gradient updates, and eventually these sources were treated equivalently to reliable sources.

6.3 Does Lost Adapted Plasticity Add Extra Compute Time to the Optimisation Process?

The extra computations required in the optimiser to calculate the depression to apply to a single batch of source data is $O(h \times |S|)$, where S is the number of sources and h is the number of losses to cache

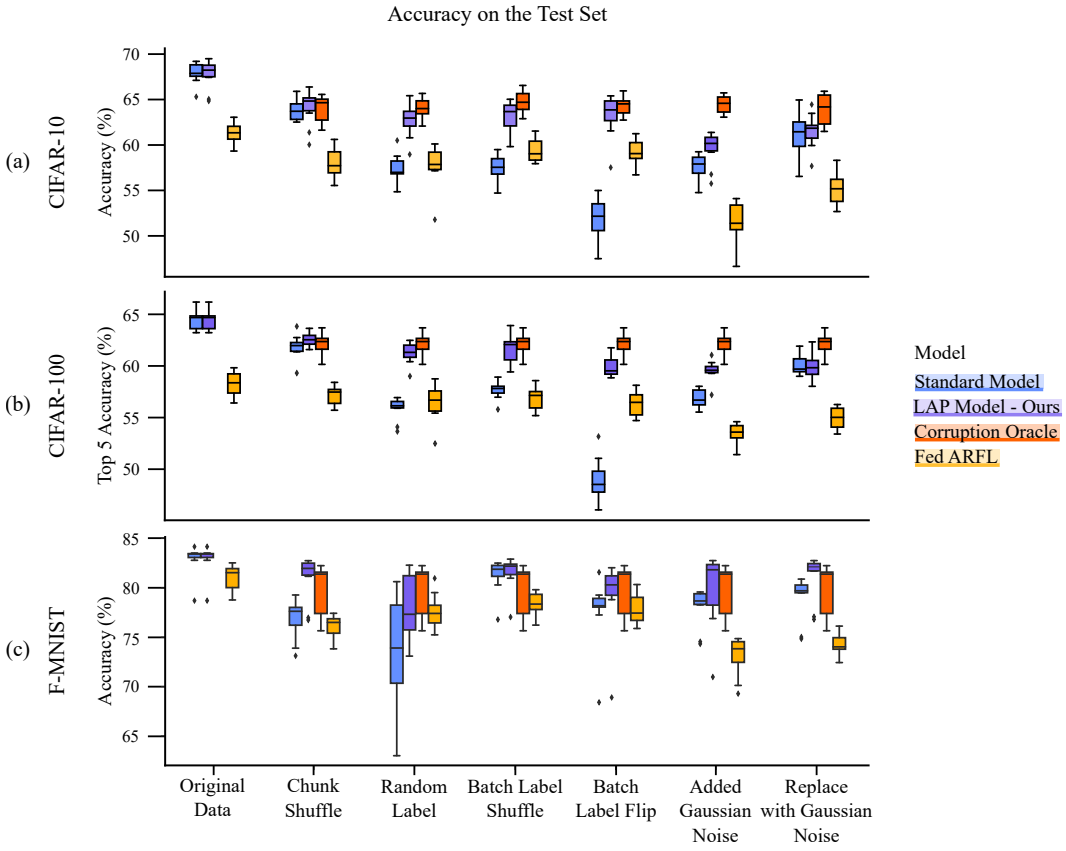


Figure 3: Accuracy when trained on unreliable data and tested on reliable data, with 10 repeats. Box-plot whiskers represent $1.5\times$ the interquartile range. a: CIFAR-10; 4 out of 10 total sources were corrupted. b: CIFAR-100; 2 out of 10 total sources were corrupted. c: F-MNIST; 6 out of 10 total sources were corrupted.

for each source. This is also the extra memory cost of using LAP. Training on a single NVIDIA RTX 3080 TI, using Pytorch (Paszke et al., 2019) and the Adam (Kingma & Ba, 2014) implementation, the times to complete the backward pass for a batch size of 128 on CIFAR-10 (for model and training description, see Section 5.2, and for dataset description see 5.1) are 0.0026 ± 0.0006s and 0.0088 ± 0.0021s (mean $\pm$ standard deviation) for the standard training procedure and for LAP training respectively. For the entire forward and backward pass, the computational times are 0.0038 ± 0.0007s and 0.0098 ± 0.0021s respectively. In total, this gives the additional computational time for the forward and backward pass per batch of 0.0060 ± 0.0023s for this experimental set-up.

