NORTHWESTERN UNIVERSITY

Topics in Machine Learning Optimization

A DISSERTATION

SUBMITTED TO THE GRADUATE SCHOOL

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

for the degree

DOCTOR OF PHILOSOPHY

Field of Engineering Science and Applied Mathematics

By

Biyi Fang

EVANSTON, ILLINOIS

December 2021

Abstract

Topics in Machine Learning Optimization

Biyi Fang

Recently, machine learning and deep learning, which have made many theoretical and empirical breakthroughs and is widely applied in various fields, attract a great number of researchers and practitioners. They have become one of the most popular research directions and plays a significant role in many fields, such as machine translation, speech recognition, image recognition, recommendation system, etc. Optimization, as one of the core components, attracts much attention of researchers. The essence of most machine learning and deep learning algorithms is to build an optimization model and learn the parameters in the objective function from the given data. With the exponential growth of data amount and the increase of model complexity, optimization methods in machine learning face more and more challenges. In the era of immense data, the effectiveness and efficiency of the numerical optimization algorithms dramatically influence the popularization and application of the machine learning and deep learning models. In this study, we propose a few effective optimization algorithms for different optimization problems, which have improved the performance and efficiency of machine learning and deep learning methods. This dissertation consists of four chapters, 1) Stochastic

Large-scale Machine Learning Algorithms with Distributed Features and Observations, 2) Convergence Analyses of Online ADAM, 3) Topic Analysis for Text with Side Data and 4) Tricks and Plugins to GBM on Images and Sequences.

In the first chapter, We propose a general stochastic offline algorithm where observations, features, and gradient components can be sampled in a double distributed setting, i.e., with both features and observations distributed. Moreover, very technical analyses establish convergence properties of the algorithm under different conditions on the learning rate (diminishing to zero or constant). Furthermore, computational experiments in Spark demonstrate a superior performance of our algorithm versus a benchmark in early iterations of the algorithm, which is due to the stochastic components of the algorithm.

In the second chapter, we explore how to apply optimization algorithms with fixed learning rate in online learning. Online learning is an appealing learning paradigm, which is of great interest in practice due to the recent emergence of large scale applications. Standard online learning assumes a finite number of samples while in practice data is streamed infinitely. In such a setting gradient descent with a diminishing learning rate does not work. In this chapter, we first introduce regret with rolling window, a performance metric, which measures the performance of an algorithm on every fixed number of contiguous samples. Meanwhile, we propose a family of algorithms with a constant or adaptive learning rate and provide very technical analyses establishing regret bound properties. We cover the convex setting showing the regret of the order of the square root of the size of the window in the constant and dynamic learning rate scenarios. Our proof is applicable also to the standard online setting where we provide analyses of the same regret order (the previous proofs have flaws). We also study a two layer neural network setting with reLU activation. In this case we establish that if initial weights are close to

a stationary point, the same regret bound is attainable. We conduct computational experiments demonstrating a superior performance of the proposed algorithms.

In the third paper, we employ text with side data to tackle the limitations like cold-start and non-transparency in latent factor models (e.g., matrix factorization). We introduce a hybrid generative probabilistic model that combines a neural network with a latent topic model, which is a four-level hierarchical Bayesian model. In the model, each document is modeled as a finite mixture over an underlying set of topics and each topic is modeled as an infinite mixture over an underlying set of topic probabilities. Furthermore, each topic probability is modeled as a finite mixture over side data. In the context of text, the neural network provides an overview distribution about side data for the corresponding text, which is the prior distribution in LDA to help perform topic grouping. The approach is evaluated on several different datasets, where the model is shown to outperform standard LDA and Dirichlet-multinomial regression (DMR) in terms of topic grouping, model perplexity, classification and comment generation.

In the forth paper, we propose a new algorithm for boosting Deep Convolutional Neural Networks (BoostCNN) to combine the merits of dynamic feature selection and BoostCNN, and another new family of algorithms combining boosting and transformers. To learn these new models, we introduce subgrid selection and importance sampling strategies and propose a set of algorithms to incorporate boosting weights into a deep learning architecture based on a least squares objective function. These algorithms not only reduce the required manual effort for finding an appropriate network architecture but also result in superior performance and lower running time. Experiments show that the proposed methods outperform benchmarks on several fine-grained classification tasks.

The systematic retrospect and summary of the optimization methods from the perspective of machine learning are of great significance, which can offer guidance for both developments of optimization and machine learning research.

Acknowledgements

I would like to express my greatest gratitude to my advisor Professor Diego Klabjan for his guidance and mentorship over the past four years. Professor Klabjan has been tremendously supportive and patient. He has provided me with insightful scientific advice and useful suggestions for career development. He also has a great sense of humor; his jokes always add vibrant color to my busy research life. I am extremely grateful to have such a great professor.

