# Sustaining learning under density shift: A big data volume to velocity reduction

**Anonymous Authors**[1]

## Abstract

The continuous surge in data volume which is often dealt using data orchestration and distributed processing approaches, abstracting away the machine learning challenges that exist at the algorithmic level. These either require in-memory presence of complete dataset, lead to synchronization barriers, and makes the memory cost a function of the size of the dataset; or have stale gradients issues. Commercial libraries have too pick up on algorithm-level solutions. This works takes a distributed density estimation angle to the problem where data are temporally or geographically distributed, and are too voluminous to be brought into contiguous memory. It processes data in batches, and allows a neural network to treat a batch as training data. The method accumulates knowledge about the data density via posterior probability absorption using Cramer Rao Lower Bound Optimization (CRLBO), and therefore the density estimate for the entire dataset is more robust to the non-iid distribution drift. The space cost is linear and no longer a function of the size of the complete, distributed dataset. The robustness of CRLBO to distribution drift is seen in improved accuracy on an average of $19\%$ in both online settings and a 20 batch split and $2\%$ in a 5 batch split where variance/drift across batches is high.

## 1. Introduction

Organizations continue to invest in big data strategy but successful transitions from pilot to large scale deployments remains a challenge. (Morales & Bifet, 2015) reports 49 success in the projects undertaken by Fortune 1000 business in last 5 years. (Kourtellis et al., 2019) and (Kourtellis et al., 2016) confirmed a transfer to production rate of 10 and 15 in separate studies. Big data are firstly defined by huge volumes. Harnessing this volume emanating from temporally or geographically distributed environments constitutes a considerable proportion of this conundrum. Currently two variations of distributed cluster computing using data parallelism namely cloud and on-premises based are used for high volume analysis. However, both have their demerits:

1. *Cloud-based services* These offer on-cloud compute resources enabling data parallelism but dataset residency on provider's cloud is mandatory [colab]. This has following disadvantages:
   - Regulatory compliance: Industry regulators restrict on-cloud customer/transactional data placement like financial datasets. This makes the cloud offering useless in these settings.
   - Data security and governance: The cloud model has yet to earn a secure data hosting platform status as data breach incidents occurrences increase. Namely, 100 million consumer data breach at CapitalOne on AWS, (Flitter & Weise, 2019), 80 million households data leak at unprotected server on Microsoft Cloud , (Baig, 2019), 6500 stores data compromised at Volusion on Google cloud, (Jowitt, 2019).
2. *On-premises frameworks* These implementations provide distributed machine learning capabilities but have following disadvantages:
   - Synchronization barriers and stale gradients: The prevalent implementations use synchronous or asynchronous variants of distributed mini batch gradient descent. The former suffers from synchronization barrier where the reduce process stalls until all worker nodes have retuned their updated models. While the later has stale gradient problem due to inherent asynchronization feature.
   - High memory cost: These require in memory dataset presence, large memory being an expensive resource increases the solution cost by the factor of the resource required.

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

- High native dependency: The implementations are completely dependant on the offering's native library algorithms. Nonexistence of a required algorithm may lead to stalling the process.

The literature for online learning in deep neural networks has struggled to produce techniques that take an approach at the algorithmic level, rather, depending on data management and orchestration techniques that these frameworks too employ.Dean et al. (2012) and Huang et al. (2019) This does seem to fulfil the needs of the data-led computing community, however none seem to really address the problem of the breaking down of the iid assumption. The specific case of that which presents itself in the case of density estimates being made on parts (by distribution) of the data is the problem of distribution drift Žliobaitė (2010) . This work addresses the machine learning problem of iid loss directly, and therefore offers a way for these frameworks to adopt such an approach. Designed on the concept of knowledge accumulation via posterior probability absorption; the proposed model requires only previous and current batch in-memory presence for parameter update. The model uses Laplace approximation to estimate the current data batch's class-conditional given that of the previous batch, allowing a summarized view of the data in the previous batch. This approach is a ligt-wight posterior-estimation alternative to both MCMC and variational inference. The choice of the method is made on the fact that unlike MCMC, its computation could be elegantly incorporated the existing backpropagation mechanism as a regularisation penalty.

**Contributions**
The major contributions of the paper are:

1. It addresses the problem of distribution drift using a computationally-tractable method for estimating the priors for a data subset's weight estimates given those from another, previous data subset. The estimates regularise the cross-entropy loss for the subset in training.

2. It breaks the synchronization barrier inherent in prevalent approaches as it requires presence of a small subset of data for accomplishing an iteration.