6.4 When To Start Applying Lost Adapted Plasticity

LAP first requires the model to learn which of the sources are producing data from the same distribution and those that are not. Liu et al. (2020) and Xia et al. (2021) discuss the learning mechanics of deep learning models; models first fit to reliable data before fitting to unreliable data. Therefore LAP training should commence after the initial model warm-up stage to ensure that it correctly identifies the unreliable sources. In some cases, it might take several steps before the model is able to discern between reliable and unreliable sources. An example of the early learning of a model on CIFAR-100 is given in the Appendix A.2. To control when LAP is applied, we have added a hold-off parameter to our implementation, which corresponds to the number of update steps after loss histories are filled before gradient depression commences.

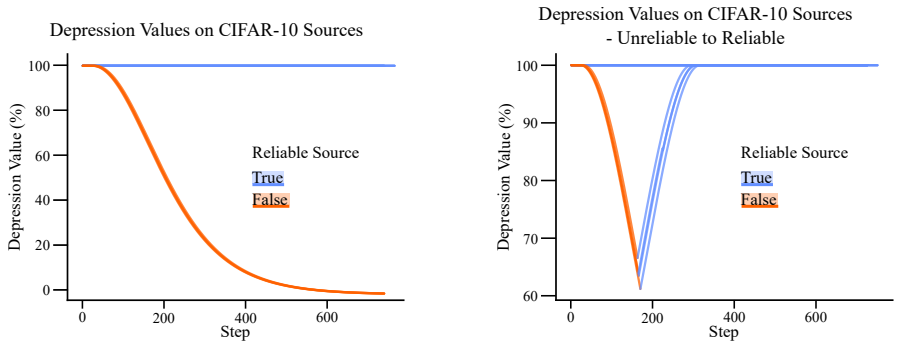


Figure 4: a: The depression values for each of the sources when training the model using LAP. In total there are 6 reliable sources and 4 unreliable sources (corrupted using random labelling), corresponding to the hue of the lines. b: The depression values for the LAP trained model on CIFAR-10, in which the 4 sources become reliable after an amount of steps, corresponding to the hue of the lines. In both, the depression value represents the value of d_ζ in Equation 4 and a value of 1 represents no depression applied to the gradient updates for that given batch.

6.5 Limitations

When using Loss Adapted Plasticity (LAP), thought should be given to the underlying data and causes of unreliable data sources. Data sources that are producing consistently low quality measurements will be identified and prevented from negatively affecting the learning on the reliable data. Therefore, LAP training minimises the immediate consequences of training a model on data consisting of reliable and unreliable data sources, and for many applications will produce a final model. Moreover, in some contexts, the useful information that LAP training provides about the data sources and their individual perceived reliability will be helpful in itself for improving the quality of a dataset. It should be noted that in some cases users want to learn a model for the unreliable data. An example could be learning a model to monitor electrocardiogram (ECG) recordings in which the dataset consists of readings from different machines at different sites. A practitioner applying LAP can identify data sources that are not perceived to be from the same distribution as others, possibly arising from a calibration issue in one of the machines, and can act accordingly. In this case, the practitioner might take action by replacing the erroneous machine, or by fine-tuning the model on that machine's data to produce another predictive model. The key point being that with LAP training, a single machine's data issues will not cause negative effects on the learning of a model for monitoring on all other machines, and any data quality issues can be quickly identified.