I would like to thank my thesis committee members Professor David Chopp and Professor Alvin Bayliss, for their insightful suggestions and encouragement. I enjoyed Professor Chopp's and Professor Bayliss's lectures a lot, particularly their algorithm classes, which inspired me to develop my research interest in optimization algorithms.

My gratitude also goes to the students, faculty, and staff at the Northwestern Engineering Science and Applied Mathematics Department for their kind support. I would also like to thank Dr. Jean Utke from AllState company and Kripa Rajshekhar from Metonymize company for insightful discussions and feedback about my research projects.

Finally, I would like to thank my parents, my fiancée, and my fiancée's family for their unconditional love, sacrifice, and support. I am forever in their debt.

Table of Contents

List of Tables

List of Figures

CHAPTER 1

Stochastic Large-scale Machine Learning Algorithms with Distributed Features and Observations

1.1. Introduction

As technology advances, collecting and analyzing large-scale and real time data has become widely used in a variety of fields. Large-scale machine learning can not only present a useful summary of a dataset but it can also make predictions. In the era of big data, large scale datasets have become more accessible. "Large" usually refers to both the number of observations and high feature dimension. For example, an English Wikipedia dataset can have 11 million documents (observations) and over several hundred of thousands unique word types (features). This demands a sophisticated large-scale machine learning system able to take advantages of all available information from such datasets and not only random samples. On the other hand, as large scale data is becoming more accessible, storing the whole dataset on a single server is often impossible due to the inadequate disk and memory capacities. Consequently, it is considerable to store and analyze them distributively. Often the data collection process by design distributes observations and features. There is a large amount of literature dealing with optimization problems subject to a dataset with either distributed observations or distributed features. Nevertheless, very limited contribution has been made to the case where both the observations and features are in a distributed environment.

In this paper, we propose an algorithm, namely SODDA (StOchastic Doubly Distributed Algorithm), designed for a doubly distributed dataset and inspired by the work (Harikandeh et al., 2015) and (Nathan and Klabjan, 2017). The algorithm is aimed to solve a series of optimization problems which can be formulated as the minimization of a finite sum of convex functions plus a convex regularization term if necessary. SODDA is a primal method building on the previous RAndom Distributed Stochastic Algorithm (RADiSA) (Nathan and Klabjan, 2017). SODDA first further splits the partitions (a partition is a set of features and observations stored locally) with respect to features into sub-partitions with no overlap; then in each iteration, randomly chooses sub-partitions associated with different blocks of features; lastly, similar to stochastic gradient descent (SGD), updates in parallel each sub-block of the current local solution by using observations from the randomly selected sub-partition of local observations and the local sub-block of features, coupled with the Stochastic Variance-Reduced Gradient (SVRG). One generalization of SVRG utilized in SODDA is that SODDA does not require a full solution update; instead, it allows each sub-block of the current local solution to be updated individually and assembled at the end of each iteration. Although, we might amplify the error by approximately computing the gradient, SODDA reduces the communication cost significantly. Another technique aiming to cut down the communication cost is estimating the full gradient needed as part of the SVRG component, which is a big distinction between SODDA and RADiSA. RADiSA requires the full gradient in each outer iteration, which is computationally demanding, especially when the solution is far from an optimal solution. SODDA has three stochastic components: the first two are that it randomly selects blocks of local features and subsets of local observations to execute the estimated gradient, and the third component that randomly chooses further sub-blocks of local features to record the approximated gradient,

which contributes to a reduction of the number of gradient coordinate computations required in early iterations. In other words, only random coordinates of the gradient are computed.

In this paper, we not only propose a more computationally efficient method, SODDA, when compared to RADiSA (Nathan and Klabjan, 2017), but also present a complete technical proof of convergence. For a smooth and strongly convex function, we prove that SODDA enjoys at least a sublinear convergence rate and a linear convergence rate for a diminishing learning rate and a constant learning rate, respectively. Furthermore, we prove that SODDA iterates converge to an optimal solution when using a constant learning rate selected from a certain interval. Moreover, the convergence property of RADiSA, which is not provided in (Nathan and Klabjan, 2017), is implied directly from SODDA. In summary, we make the following five contributions.

- We provide a better scalable stochastic doubly distributed method, i.e. SODDA, for doubly distributed datasets. This algorithm does not require the calculation of a full gradient, thus it is a less computationally intensive methodology for doubly distributed setting problems. We provide a proof of a geometric (without feature and sample sampling) and sublinear (with feature and sample sampling) convergence result for smooth and strongly-convex loss functions when using a sequence of decreasing learning rates.
- We show that SODDA iterates converge with linear rate to a neighborhood of an optimal solution when using an arbitrary constant learning rate.
- We further argue that SODDA iterates converge to an optimal solution in the strongly-convex case when using any constant learning rate in a specified interval.
- We present numerical results showing that SODDA outperforms RADiSA-avg, which is the best doubly distributed optimization algorithm in (Nathan and Klabjan, 2017),

on all instances considered in early iterations. More precisely, SODDA finds good quality solutions faster than RADiSA-avg.