3. Memory cost is linear in the size of a data subset/batch and no longer a function of complete dataset.

## 2. Related works

Lifelong learning is usually studied under single-task or multi-task contexts, respectively as *online learning* and *incremental learning*. The works presented here fall within the = first category, although we have addressed the second in the latter half pf this section becuse the approch we develop bears some similarity with the concepts.The related works are organized as follows. Distributed machine learning libraries countering high volume are discussed. This is followed by incremental learning techniques. This arrangement follows from the idea behind the proposed technique which combats high volume using an incremental learning approach.

*MLib* a machine learning library by Meng et al. (2016) uses *Spark* as distributed in memory data processing engine, Zaharia et al. (2016). It uses Resilient Distributed Datasets for fast distributed in-memory processing, Zaharia et al. (2012). In-memory fault tolerance is provided using directed acyclic graphs, Karau et al. (2015). The library has distinctive extensibility feature where users can develop and extend algorithms, Venkataraman et al. (2016). Matrix computations are optimized by distributing these to the cluster while retaining vectors locally, Bosagh Zadeh et al. (2016). Classification algorithms include Naïve Bayes, Random Forest, SVM and GBM. Regression algorithms supported are linear regression and gradient boosting while gaussian mixture and k-means with variances are the supported clustering counterparts, Richter et al. (2015). Wide scale adoptions include Bioinformatics, Guo et al. (2018) and streaming, Armbrust et al. (2018).

*Apache Mahout* a distributed machine-learning library designed for recommendation, classification and clustering on very large datasets, Owen et al. (2011). It is built on top of *Hadoop file system* with current implementation *Samsara* supporting in-memory processing, Schelter et al. (2016). Wide scale implementations of recommendation systems with item and user collaborative filtering are available, Manu & Ramesh (2017). Works by Wu et al. (2018) and Beck et al. (2017) discuss movie and online news recommendation models. It has limited support for classification namely Random Forest and Naïve Bayes with implementations in healthcare Ko et al. (2014) and social media data analysis, Demirbaga & Jha (2018). Clustering support includes k-means with variances, Hefeeda et al. (2012) and spectral clustering , Esteves et al. (2011), Sharma et al. (2018).

*H2O* a machine learning platform with distributed in memory computation capabilities. It uses distributed Fork/Join technique for parallel in-memory computation. Excluding SVM, it supports all classification and regression algorithms as Spark. The platform derives its high usability distinction via a rich GUI with large number of languages support(Ghodsi, 2019). H2O provides strong integration with Spark using Sparkling Water where models share the same JVM container. Current implementations use Spark RDD for pre-processing and H2O frames for modelling

and analytics. This has in-memory data duplication issues as the later uses its own memory store, data structure and execution engine.

*Apache Samoa* is a distributed streaming machine learning framework, Morales & Bifet (2015). Designed for theoretically infinite streams with in-model concept drift capture, Kourtellis et al. (2019). A solution to non stationery and large volume datasets supporting Samza, S4 and Storm streaming engines (Kourtellis et al., 2019). The alogrithms include Vertical Hoeffding tree (VHT) for classification, Kourtellis et al. (2016), Clustream for clustering , Richter et al. (2015) and adaptive model rules for regression, Nasir et al. (2015). Instance wise horizontal and attribute wise vertical parallelism feature provided, Bifet & Morales (2014). Works by Kourtellis et al. (2015) used VHT with Bagging for Twitter streams real time analysis. Di Mauro & Di Sarno (2014) used VHT with Preqential feature for Skype internet traffic detection.

*Dask* a machine learning library designed for parallelizing numeric python eco system is used for large datasets that do not fit in memory, Rocklin (2015). It achieves scale out feature by building a task graph executed via dynamic scheduler executed across the cluster, Christ et al. (2018). The distributed algorithms supported are GLM, k-means and spectral clustering.

Catastrophic forgetting occurs in high velocity dynamics where the neural network forgets information of the first task after training the second task. *Incremental moment matching* by (Lee, 2017) counters catastrophic forgetting uses bayesian neural networks with mean and mode based moment matching. Average of two networks is calculated using mean while maximum of mixture of Gaussian posteriors is calculated using mode. The model uses drop transfer novelty for making the search space convex likeable. *PathNet* by (Fernando, 2017) achieves multi task learning by utilizing available large space of the network for distributive storing tasks information[w]. It associates a subset of modules in each layer to a specific task. *Learning without forgetting* exhibits multi task learning by combining knowledge distillation with fine tuning. It follows a regularization approach by optimizing current task data and pseudo training data of the old task.

The posterior of the previous task updates the new prior which learns the new posterior of the new task. *Hard interest mechanism* by (Serrà, 2018) mask the new task while keeping the previous masks intact. The technique using SGD jointly trains the main and ground network with fractional addition to its weights in a lightweight manner. *Crème* an online learning library designed for large datasets that have streaming dynamics or are large for the memory, Rocklin (2019). The model learns one observation at a time with interleaving training and prediction capability. The technique can scale incoming data via online mean and variance and extract features. Supported online gradient descent algorithms include SGD, Adam, RMS Prop, FTRL.