7 Conclusion

In this research, we propose LAP, Loss Adapted Plasticity; as an addition to any gradient descent algorithm used for deep learning that improves model performance on data produced by multiple sources in which some of the sources are unreliable. LAP training also maintains model performance on reliable data and out-performs the benchmark from the field of Federated Learning (FED ARFL Li et al. (2021)) when training on unreliable data. For deep learning applications in which Federated Learning is not required, and data is produced by multiple data sources or at multiple sites, we believe that there is a strong argument for the use of LAP training to control the model updates on reliable data possibly contaminated by unreliable data sources. Future work will explore the use of LAP in a continual learning setting, in which data is streaming online from multiple data sources. Here, LAP may be used to control the model plasticity to data sources producing unreliable data, or data sources that could cause the model to have significant "forgetting" on previously learnt data.

8 ETHICS STATEMENT

This work presents an algorithm for improving the performance of a model on reliable data, when trained on data stemming from sources that are producing reliable and unreliable data. We see no immediate ethical consequences of our algorithm, and in fact would argue that Loss Adapted Plasticity (LAP) allows the practitioner to understand more about a model's learning on a dataset made of collective source data, by identifying those data sources that are producing data that differs from the majority distribution whilst improving model predictive performance.

9 REPRODUCIBILITY STATEMENT

All code is available within the supplementary materials and will be made available on GitHub once the reviewing process has concluded. Experimental results are made available and instructions are given for testing the models on new data or with different parameters. Additionally, all data and benchmarks used within this work are publicly available.

Moreover, all figures within this work contain colour palettes which are accessible for readers with colour blindness.

REFERENCES

Emilie Baro, Samuel Degoul, Régis Beuscart, and Emmanuel Chazard. Toward a literature-driven definition of big data in healthcare. *BioMed research international*, 2015, 2015.

Jacob Goldberger and Ehud Ben-Reuven. Training deep neural-networks using a noise adaptation layer. 2016.

Najeeb Moharram Jebreel, Josep Domingo-Ferrer, David Sánchez, and Alberto Blanco-Justicia. Defending against the label-flipping attack in federated learning, 2022. URL https://arxiv.org/abs/2207.01982.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2014. URL https://arxiv.org/abs/1412.6980.

Melissa A Koenig and Catharine H Echols. Infants' understanding of false labeling events: The referential roles of words and the speakers who use them. *Cognition*, 87(3):179–208, 2003.

Jakub Konečnỳ, H Brendan McMahan, Felix X Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 2016.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Shenghui Li, Edith Ngai, Fanghua Ye, and Thiemo Voigt. Auto-weighted Robust Federated Learning with Corrupted Data Sources. *arXiv:2101.05880 [cs]*, March 2021. URL http://arxiv.org/abs/2101.05880. arXiv: 2101.05880.

Tian Li, Anit Kumar Sahu, Ameet Talwalkar, and Virginia Smith. Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3):50–60, 2020.

Sheng Liu, Jonathan Niles-Weed, Narges Razavian, and Carlos Fernandez-Granda. Early-learning regularization prevents memorization of noisy labels. *Advances in Neural Information Processing Systems*, 33, 2020.

Negin Majidi, Ehsan Amid, Hossein Talebi, and Manfred K. Warmuth. Exponentiated gradient reweighting for robust training under label noise and beyond, 2021. URL https://arxiv.org/abs/2104.01493.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 32*, pp. 8024–8035. Curran Associates, Inc., 2019. URL http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. Learning to reweight examples for robust deep learning. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 4334–4343. PMLR, 10–15 Jul 2018. URL https://proceedings.mlr.press/v80/ren18a.html.

Jason Scofield and Douglas A Behrend. Learning words from reliable and unreliable speakers. *Cognitive Development*, 23(2):278–290, 2008.

Hwanjun Song, Minseok Kim, Dongmin Park, Yooju Shin, and Jae-Gil Lee. Learning from noisy labels with deep neural networks: A survey. *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

Dimitris Stripelis, Marcin Abram, and Jose Luis Ambite. Performance weighting for robust federated learning against corrupted sources, 2022. URL https://arxiv.org/abs/2205.01184.