The paper is organized as follow. In the next section, we review several related works in distributed optimization. In Section 3, we state the formal optimization problem and standard assumptions underlying our analyses, followed by the exposition of the SODDA algorithm. In Section 4, we show the convergence analyses of SODDA with respect to both a decreasing learning rate and a constant learning rate. In Section 5, we present experimental results comparing SODDA with RADiSA-avg.

1.2. Related Work

There are a large number of extensions of the plain stochastic gradient descent algorithm related to distributed datasets, however, a full retrospection of this immense literature exceeds the scope of this work. In this section, we state several approaches which are most related to our new method and interpret the relationships among them.

In plain SGD, the gradient of the aggregate function is approximated by one randomly picked function (Robbins and Monro, 1951). It saves a heavy load of computation when compared with gradient descent, whereas more often than not, the convergence happens to be slow. Recently, a large variety of approaches have been proposed targeted on accelerating the convergence rate and dealing with observations in the distributed setting.

SGD for distributed observations: One attempt that works for datasets with distributed observations is parallelizing it by means of mini-batch SGD. Both the synchronous version (Chen et al., 2016) and the asynchronous version (Tsitsiklis et al., 1986) basically work in the following way: the parameter server performs parameter updates after all worker nodes send their own

gradients based on local information in parallel, and then broadcasts the updated parameters to all worker nodes afterwards. In the synchronous approach, the master node needs to wait until all gradients are collected but in the asynchronous approach, the master node performs updates whenever it is needed. An alternative method which introduces the concept of variance reduced is CentralVR (De and Goldstein, 2016), where the master node not only needs to spread parameters but also the full gradient after every certain number of iterations, and each worker node would involve the full gradient as a corrector when computing their own gradients. As a consequence, the variance in the estimation of the stochastic gradient could be reduced and a larger learning rate is allowed to accomplish faster convergence and higher accuracy.

SGD for distributed features: Another attempt for distributed features is parallelization via features. Block successive upper bound minimization (BSUM) (Hong et al., 2015) is one of the methods working for datasets with distributed features, where the master node spreads all parameters and each worker node conducts parameter updates on a randomly chosen and non-overlapping subset of the feature vector. Distributed Block Coordinate Descent (Mareček et al., 2015) is another approach designed for datasets with distributed features. The parameters associated with these feature blocks are partitioned accordingly. In the algorithm, each processor randomly chooses a certain number of blocks out of those stored locally, performs the corresponding parameter updates in parallel, and then transmits to other processors. However, it is impossible to avoid communication when computing the gradient of all parameters unless there are extra assumptions on the objective loss function, which does not usually hold. An alternative approach is Communication-Efficient Distributed Dual Coordinate Ascent (CoCoA) (Jaggi et al., 2014), which is a primal-dual method also working for data with distributed features.

By exploiting the fact that the associated blocks of dual variables work in different processors without overlap, the algorithm aggregates the parallel updates efficiently from the different processors without much conflict and reduces the necessary communication dramatically. A faster converging method extended from the aforementioned approach is CoCoA$^+$ (Ma et al., 2015), which allows a larger learning rate for parameter updates by introducing a more generalized local CoCoA subproblem at each processor.

SGD for distributed observations and features: Given datasets with distributed observations and features, all the methods mentioned so far are not applicable. One of the algorithms fitting the bill is a block distributed ADMM (Parikh and Boyd, 2014), which is the block splitting variant of ADMM. Nonetheless, the convergence rate of ADMM-based methods is slow. Random parallel stochastic algorithm (RAPSA)(Mokhtari et al., 2016) is another algorithm which utilizes multiple parallel units to operate on a randomly chosen subset of blocks of the feature vector. It needs to access the whole feature vector to perform a parameter update while SODDA only needs a subset of the feature. Decentralized double stochastic averaging gradient algorithm (DSA) (Mokhtari and Ribeiro, 2016) is an alternative method designed for doubly distributed datasets, whereas the global cost function that DSA optimizes is a linear combination of the local objective functions which only contain local parameters, compared to the loss function of SODDA which contains global parameters. A faster converging and more pertinent algorithm is RADiSA (Nathan and Klabjan, 2017), which is also focusing on settings where both the observations and features of the problem at hand are stored in a distributed fashion. RADiSA conducts parameter updates based on stochastic partial gradients, which are calculated from randomly selected local observations and a randomly assigned sub-block of local features in

parallel. RADiSA is a special case of SODDA, since the full gradient required by it is replaced by an approximated gradient which only uses partial observations and features. Consequently, SODDA provides a faster convergence than RADiSA without sacrificing too much accuracy. Meanwhile, we present technical convergence analyses for SODDA under different types of learning rates which imply the convergence of RADiSA.