## 3. Proposed Method

This work extends the idea of catastrophic forgetting in neural networks to high voluminous and temporally distributed data streams. Works byLi & Hoiem (2018)minimize catastrophic forgetting in neural networks by using adaptable weight linkages. Their model attempts to build an analogy with brain functioning where past knowledge is retained by reducing elasticity of the relevant significant synapses. The proposed method in this work accumulates and retains knowledge on massive yet temporally disseminated datasets. The technique derives its novelty by requiring in-memory presence of only two data batches the previous and current for an effective parameter update iteration fig. 1 . This is in contrast to prevalent approaches which require complete dataset in memory presence for similar operation.

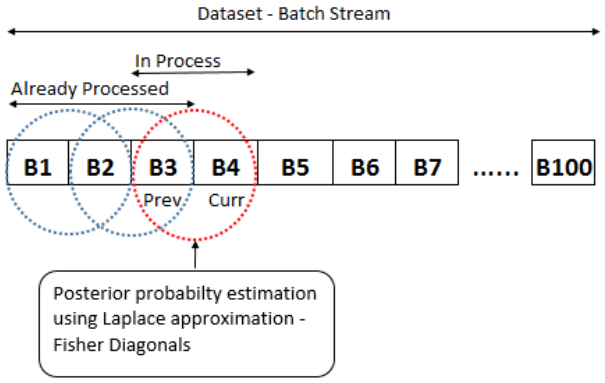


*Figure 1.* Online learning framework for big data challenges

The proposed model is expressed using Bayes rule in **Equation(1)**. This shows the posterior $P(\theta|D)$ calculated from $P(D_t|\theta)$ data likelihood of current and $P(\theta|D_{t-1})$ prior from previous batch. The arrangement presumes independence of the two batches. This advocates significance of absorbing knowledge from the previous batch mandating its true distribution to be approximated. This is done using Laplace approximation for Bayesian estimation MacKay (1992).

From Bayes' rule,

$$(\log(p(\boldsymbol{\theta}|\mathbf{D})) = \log(p(\mathbf{D}|\boldsymbol{\theta})) + \log(p(\boldsymbol{\theta})) - \log(p(\mathbf{D}))) \tag{1}$$

For a pair of data batches in a sequence,

$$logP(\theta|D) = logP(D_t|\theta) + logP(\theta|D_{t-1}) - logP(D_t)$$

*Laplace method* approximates intractable distribution in Bayesian statistics. It results in Gaussian approximation of the posterior using second-order Taylor expansion. The resulting bell-shaped curve has variance and likelihood inversely related. Low variance means a steep peak with high values of curvature, likelihood and information number and vice versa.

To elaborate, the previous batch $P(\theta|D_{t-1})$ is expressed as intractable true distribution P(x) in equation (2). Taylor expansion on f(x) results in normalized gaussian distribution with mean $x_0$ and variance R as shown in equation (3). The variance is the Fisher information matrix. Formulation of the parametric Laplace approximator now makes the distribution an absorber of past knowledge.

We look to optimize

$$\arg\max_{\theta}\{\ell(\theta) = \log(p(\theta|D))\} \tag{2}$$

The Laplace approxiation fits a Gaussian around a mode (the supremum above) to approximate the latter's magnitude **cite Edgeworth**. Pascano and Bengio determine that the following derivation may be performed using the first derivative intsead of computing the Hessian.

The intractability of the estimation draws from the computation of the partition function $Z$ below, which necessitates extensive marginalisaion in high dimensions.

$$P(x) = \frac{1}{Z}f(x) \tag{3}$$

The optimization objective, by Taylor approximation woule be:

$$f(x) \approx f(x_0)exp(-\frac{1}{2}R(x-x_0)^2) \tag{4}$$

$$\ell(\boldsymbol{\theta}) \approx \ell\left(\boldsymbol{\theta}^{\star}_{\mathcal{D}_{\sqcup-\infty}}\right) + \frac{1}{2}\left(\boldsymbol{\theta} - \boldsymbol{\theta}^{\star}_{\mathcal{D}_{\sqcup-\infty}}\right)^{\top} \left(\frac{\partial^2\ell(\boldsymbol{\theta})}{\partial^2\boldsymbol{\theta}}\bigg|_{\boldsymbol{\theta}^{\star}_{\mathcal{D}_{\sqcup-\infty}}}\right)\left(\boldsymbol{\theta} - \boldsymbol{\theta}^{\star}_{\mathcal{D}_{\sqcup-\infty}}\right) \tag{5}$$