Sainbayar Sukhbaatar and Rob Fergus. Learning from noisy labels with deep neural networks. *arXiv preprint arXiv:1406.2080*, 2(3):4, 2014.

Divya Tomar and Sonali Agarwal. A survey on data mining approaches for healthcare. *International Journal of Bio-Science and Bio-Technology*, 5(5):241–266, 2013.

Jiaheng Wei, Zhaowei Zhu, Hao Cheng, Tongliang Liu, Gang Niu, and Yang Liu. Learning with noisy labels revisited: A study using real-world human annotations. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=TBWA6PLJZQm.

Xiaobo Xia, Tongliang Liu, Bo Han, Chen Gong, Nannan Wang, Zongyuan Ge, and Yi Chang. Robust early-learning: Hindering the memorization of noisy labels. In *ICLR*, 2021.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

A Appendix

A.1 Parameter Testing

Figure 5 shows the results of LAP training on CIFAR-10 with the model described in 5.2, with batch label flipping applied to 4 of the 10 sources. Arguably, depression strength, $\delta = 1$; strictness, $\lambda = 0.8$ or 0.4; and history length, $h = 25$ are the most consistent parameter choices across corruption levels for this dataset, but different values might achieve higher performance in other applications.

A.2 Early Learning on CIFAR-100

Figure 6 shows the loss values, plotted by source when training on CIFAR-100 (model description in Section 5.2), with 2 unreliable sources and 8 reliable sources. In this example, the zoomed-in section shows that for the first ~ 10 steps, the difference in loss on the reliable and unreliable sources are not clear. For other datasets, this may require more time.

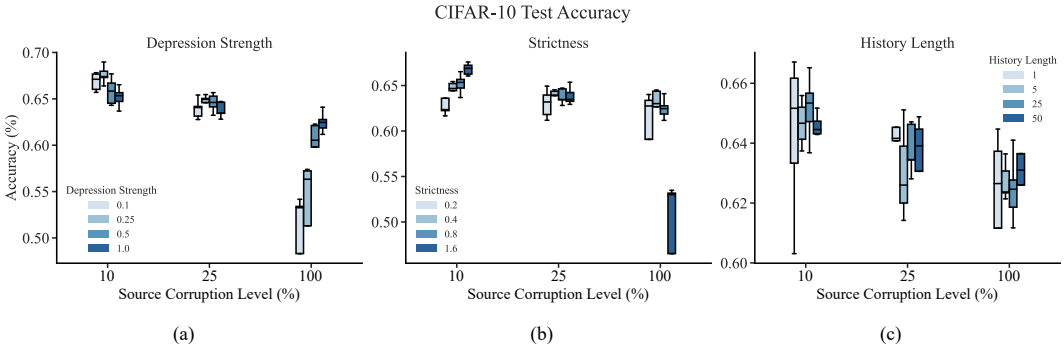


Figure 5: The test performance on CIFAR-10 after being trained on 4 unreliable sources (corrupted using batch label flipping) all at the given corruption level, and 6 reliable sources. The performance here is shown for various values of each new parameter described in Section 4.2. This figure is the result of 5 runs of each experiment. a: shows the performance for varied depression strength, δ values; b: for strictness, λ; and c: for history length, h. Unless otherwise stated, depression strength $= 1$, strictness $= 0.8$, and history length $= 25$.

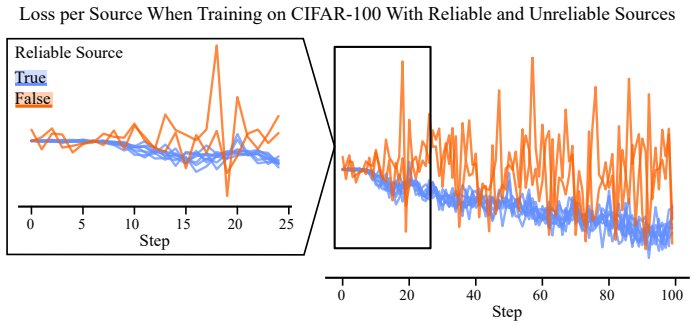


Figure 6: The loss values, plotted per source when training a model in the standard way on CIFAR-100, with 2 unreliable sources (corrupted using batch label flipping) and 8 reliable sources. The left hand graph shows a section of the right hand graph, with loss plotted for step values up to 25.