1.3. Algorithm

We consider the problem of optimizing a finite but large sum of smooth functions, i.e., given a training set $\{(x_i, y_i)\}_{i=1}^N$, where each $x_i^T \in \mathbb{R}^d$ is associated with a corresponding label y_i,

$$\min_{\omega \in \mathbb{R}^d} F(\omega) := \frac{1}{N} \sum_{i=1}^{N} \bar{f}(x_i\omega, y_i) = \frac{1}{N} \sum_{i=1}^{N} f_i(x_i\omega). \tag{1.1}$$

Several machine learning loss functions fit this model, e.g. least square, logistic regression, and hinge loss.

In SODDA, we assume that the training set $\{(x_i, y_i)\}_{i=1}^N$ is distributed across both observations and features. More specifically, the features and observations are split into Q and P partitions, respectively. We denote the matrix corresponding to the p'th observation partition and its q'th feature partition as $x^{p,q} \in \mathbb{R}^{\frac{N}{P} \times \frac{d}{Q}}$. Note that the partitions consisting of same features share the common block of parameters $\omega_{[q]}$. In Figure 1, there are 12 partitions. Parameters $\omega_{[1]}$ correspond to all parameters under $q = 1$. In order to efficiently parallelize the computation, optimization of each partition can be done concurrently by considering only local observations and features. This strategy poses a big challenge in how to combine the parameters. For example, $\omega_{[1]}$ are modified by all processors working on $x^{1,1}$, $x^{2,1}$, $x^{3,1}$, $x^{4,1}$. It is unclear how to combine them (averaging them is a possible strategy however this would not

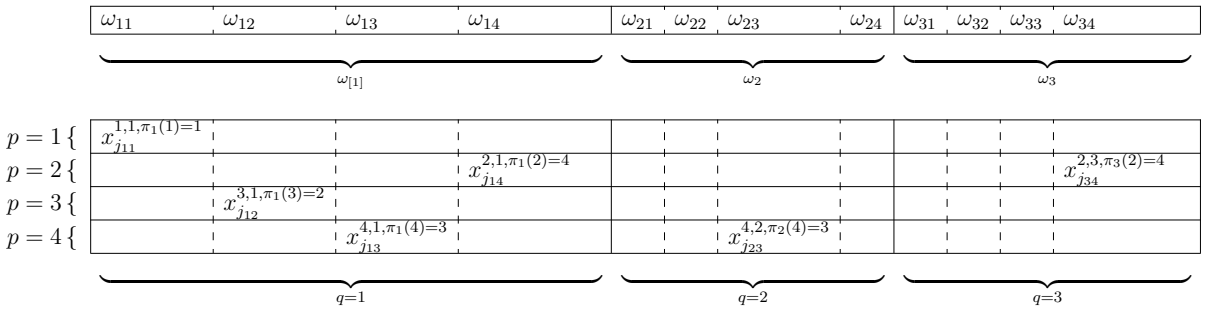


Figure 1.1. $Q = 3$, $P = 4$

yield convergence, see e.g. (Weimer et al., 2010), (Zinkevich et al., 2010)). To circumvent this, we further artificially subdivide the features.

To this end, we define a function $\pi_q(p) : \{1, 2, \cdots, P\} \to \{1, 2, \cdots, P\}$ in the corresponding q'th feature partition, where P is the number of partitions for observations. A sub-matrix of the training set from the partition $x^{p,q}$ corresponding to block $\omega_{q,\pi_q(p)}$ is denoted by $x^{p,q,\pi_q(p)}$, see Figure 1. Each processor operates on random observations from partition $x^{p,q}$ and all feature in $x^{p,q,\pi_q(p)}$. Note that $\omega_{[q]} = \left(\omega_{q,\pi_q(p)}\right)_{p=1}^{P}$. Let us define $n = N/P$, $m = d/Q$, $\tilde{m} = d/QP$, and $j_{q,\pi_q(p)}$ be randomly chosen from $\{1, 2, \cdots, n\}$ associated with sub-block $x^{p,q,\pi_q(p)}$. In Figure 1, where $P = 3$ and $Q = 4$, $x_{j_{12}}^{3,1,\pi_1(3)=2} \in \mathrm{R}^{\tilde{m}}$ represents a random observation j_{12} with a subset of features selected from the 2nd sub-block of the block corresponding to the observation partition 3 and feature partition 1 given $\pi_1(3) = 2$. Similarly, $x_{j_{23}}^{4,2,\pi_2(4)=3}$ symbolizes a random observation j_{23} with a subset of features selected from the 3rd sub-block of the block $x^{4,2}$.