Discounting higher order terms, we arrive at the closed form or the loss:

The loss function in (equation 4) consolidates current batch's cross entropy loss (first term) regularized by the Laplace approximator (second term) substituted from equation 3. The former encapsulates current and the latter ensures prior knowledge preservation by approximating its posterior as Gaussian.The equation variables are $C_{C}(\theta)$ current batch loss, $D_{P}$ previous data batch, $\overline{\theta}_{C,i}$ current and $\overline{\theta}_{P,i}$ previous mean value of the $i^{th}$ model parameter.

$$C(\theta) = C_{C}(\theta) + \sum_{i}\frac{\phi}{2}[\frac{\partial^2 L(\overline{\theta}_{P,i}|D_{P})}{\partial^2\overline{\theta}_{P,i}}][\overline{\theta}_{C,i} - \overline{\theta}_{P,i}]^2 \tag{6}$$

The parametric quantification of the distance from the mean between previous and current batch scaled by the Fisher variance produces the required delay learning effect. It decelerates learning on parameters significant to previous batch by constraining their value boundaries to be within archival range.

The algorithm in detail is stated as follows:

## 4. Experimental Setup

In this section details of dataset, model configurations, evaluation metrics and results of similarity experiments carried out under this experimental setup. The results of the experiments of the technique are discussed in Results and discussion chapter. Table 1 shows the datasets used later in the experiments.

### 4.1. Datasets

| Dataset | Access | Type | Total |
|---|---|---|---|
| AML | Proprietary | Tabular | 10,000,000 |
| Credit Default | Public | Tabular | 270 ,915 |
| Credit Fraud | Public | Tabular | 256 ,326 |
| Paysim | Public | Tabular | 6,353,307 |
| CIFAR10 | Public | Image | 60,000 |

*Table 1.* Datasets

*AML dataset:* Anti-money laundering dataset comprises of SAR (suspicious activity report) and Non-SAR transactions of the individuals in a financial institution accumulated for the period of 1 year.

*Home Credit Default dataset:* Home credit default contain the transactional and repayment history of the customers which are provided loan with no credit history.

*Credit card fraud dataset:* This dataset contain the fraudulent and non-fraudulent credit card transactions of all the

**Algorithm 1** Weights preservation using CRLBO

**Initialization:** Split dataset into $\beta$ batches where $\beta \in \{5, 20, 40\}$
batch = b $\in \beta$
$t \leftarrow 1$ = current batch
$t-1$ = previous batch
**Training:**
**if** t = 1 **then**
1. Initialize classifier with random weights
2. Weights optimization - for $b_t$ are learned using CrossEntropyLoss
//Distribution learnt using cross entropy which constitutes the term 1 of the proposed model
**else if** t > 1 **then**
**while** t $\leq \beta$ **do**
1. Initialize classifier with weights of $b_{t-1}$
2. Weight update - for $b_t$ are learned using CrossEntropyLoss
3. Weight update - for $b_{t-1}$ are adjusted using Cramer Rao bound
3.1 Mean $\leftarrow \theta_t - \theta_{t-1}$
3.2 Fisher Variance $\leftarrow E\left[\frac{\partial L(\theta|X_{t-1})}{\partial \theta}\right]$
3.3 Loss $L(\theta) \leftarrow L_B(\theta) + \sum_i E\left[\frac{\partial L(\theta|X_{t_i-1})}{\partial \theta}\right] \cdot (\theta_{t_i} - \theta_{t_i-1})^2$
//Weights adjusted using Cramer Rao Lower bound which constitutes the term 2 of the proposed model
t++
**end while**
**end if**

customers for a period of one month which are European credit card holders.

*Paysim:* The dataset comprises of simulated mobile money transactions based real transactions extracted from one month of financial logs which consists of around 6 million transactions out of which 8312 transactions are labeled as fraud.

*CIFAR:* This dataset consists of 32x32 colour images in 10 classes, with 6000 images per class. There are 50000 training images and 10000 test images.

### 4.2. Model Configurations

Model wise configurations used in the experiments are listed in Table 2.

*Fisher multiplier: The Fisher is scaled by this number to form the Cramer bound.

| Model Configurations | |
|---|---|
| Optimizer \| Activation | SGD \| Relu |
| Loss function | CrossEntropy Loss |
| Epochs \| Batch size | 20 \| 100 |
| Learning rate \| Momentum | 0.001 \| 0.1 |
| No. of hidden layers \| *Fisher multiplier | 2 \| 400 |

*Table 2.* Model Configurations

### 4.3. Evaluation metrics

The evaluation metrics used in this experimental setup is accuracy which is measured as correctly predicted to the total predictions in the dataset. (Kirkpatrick et al., 2017) suggest the use of $L2$ regularisation and the noise in the SGD training as alternative processes to carry estimates across batches, and therefore these form our benchmarks.