A.3 Real World Data

To understand the performance of Loss Adapted Plasticity (LAP) on real world data, a human labelled CIFAR-10 dataset was chosen:

A.3.1 Human Labelled CIFAR-10

The real-world dataset that we have used to better understand the performance of the LAP model is CIFAR-10N (Wei et al., 2022) [2], which contains human labels produced for the CIFAR-10 dataset. This dataset is often used in the literature surrounding learning from noisy labels and can provide a good test for the LAP model.

The CIFAR-10N dataset contains the human labels for the training set of images contained in the CIFAR-10 dataset (Krizhevsky et al., 2009). For our experiments, we chose the set of human labels titled "CIFAR-10N Worst", which contains the human labelling with the highest noise rate, 40.2%. We then binned the noisy labels, along with the clean labels into 10 sources with different noise levels. We chose combinations of 1, 2, 3 and 4 unreliable sources (containing noisy human labels) and noise levels of 25%, 50%, 75% and 100%, to get a complete understanding of the performance difference between training with and without LAP, on this real-world dataset. The training dataset was split in the ratio $3 : 1$ to produce a validation set that also contained unreliable data sources. The test set contained only clean labels and corresponded to the standard CIFAR-10 test set.

[2]Dataset available at `http://noisylabels.com/`

A convolutional network was tested on this dataset with the following architecture: A convolutional layer with 32 channels and a kernel size of $(3, 3)$ with a ReLU activation function and a max pooling layer with kernel size $(2, 2)$; a convolutional layer with 64 channels and a kernel size of $(3, 3)$ with a ReLU activation function and a max pooling layer with kernel size $(2, 2)$; a convolutional layer with 64 channels and a kernel size of $(3, 3)$ with a ReLU activation function and a max pooling layer with kernel size $(2, 2)$; a fully connected layer with size $(1024, 64)$ and a ReLU activation layer; and another fully connected layer with size $(64, 10)$ CIFAR-10. The models are trained for 25 epochs, with the Adam algorithm, and a learning rate of 0.001 and a batch size of 128. When using the LAP Adam algorithm, a depression strength of $\delta_s = 1.0$, a strictness of $\lambda = 0.8$, and loss history length of 25 is used.

The results for these training algorithms can be seen in Figure 7 and 8. We compare the standard Adam optimiser training with the LAP Adam optimiser training, as well as a model that is trained on only clean labels (for context to the performance). For all but 2 combinations, it is clear that LAP training outperforms standard training, and reduces the performance gap to the model that is trained on clean labels. This suggests that the proposed algorithm outperforms standard training for a variety of noise levels and number of corrupt sources

We note that LAP outperforms the standard training on both the validation and test sets, suggesting that LAP is better at providing predictions on datasets with sources of mixed reliability, as well as those with reliable sources only.