Next, we introduce notation for the partial gradient. For any $\mathcal{C} \subseteq \{1, \cdots, d\}$ and any j, let us denote $\bar{\nabla}_{w_{\mathcal{C}}} f_j(\cdot) \in \mathbb{R}^d$ as the vector defined by

$$\left(\bar{\nabla}_{\omega_{\mathcal{C}}} f_j\left(\cdot\right)\right)_k = \begin{cases} 0, & k \notin \mathcal{C} \\ \left(\nabla f_j\left(\cdot\right)\right)_k, & k \in \mathcal{C}. \end{cases}$$

We need this notation since we sample gradient components. The loss function using this notation becomes

$$F(\omega) = \frac{1}{N} \sum_{k=1}^{P} \sum_{j=1}^{n} f_j^k \left(\sum_{q=1}^{Q} \sum_{p=1}^{P} x_j^{k,q,\pi_q(p)} \omega_{q,\pi_q(p)} \right),$$

where f_j^k is $\bar{f}$ associated with observation j in observation partition k.

Given the fact that the data is doubly distributed, SODDA further divides the features $x^{\cdot,q}$ into P subsets along all observations, i.e. $x^{\cdot,q} = [x^{\cdot,q,1}, \cdots, x^{\cdot,q,P}]$. In each iteration, SODDA first computes an approximation of the full gradient at the current parameter vector. Then, after randomly choosing a sub-matrix from each matrix $x^{q,p}$ as long as there is no overlap with respect to ω, each processor is assigned a sub-matrix of the local dataset and updates its local parameter by employing generalized SVRG. In the end of each iteration, SODDA concatenates all partial parameters which becomes the incumbent parameter vector for the next iteration.

The entire algorithm is exhibited in Algorithm 1. Steps 1- 3 initiate all the parameters. Steps 5-7 give the subsets of features and observations used to compute the partial gradient of the current iterate $\tilde{\omega}$ in step 8. Since the dataset is doubly distributed, the algorithm computes an estimate of the exact full gradient so as to reduce the communication cost in step 8. Additionally, we use the term no feature sampling to address the case when $\mathcal{b}^t = d$; in other words, the whole feature vector is employed to compute the gradient μ^t. To this end, we have three random components. The first one is the common one to sample observations. The second one is to compute only a random subset of subgradient coordinates, and the third one is to evaluate these components not at the exact $x\omega$ but only on the subset of the underlying inner product summation terms. Step 10 determines how sub-blocks are selected in each block of the dataset

associated with $\omega_{[q]}$, for each q. The definition of $(\pi_q)_{q=1}^{Q}$ guarantees that one, and only one sub-block is selected with respect to $\omega_{q,\pi_q(p)}$, i.e. $x^{p,q,\pi_q(p)}$. Then, for each sub-block $x^{p,q,\pi_q(p)}$, after randomly picking an observation $x^{p,q,\pi_q(p)}_{j_{q,\pi_q(p)}}$ in the selected sub-block in step 15, each block of parameter updates is given in step 16. Instead of using the full vector, we estimate the stochastic gradient by using local features and narrow down the variance by involving the approximated full gradient. Finally, at the end of each iteration, after each processor finishes its own task, step 19 aggregates all the updated partial solutions, i.e. $\omega = \left[\omega_{[1]}, \omega_{[2]}, \cdots, \omega_{[Q]}\right]$, where partial parameters $\omega_{[q]}$ represent the concatenation of the local parameters $\omega_{q,\pi_q(p)}$, for $p = 1, 2, \cdots, P$.

Algorithm 1 SODDA

Inputs:

batch size B, learning rate γ_t, sequence $\left\{\mathscr{b}^t, c^t, \mathscr{d}^t\right\}_{t=0}^{\infty}$ where $c^t \leq \mathscr{b}^t \leq d$, $\mathscr{d}^t \leq N$ for every t

Data:

$x^{p,q,k} \in \mathrm{R}^{n \times \tilde{m}}$ for $p, k = 1, \cdots, P$ and $q = 1, \cdots, Q$, $\omega_{q,\pi_q(p)} \in \mathrm{R}^{\tilde{m}}$

Initialize:

$w^0 \leftarrow 0$

for t = $0, 1, 2, \cdots$ **do**

  $\mathscr{B}^t$=$\mathscr{b}^t$ elements uniformly at random sampled without replacement from all features