### 4.4. Experimental Nomenclature

The experiments are categorized into 3 types: Cramer Rao Lower Bound, No Penalty and L2 penalty based on multiple batch settings e.g., 5 and 20 batches for each data-set. Cramer Rao Lower Bound Optimization is one of artifacts of the proposed model in this study which ensures the information retention of the previous batches when training on the current batch, however the competing techniques of optimization from the literature were also compared namely SGD in which gradient steps according to the current batch only and will minimize the loss of the current batch but destroy information of the previous batch and L2 penalty which is too severe that it tries to retain the information of previous batch only at the expense of not learning current batch data distribution. The experiments are carried out using multiple batch sizes as to validate the technique on different variances among the batches. Also in case of online setting, the technique is validated in comparison of the prevailing techniques.

### 4.5. Investigating for similarity among batches

To further investigate the results of the technique, an analysis of the percentage difference in the two means was performed for finding the similarity among batches of a dataset.

The mean of a batch was calculated after averaging the mean of each column in the batch set. Following are the mean percentage difference between batches which will be referred in the results section below.

The percentage difference in the two means shows that the batches appears quite different in distribution and inter-batch variance across batches was observed quite high with the net difference of 184.32% in all the batches of Credit Default dataset and vice versa in all the rest of the datasets.

| Dataset | % Diff in means |
|---|---|
| AML | 0.75 |
| Credit Default | 184.32 |
| Credit Fraud | 2.05 |
| Paysim | 4.07 |

*Table 3.* Datasets

## 5. Results and Discussion

This section is organized in further sub-sections based on the batch size of the dataset as the experiments were carried out under different settings discussed in experimental setup. Following are the results of our technique with competing state of the art techniques in online mode, 20 batches and 5 batches on each data-set which are formulated as:

| | | |
|---|---|---|
| **AML** | L2 | 0.9986 |
| | CRLBO | 0.9985 |
| **Credit Default** | L2 | 0.9192 |
| | CRLBO | 0.9194 |
| **Paysim** | L2 | 0.9987 |
| | CRLBO | 0.9986 |
| **CIFAR10** | L2 | 0.332 |
| | CRLBO | 0.526 |

*Table 4.* Online Results

| **L2** | Testing | | | |
|---|---|---|---|---|
| *Training* | *Batch 1* | *Batch 10* | *Batch 15* | *Batch 20* |
| Batch 1 | 0.7043 | | | |
| Batch 10 | 0.8199 | 0.8098 | | |
| Batch 15 | 0.8047 | 0.8206 | 0.7943 | |
| Batch 20 | 0.7413 | 0.7502 | 0.7468 | 0.7370 |
| **SGD** | | | | |
| Batch 1 | 0.7755 | | | |
| Batch 10 | 0.8728 | 0.8896 | | |
| Batch 15 | 0.2823 | 0.2737 | 0.2822 | |
| Batch 20 | 0.6766 | 0.6874 | 0.6706 | 0.6966 |
| **CRLBO** | | | | |
| Batch 1 | 0.9067 | | | |
| Batch 10 | 0.9032 | 0.9114 | | |
| Batch 15 | 0.9035 | 0.9076 | 0.9078 | |
| Batch 20 | 0.9074 | 0.9114 | 0.9109 | 0.9119 |

*Table 5.* Credit Default Results (20 Batches)

As shown in Table 5, CRLBO outperforms other techniques on a tabular dataset in 20 batch setting by 18%. This can be attributed to the high inter-batch variance present in the dataset. Thus it can be inferred that CRLBO retains maximum information on tabular datasets in high variant settings.

As shown in Table 4 CRLBO outperforms other techniques

| **L2** | Testing | | | |
|---|---|---|---|---|
| *Training* | *Batch 1* | *Batch 10* | *Batch 15* | *Batch 20* |
| Batch 1 | 0.9978 | | | |
| Batch 10 | 0.9930 | 0.9985 | | |
| Batch 15 | 0.9987 | 0.9984 | 0.9988 | |
| Batch 20 | 0.9893 | 0.9984 | 0.9972 | 0.9985 |
| **SGD** | | | | |
| Batch 1 | 0.9987 | | | |
| Batch 10 | 0.9987 | 0.9987 | | |
| Batch 15 | 0.9969 | 0.9919 | 0.9984 | |
| Batch 20 | 0.9987 | 0.9987 | 0.9987 | 0.9987 |
| **CRLBO** | | | | |
| Batch 1 | 0.9989 | | | |
| Batch 10 | 0.9802 | 0.9982 | | |
| Batch 15 | 0.9897 | 0.9981 | 0.9985 | |
| Batch 20 | 0.9923 | 0.9982 | 0.9988 | 0.9990 |