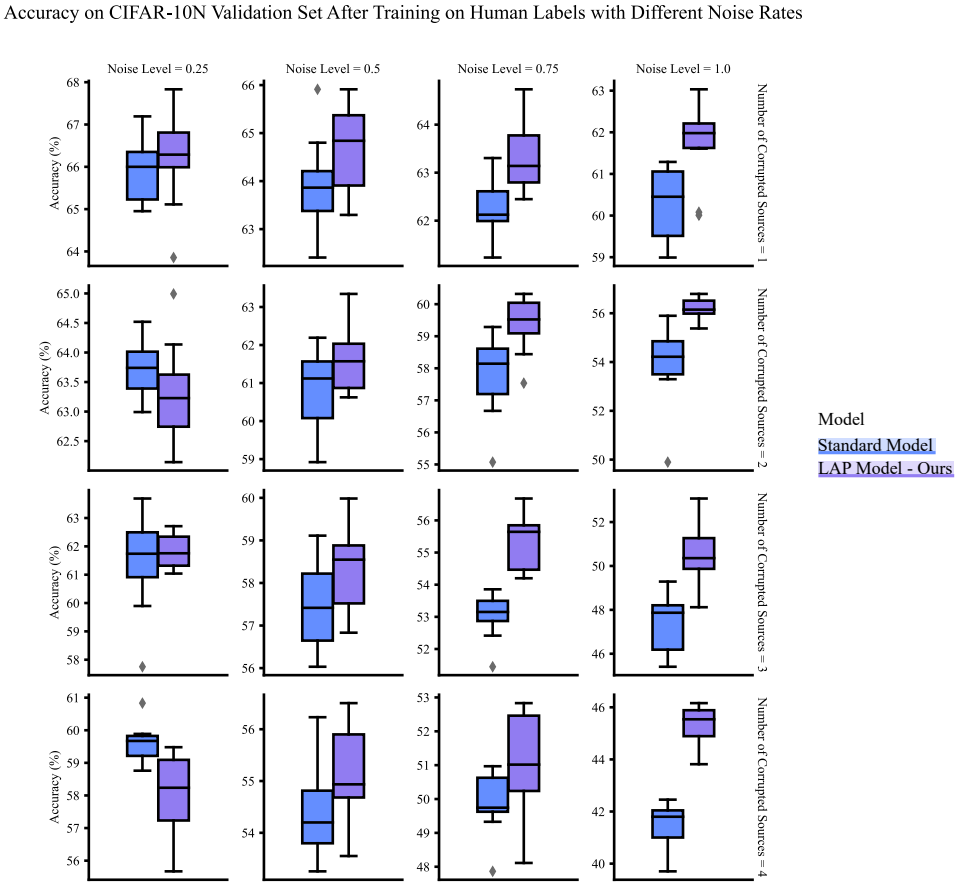


Figure 7: Accuracy score on the validation set of CIFAR-10N labels, after models were trained on varying amounts of the human labelled (with noisy labels) CIFAR-10N dataset. Results here are shown for varying amounts of unreliable sources and proportions of unreliable labels. For all experiments, there were 10 sources present in total.