  $\mathscr{C}^t$=c^t elements uniformly at random sampled without replacement from $\mathscr{B}^t$

  $\mathscr{D}^t$=$\mathscr{d}^t$ elements uniformly at random sampled without replacement from all observations

  $\mu^t = \frac{1}{\mathscr{d}^t} \sum_{j \in \mathscr{D}^t} \bar{\nabla}_{\omega_{\mathscr{C}^t}} f_j(x_j^{\mathscr{B}^t} \omega_{\mathscr{B}^t}^t)$

  for $q = 1, 2, \cdots, Q$, **do**

    select function $(\pi_q)_{q=1}^{Q}$

  end for

  for $p = 1, 2, \cdots, P$ and $q = 1, 2, \cdots, Q$, **do in parallel**

    $\bar{\omega}_{q,\pi_q(p)}^{(0)} = \omega_{q,\pi_q(p)}^t$

    for $i = 0, \cdots, B-1$ **do**

      randomly pick $j_{q,\pi_q(p)} \in \{1, \cdots, n\}$

      $$\bar{\omega}_{q,\pi_q(p)}^{(i+1)} = \bar{\omega}_{q,\pi_q(p)}^{(i)} - \gamma_{t+1} \Big[\nabla_{\omega_{q,\pi_q(p)}} f_{j_{q,\pi_q(p)}}^{\pi_q(p)}(x_{j_{q,\pi_q(p)}}^{p,q,\pi_q(p)} \bar{\omega}_{q,\pi_q(p)}^{(i)}) - \nabla_{\omega_{q,\pi_q(p)}} f_{j_{q,\pi_q(p)}}^{\pi_q(p)}(x_{j_{q,\pi_q(p)}}^{p,q,\pi_q(p)} w_{q,\pi_q(p)}^t) + \mu_{q,\pi_q(p)}^t\Big]$$

    end for

  end for

  $\omega^{t+1} = \left[\omega_{[1]}, \omega_{[2]}, \cdots, \omega_{[Q]}\right]$, where $\omega_{[q]} = \left[\bar{\omega}_{q1}^{(B)}, \bar{\omega}_{q2}^{(B)}, \cdots, \bar{\omega}_{qP}^{(B)}\right]$

end for

1.4. Analysis

In this section, we prove that the sequence of the loss function values $F(\omega^t)$ generated by SODDA approaches the optimal loss function value $F(\omega^*)$. We assume the existence and the uniqueness of the minimizer ω^* that achieves the optimal loss function value. Meanwhile, we require the following standard assumptions.

Assumption 1:

- The expectation function $F(\omega)$ is strongly convex with parameter $\xi > 0$.

Assumption 2:

- The loss gradients $\nabla f_i(x_i\omega)$ are Lipschitz continuous with respect to the Euclidian norm with parameter $L \geq 1$, i.e., for all ω, $\hat{\omega} \in \mathbb{R}^d$ and any i, it holds

$$\|\nabla f_i(x_i\omega) - \nabla f_i(x_i\hat{\omega})\| \leq L \|\omega - \hat{\omega}\| .$$

In Assumption 1, only the expected loss function $F(\omega)$ is enforced to be strongly convex, whereas the individual loss functions f_i could even be non-convex. Notice that in Assumption 2, since each individual function ∇f_i is imposed to be Lipschitz-continuous with constant L with respect to ω, both the gradient of the expected loss function $\nabla F(\omega)$ and the individual function ∇f_i are L-Lipschitz continuous with respect to ω and $\omega_{q,\pi_q(p)}$ for any $q \in \{1, 2, \cdots, Q\}$ and any $p \in \{1, 2, \cdots, P\}$. Note that if f_i's are ϵ-Lipschitz continuous for some $0 < \epsilon \leq 1$, we can take $L = 1$. Moreover, these assumptions hold for several widely used machine learning loss functions, i.e. hinge, square, logistic loss.

Under these standard assumptions, by finding a relationship for the sequence of the loss function errors $F(\omega^t) - F(\omega^*)$ and employing the supermartingale convergence argument,

which is a standard technique for analyzing stochastic optimization problems (see e.g. textbooks (Benveniste et al., 2012), (Bertsekas and Tsitsiklis, 1989), (Borkar, 2008)), we prove that the sequence of the loss function values $F(\omega^t)$ converges to the optimal function value $F(\omega^*)$ when using the standard diminishing learning rate, i.e. non-summable and squared summable.