*Table 6.* Paysim Results (20 Batches)

| **L2** | Testing | | | |
|---|---|---|---|---|
| *Training* | *Batch 1* | *Batch 10* | *Batch 15* | *Batch 20* |
| Batch 1 | 0.1000 | | | |
| Batch 10 | 0.1000 | 0.1000 | | |
| Batch 15 | 0.1000 | 0.1000 | 0.0900 | |
| Batch 20 | 0.1000 | 0.1000 | 0.0900 | 0.0900 |
| **SGD** | | | | |
| Batch 1 | 0.1000 | | | |
| Batch 10 | 0.1000 | 0.1000 | | |
| Batch 15 | 0.1000 | 0.1000 | 0.1000 | |
| Batch 20 | 0.1000 | 0.1000 | 0.1000 | 0.1000 |
| **CRLBO** | | | | |
| Batch 1 | 0.0900 | | | |
| Batch 10 | 0.0900 | 0.0900 | | |
| Batch 15 | 0.0900 | 0.0900 | | |
| Batch 20 | 0.0900 | 0.0900 | 0.0800 | 0.1000 |

*Table 7.* CIFAR10 Results (20 Batches)

by 19% on high dimensional images in online settings. Thus it can be inferred that CRLBO retains maximum information in high dimensional settings.

The inference obtained from this investigation is that in high volume settings, if the distribution of the data is similar then any state of the art technique could achieve comparable generalization however in case of varying distribution and high variance across batches, CRLBO achieves best generalization by adjusting the weights in a way that it also retains the information of the previous batches which makes this technique work better over state of the art prevailing techniques.

In online settings where the data with high velocity and high variance is exposed to the model for training, the proposed

| **L2** | Testing | | | |
|---|---|---|---|---|
| *Training* | *Batch 1* | *Batch 10* | *Batch 15* | *Batch 20* |
| Batch 1 | 0.9985 | | | |
| Batch 10 | 0.9985 | 0.9986 | | |
| Batch 15 | 0.9975 | 0.9975 | 0.9972 | |
| Batch 20 | 0.9982 | 0.9981 | 0.9980 | 0.9981 |
| **SGD** | | | | |
| Batch 1 | 0.9980 | | | |
| Batch 10 | 0.9986 | 0.9986 | | |
| Batch 15 | 0.9985 | 0.9985 | 0.9985 | |
| Batch 20 | 0.9986 | 0.9986 | 0.9985 | 0.9986 |
| **CRLBO** | | | | |
| Batch 1 | 0.9983 | | | |
| Batch 10 | 0.9984 | 0.9984 | | |
| Batch 15 | 0.9982 | 0.9979 | 0.9978 | |
| Batch 20 | 0.9985 | 0.9984 | 0.9984 | 0.9985 |

*Table 8.* AML Results (20 Batches)

| L2 - 0.9985 | Testing | | | | |
|---|---|---|---|---|---|
| Training | Batch 1 | Batch 2 | Batch 3 | Batch 4 | Batch 5 |
| Batch 5 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9987 |
| SGD - 0.9983 Batch 5 | 0.9986 | 0.9986 | 0.9986 | 0.9986 | 0.9987 |
| CRLBO - 0.998 Batch 5 | 0.998 | 0.9979 | 0.9981 | 0.998 | 0.9981 |

*Table 9.* AML Dataset Results (5 Batches)

| L2 - 0.9131 | Testing | | | | |
|---|---|---|---|---|---|
| Training | Batch 1 | Batch 2 | Batch 3 | Batch 4 | Batch 5 |
| Batch 5 | 0.8996 | 0.9026 | 0.8993 | 0.9017 | 0.898 |
| SGD - 0.9103 Batch 5 | 0.9113 | 0.91 | 0.9096 | 0.9114 | 0.9093 |
| CRLBO - 0.9172 Batch 5 | 0.9173 | 0.9167 | 0.9152 | 0.9176 | 0.9216 |

*Table 10.* Credit Default Dataset Results (5 Batches)

technique also stand out from the state of the art prevailing techniques but in case of low variance and high similarity between the batches, the distribution to learn is not as challenging hence similar results for all the techniques. A comparison of the results is shown in the Table.4.

| L2 - 0.9987 | Testing | | | | |
|---|---|---|---|---|---|
| Training | Batch 1 | Batch 2 | Batch 3 | Batch 4 | Batch 5 |
| Batch 5 | 0.9987 | 0.9987 | 0.9987 | 0.9987 | 0.9987 |
| SGD - 0.999 Batch 5 | 0.9988 | 0.9988 | 0.9988 | 0.9988 | 0.9988 |
| CRLBO - 0.999 Batch 5 | 0.999 | 0.999 | 0.9991 | 0.9988 | 0.9991 |