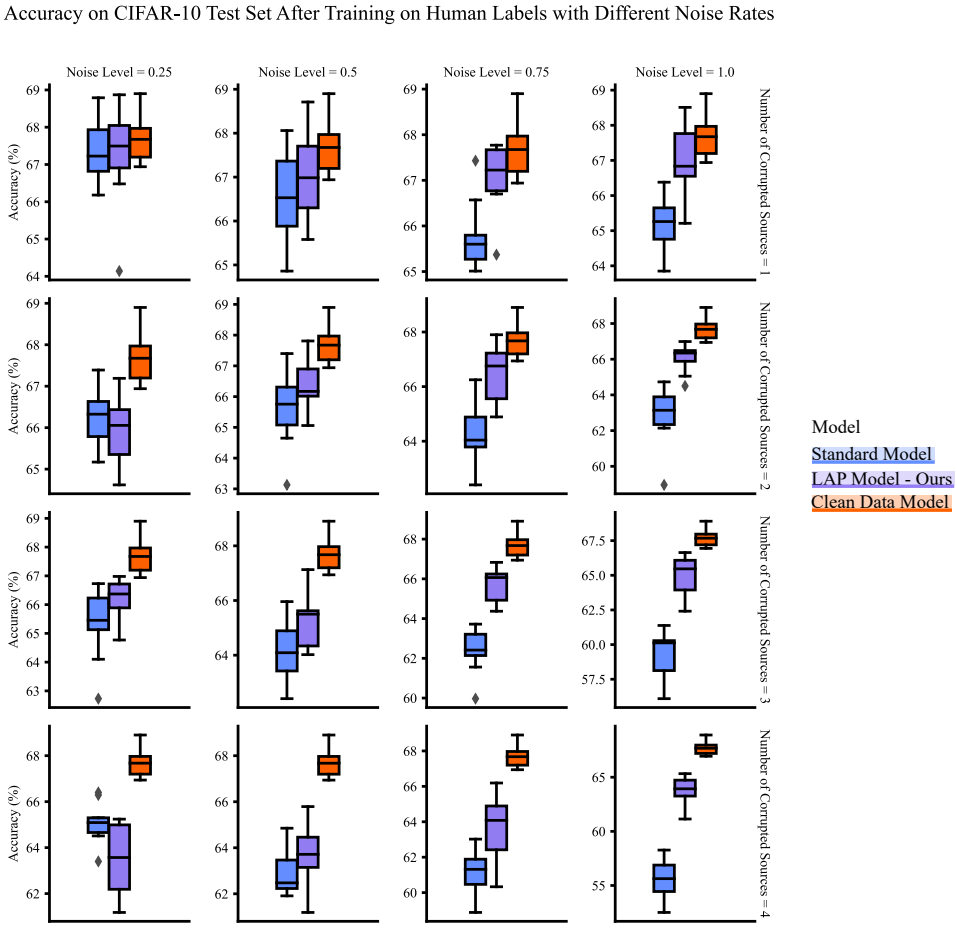


Figure 8: Accuracy score on clean CIFAR-10 labels (performance on the test set) after models were trained on varying amounts of the human labelled (with noisy labels) CIFAR-10N dataset. Results here are shown for varying amounts of unreliable sources and proportions of unreliable labels. For all experiments, there were 10 sources present in total. The 'Clean Data Model' is trained only on the true CIFAR-10 labels, and not on any of the noisy human labels, to give a context to the performance.

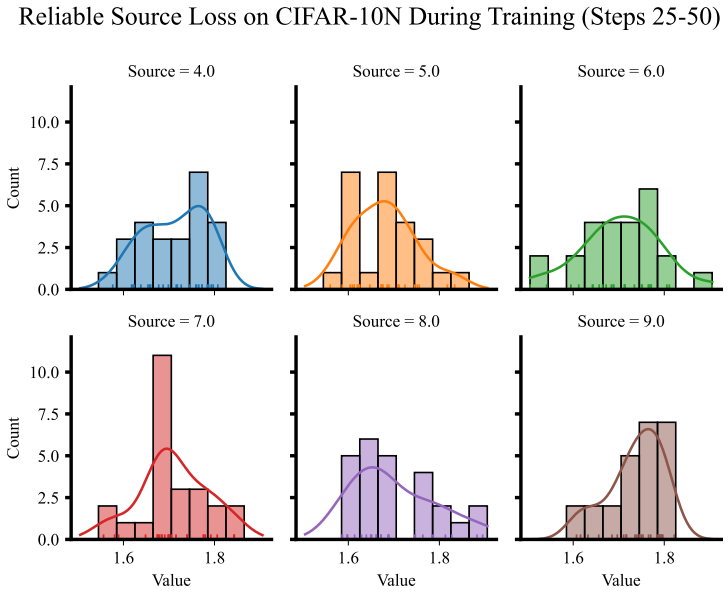
We also aimed to understand how LAP training would perform in a situation in which the model is significantly under-fitting to the data. To do this, we tested a model with reduced capacity (with 3 and 6 channels compared to the previous model of 32 and 64 channels in its convolutional layers), after being trained on the CIFAR-10N dataset containing 4 unreliable sources (with noise levels of 100%) and 10 total sources. The testing was performed on reliable data, from the CIFAR-10 dataset. The results of this are shown in Table 2, and suggest that LAP trained models also outperform the standard trained model in this context.

Table 2: Results in an under-fitting context.