Theorem 1. *If there is neither feature sampling in step 5 nor sample sampling in step 7 and Assumptions 1-2 hold, and the number of iteration for the inner loop B is* $B \geq C_1 d$ *where* C_1 *is a positive constant, and the sequence of learning rates satisfies* $\gamma_t \leq 1$ *for all* t*, and are non-summable* $\sum_{t=1}^{\infty} \gamma_t = \infty$ *and square summable* $\sum_{t=1}^{\infty} \gamma_t^2 < \infty$*, and the sequence* $(c^t)_{t=0}^{\infty}$ *is selected so that* $c^t \leq d$*, then we have geometric convergence in expectation for SODDA*

$$\mathbb{E}[F(\omega^{t+1})] \leq F(\omega^*) + (\prod_{j=1}^{t} A_j) \left[F(\omega^0) - F(\omega^*)\right], \tag{1.2}$$

where

$$\lim_{t\to\infty} \prod_{j=1}^{t} A_j = 0.$$

PROOF. See Appendix A.3. □

Based on the fact that the exact form for the update step in expectation is not available in SODDA, i.e., we do not explicitly know $\omega^{t+1} - \omega^t$ any technique that relies on such an explicit formula is inappropriate. This expectation is given by steps 9-18 in the algorithm. The main challenges are coming from evaluating individual function gradient $\triangledown f(x_{qp}\omega_{q,\pi_q(p)})$ and grouping different sub-blocks all together. The way we deal with it, which is borrowed from (Bertsekas and Tsitsiklis, 2000), is grouping the first two partial gradients together and treating the last partial gradient $\mu^t_{q,\pi_q(p)}$ as a corrector.

The very technical proof follows the following steps. In steps 14-17, SODDA utilizes local features from a random observation to update the corresponding subset of parameters, and involves the information from the estimated full gradient to reduce unreasonable fluctuation. Therefore, in association with the conditional Jensen's inequality and properties of Lipschitz continuity, the norm of the difference and the square norm of the difference of the first two terms in the bracket of the updating procedure in step 16 are able to be bounded by a function involving γ_t. Thus, representing ω^{t+1} as a function of ω^t and applying strong convexity of F, coupled with all the bounds derived before, yield a relationship for the sequence of loss function errors $F(\omega^t) - F(\omega^*)$. Combined with the property that γ_t is non-summable but square summable, (1.2) is achieved by applying Taylor expansion of logarithm.

Theorem 1 asserts the convergence of the expectation of the objective loss generated by SODDA with non-summable and squared summable learning rate.

In order to allow feature sampling we have to control the growth of ω^t.

1.4.1. Analyses with Feature Sampling

To this end, we require the following assumptions together with Assumptions 1 and 2. In this subsection, we also do not require $\mathcal{b}^t = d$, i.e., step 5 in Algorithm 1 now requires sampling.

Assumption 3:

- There exists a constant M_2, such that

$$\left\|\omega^t\right\| \leq \frac{M_2}{2},$$

  for any t.

Assumption 4:

- The sample variance of the norms of the gradients is bounded by G^2 for all ω^t, i.e.

$$\frac{1}{N-1}\sum_{j=1}^{N}\left(\left\|\nabla f_j(x_j\omega^t)\right\|^2-\left\|\nabla F(\omega^t)\right\|^2\right)\leq G^2.$$

The restriction in Assumption 3 is reasonable and also has been used in work (Harikandeh et al., 2015). Assumption 4 is also standard, see e.g. (Harikandeh et al., 2015). Moreover, these assumptions hold for several widely used machine learning loss functions, i.e. hinge, square, logistic loss. Notice that without Assumption 3, we can not further assume the boundness of the sample variance of the norms of the gradients in Assumption 4. Next, we present an example showing that SODDA does not converge under Assumptions 1 and 2.

Theorem 2. *There is a convex loss function and $\mathcal{P}$ where SODDA does not converge when only given Assumptions 1 and 2, and any $\gamma_t \leq K$ for every t and constant K depending on input data.*

PROOF. See Appendix A.4. □

The main role of Assumption 3 is to maintain a reasonable error generated by the stochastic partial gradients in steps 16. Then, under these standard assumptions and applying the similar trick as in the proof of Theorem 1, we argue that the sequence of ω^t enjoys the almost sure convergence to ω^*.

Theorem 3. *If Assumptions 1-4 hold, and the sequence of learning rates are non-summable $\sum_{t=1}^{\infty}\gamma_t=\infty$ and square summable $\sum_{t=1}^{\infty}\gamma_t^2<\infty$, and the sequences $(\mathcal{b}^t,c^t,\mathcal{d}^t)_{t=0}^{\infty}$ are selected so that $\mathcal{b}^t\in\left[\max\left\{c^t,\frac{d}{1+\frac{4d\eta\gamma_{t+1}^2}{c^tM_2^2L^2}}\right\},d\right]$ for some constant $\eta\geq 0$, $c^t\leq d$ and $\mathcal{d}^t\leq N$, then the sequence of parameters ω^t generated by SODDA converges almost surely to the optimal*

solution ω^*, that is

$$\lim_{t\to\infty} \left\|\omega^t - \omega^*\right\| = 0 \quad \text{a.s.} \tag{1.3}$$