*Table 11.* Paysim Dataset Results (5 Batches)

## 6. Conclusion

We study a distribution density estimation angle for sustaining learning where data are temporally or geographically distributed or high volume constraints their in-memory presence. In summary:

1. The proposed model accumulates knowledge about the data density via posterior probability absorption using Cramer Rao Lower Bound Optimization.
2. The model scales with high volume as the memory required is linear and merely a function of the batch size rather than the complete dataset.
3. The model is adaptable in high velocity as information is retained in online settings.
4. The model achieves an increase in 19% accuracy with competing techniques in online and multi-batch settings.
5. The model is equally effective on high dimensional image based or tabular data.

This work opens possibilities in working with online settings where neither complete dataset presence is required nor distribution drift stales the model as knowledge is sustained with a small subset of data in online and multi batch settings.

## References


Armbrust, M., Das, T., Torres, J., Yavuz, B., Zhu, S., Xin, R., Ghodsi, A., Stoica, I., and Zaharia, M. Structured streaming: A declarative api for real-time applications in apache spark. In *Proceedings of the 2018 International Conference on Management of Data*, pp. 601–613, 2018.

Baig, E. C. Massive data breach exposes ages, addresses, income on 80 million u.s. families. *USA Today*, 2019.

Beck, P. D., Blaser, M., Michalke, A., and Lommatzsch, A. A system for online news recommendations in real-time with apache mahout. In *CLEF (Working Notes)*, 2017.

Bifet, A. and Morales, G. D. F. Big data stream learning with samoa. In *2014 IEEE International Conference on Data Mining Workshop*, pp. 1199–1202. IEEE, 2014.

Bosagh Zadeh, R., Meng, X., Ulanov, A., Yavuz, B., Pu, L., Venkataraman, S., Sparks, E., Staple, A., and Zaharia, M. Matrix computations and optimization in apache spark. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 31–38, 2016.

Christ, M., Braun, N., Neuffer, J., and Kempa-Liehr, A. W. Time series feature extraction on basis of scalable hypothesis tests (tsfresh–a python package). *Neurocomputing*, 307:72–77, 2018.

Dean, J., Corrado, G., Monga, R., Chen, K., Devin, M., Mao, M., Ranzato, M., Senior, A., Tucker, P., Yang, K., et al. Large scale distributed deep networks. In *Advances in neural information processing systems*, pp. 1223–1231, 2012.

Demirbaga, U. and Jha, D. N. Social media data analysis using mapreduce programming model and training a tweet classifier using apache mahout. In *2018 IEEE 8th International Symposium on Cloud and Service Computing (SC2)*, pp. 116–121. IEEE, 2018.

Di Mauro, M. and Di Sarno, C. A framework for internet data real-time processing: A machine-learning approach. In *2014 International Carnahan Conference on Security Technology (ICCST)*, pp. 1–6. IEEE, 2014.

Esteves, R. M., Pais, R., and Rong, C. K-means clustering in the cloud–a mahout test. In *2011 IEEE Workshops of International Conference on Advanced Information Networking and Applications*, pp. 514–519. IEEE, 2011.

Fernando, C., B. D. B. C. Z. Y. H. D. R. A. . W. D. Pathnet: Evolution channels gradient descent in super neural networks. *arXiv*, 2017.

Flitter, E. and Weise, K. Capital one data breach compromises data of over 100 million. *The NewYork Times*, 2019.

Ghodsi, A. Spark support. 2019. URL https://databricks.com/support.

Guo, R., Zhao, Y., Zou, Q., Fang, X., and Peng, S. Bioinformatics applications on apache spark. *GigaScience*, 7(8): giy098, 2018.

Hefeeda, M., Gao, F., and Abd-Almageed, W. Distributed approximate spectral clustering for large-scale datasets. In *Proceedings of the 21st international symposium on High-Performance Parallel and Distributed Computing*, pp. 223–234, 2012.

Huang, Y., Cheng, Y., Bapna, A., Firat, O., Chen, D., Chen, M., Lee, H., Ngiam, J., Le, Q. V., Wu, Y., et al. Gpipe: Efficient training of giant neural networks using pipeline parallelism. In *Advances in Neural Information Processing Systems*, pp. 103–112, 2019.

Jowitt, T. Hackers compromise volusion, steal card details from 6500 websites. *Silicon.co.uk*, 2019.

Karau, H., Konwinski, A., Wendell, P., and Zaharia, M. *Learning spark: lightning-fast big data analysis*. " O'Reilly Media, Inc.", 2015.

Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

Ko, K. D., El-Ghazawi, T., Kim, D., and Morizono, H. Predicting the severity of motor neuron disease progression using electronic health record data with a cloud computing big data approach. In *2014 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology*, pp. 1–6. IEEE, 2014.

Kourtellis, N., Morales, G. D. F., and Bonchi, F. Scalable online betweenness centrality in evolving graphs. *IEEE Transactions on Knowledge and Data Engineering*, 27 (9):2494–2506, 2015.

Kourtellis, N., Morales, G. D. F., Bifet, A., and Murdopo, A. Vht: Vertical hoeffding tree. In *2016 ieee international conference on big data (big data)*, pp. 915–922. IEEE, 2016.

Kourtellis, N., Morales, G. D. F., and Bifet, A. Large-scale learning from data streams with apache samoa. In *Learning from Data Streams in Evolving Environments*, pp. 177–207. Springer, 2019.

Lee, S. W., K. J. H. J. J. H. J. W. . Z. B. T. Overcoming catastrophic forgetting by incremental moment matching. In *Advances in neural information processing systems*, pp. 4562–4662, 2017.

Li, Z. and Hoiem, D. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2018.

MacKay, D. J. Bayesian interpolation. *Neural computation*, 4(3):415–447, 1992.

Manu, M. and Ramesh, B. Single-criteria collaborative filter implementation using apache mahout in big data. *International Journal of Computer Sciences and Engineering Open Access*, 5(1):7–13, 2017.

Meng, X., Bradley, J., Yavuz, B., Sparks, E., Venkataraman, S., Liu, D., Freeman, J., Tsai, D., Amde, M., Owen, S., et al. Mllib: Machine learning in apache spark. *The Journal of Machine Learning Research*, 17(1):1235–1241, 2016.

Morales, G. D. F. and Bifet, A. Samoa: scalable advanced massive online analysis. *Journal of Machine Learning Research*, 16(1):149–153, 2015.

Nasir, M. A. U., Morales, G. D. F., Garcia-Soriano, D., Kourtellis, N., and Serafini, M. The power of both choices: Practical load balancing for distributed stream processing engines. In *2015 IEEE 31st International Conference on Data Engineering*, pp. 137–148. IEEE, 2015.

Owen, S., Anil, R., Dunning, T., and Friedman, E. Mahout in action: Manning shelter island. 2011.

Richter, A. N., Khoshgoftaar, T. M., Landset, S., and Hasanin, T. A multi-dimensional comparison of toolkits for machine learning with big data. In *2015 IEEE International Conference on Information Reuse and Integration*, pp. 1–8. IEEE, 2015.

Rocklin, M. Dask: Parallel computation with blocked algorithms and task scheduling. In *Proceedings of the 14th python in science conference*, number 130-136. Citeseer, 2015.

Rocklin, M. Online machine learning with crème, 2019. URL https://github.com/creme-ml/creme.

Schelter, S., Palumbo, A., Quinn, S., Marthi, S., and Musselman, A. Samsara: Declarative machine learning on distributed dataflow systems. In *NIPS Workshop MLSystems*, 2016.

Serrà, J., S. D. M. M. . K. A. Overcoming catastrophic forgetting with hard attention to the task. *arXiv*, 2018.

Sharma, I., Tiwari, R., Rana, H. S., and Anand, A. Analysis of mahout big data clustering algorithms. In *Intelligent Communication, Control and Devices*, pp. 999–1008. Springer, 2018.

Venkataraman, S., Yang, Z., Liu, D., Liang, E., Falaki, H., Meng, X., Xin, R., Ghodsi, A., Franklin, M., Stoica, I., et al. Sparkr: Scaling r programs with spark. In *Proceedings of the 2016 International Conference on Management of Data*, pp. 1099–1104, 2016.

Wu, C.-S. M., Garg, D., and Bhandary, U. Movie recommendation system using collaborative filtering. In *2018 IEEE 9th International Conference on Software Engineering and Service Science (ICSESS)*, pp. 11–15. IEEE, 2018.

Zaharia, M., Chowdhury, M., Das, T., Dave, A., Ma, J., McCauly, M., Franklin, M. J., Shenker, S., and Stoica, I. Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing. In *Presented as part of the 9th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 12)*, pp. 15–28, 2012.

Zaharia, M., Xin, R. S., Wendell, P., Das, T., Armbrust, M., Dave, A., Meng, X., Rosen, J., Venkataraman, S., Franklin, M. J., et al. Apache spark: a unified engine for big data processing. *Communications of the ACM*, 59 (11):56–65, 2016.

Žliobaitė, I. Learning under concept drift: an overview. *arXiv preprint arXiv:1010.4784*, 2010.