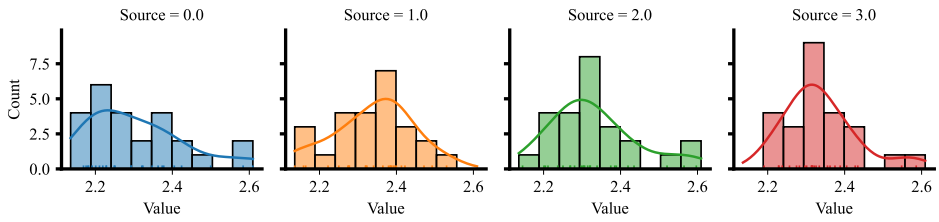
Model Name	**Accuracy $\pm$ Standard Deviation (%)**
Standard Model	42.85 ± 2.56
LAP Model (Ours)	$\mathbf{45.97 \pm 1.01}$

A.4 Loss Assumptions

One of our assumptions when designing the Loss Adapted Plasticity algorithm is that the loss values are distributed using a normal distribution over a given number of steps. We show in Figure 9 and 10 that the loss values approximate a normal distribution at higher step values, but less so at lower step values. This allows us to compare source loss values based on the distance between their mean in standard deviations when the step values are sufficiently large. This is why the hyper-parameter, hold-off, can be important - it allows for more control on when LAP is applied.

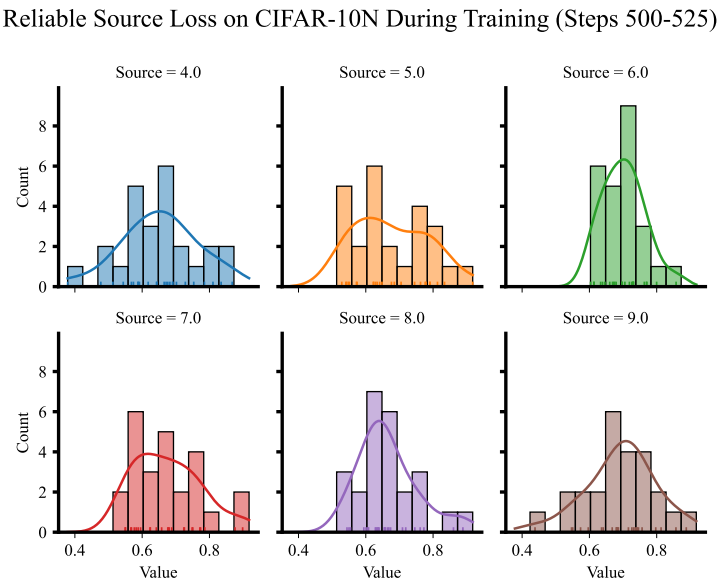


(a) Reliable Sources

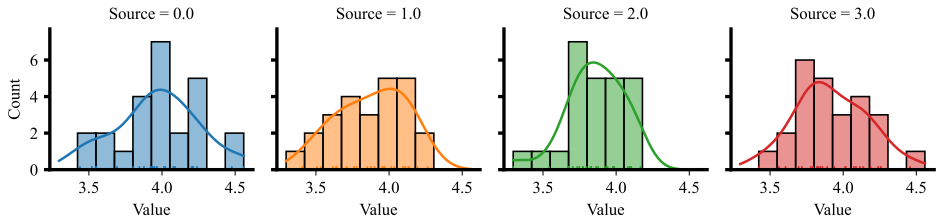


(b) Unreliable Sources

Figure 9: Both figures show the distribution of loss values over a period of 25 steps between step 25 and 50, when training a predictive model on the CIFAR-10N dataset. Here, a show the loss on batches from reliable sources only, binned as described in A.3.1. Similarly b shows the same information, except for the unreliable sources. Here, we have 4 unreliable sources, with noise levels of 100%.



(a) Reliable Sources



(b) Unreliable Sources

Figure 10: Both figures show the distribution of loss values over a period of 25 steps between step 500 and 525, when training a predictive model with LAP on the CIFAR-10N dataset. Here, a show the loss on batches from reliable sources only, binned as described in A.3.1. Similarly b shows the same information, except for the unreliable sources. Here, we have 4 unreliable sources, with noise levels of 100%.