PROOF. See Appendix A.5. □

The proof of Theorem 3 is very similar to the proof of Theorem 1. The only difference is caused by feature sampling. The main challenges are how to pick a suitable $\mathcal{b}^t$ and how to narrow down the error generated by $\mathcal{b}^t$. Then, given an appropriate $\mathcal{b}^t$, the norm of the estimator of the full gradient μ^t and its square are bounded by a function containing the full gradient at ω^t and the learning rate γ_t. Meanwhile, η is a positive constant which controls the divergence of the approximate full gradient from the exact full gradient in step 8, i.e. when η is 0, the whole feature vector is used as $\mathcal{b}^t = d$. The rest of the proof is identical to the proof of Theorem 1.

Theorem 3 asserts the almost sure convergence of the iterates generated by SODDA with non-summable and squared summable learning rate. Furthermore, given $\gamma_t = \frac{1}{t}$ and B big enough, the following theorem states that the loss function $F(\omega^t)$ converges to the optimal value $F(\omega^*)$ with probability 1 and the rate of convergence in expectation is at least in the order of $\mathcal{O}(\frac{1}{t})$.

Theorem 4. *Under Assumptions 1-4, if the learning rate is defined as $\gamma_t := \frac{1}{t}$ for $t = 1, 2, \cdots$, and the batch size is chosen such that $B \geq \frac{d}{2\xi}$, and the sequence $(\mathcal{b}^t, c^t, \mathcal{d}^t)_{t=0}^{\infty}$ satisfies the same conditions as in Theorem 3, then there exists a positive constant C_3 such that the expected loss function errors $\mathbb{E}[F(\omega^t) - F(\omega^*)]$ of SODDA converges to 0 at least with a*

sublinear convergence rate of order $\mathcal{O}(1/t)$*, i.e.*

$$\mathbb{E}[F(\omega^t) - F(\omega^*)] \leq \frac{Q}{1+t}, \tag{1.4}$$

where constant Q *is defined as*

$$Q = \max\left\{F(\omega^0) - F(\omega^*), \cdots, ([\lambda]+2)\,\mathbb{E}[F(\omega^{[\lambda]+1}) - F(\omega^*)], \frac{C_3}{\lambda - 1}\right\}, \tag{1.5}$$

with $\lambda = \frac{2\xi B}{d}$.

PROOF. See Appendix A.5. □

Given the specific relationship between the learning rate γ_t and the iterator t, i.e. $\gamma_t = 1/t$, applying the supermartingale convergence theorem and performing induction on an upper bound of $\mathbb{E}\left[F(\omega^t) - F(\omega^*)\right]$ allow us to establish at least sublinear convergence of SODDA.

A diminishing learning rate is beneficial if the exact convergence is required. If we are only interested in a specific accuracy, it is more efficient to choose a constant learning rate. In the following theorem, we employ a similar argument used in proving Theorem 3 and Theorem 4 except that B and γ are linked by a condition. Again by providing a supermartingale relationship for the sequence of the loss function errors $F(\omega^t) - F(\omega^*)$, we are able to study the convergence properties generated by SODDA for a constant learning rate γ.

Theorem 5. *If Assumptions 1-4 hold, and the learning rate is constant* $\gamma_t = \gamma$ *such that* $BL\gamma QP \leq 1$*, which also implies that* $\gamma \leq 1$*, and the sequence* $(\mathcal{b}^t, c^t, \mathcal{d}^t)_{t=0}^{\infty}$ *satisfies the same conditions as in Theorem 3, then there exists a positive constant* C_4 *such that the sequence of parameters* ω^t *generated by SODDA converges almost surely to a neighborhood of the optimal*

solution ω^*, that is

$$\liminf_{t\to\infty} F(\omega^t) - F(\omega^*) \leq \frac{C_4 d B^3 \gamma}{2\xi} \quad \text{a.s.} \tag{1.6}$$

Moreover, if the constant learning rate γ is chosen such that $\gamma < \min\left\{\frac{d}{2\xi B}, \frac{1}{BLQP}, 1\right\}$, then the expected loss function errors $\mathbb{E}\left[F(\omega^t) - F(\omega^*)\right]$ converges linearly to an error bound as

$$\mathbb{E}\left[F(\omega^t) - F(\omega^*)\right] \leq \left(1 - \frac{2\xi B}{d}\gamma\right)^t \left(F(\omega^0) - F(\omega^*)\right) + \frac{C_4 d B^3 \gamma}{2\xi}. \tag{1.7}$$

PROOF. See Appendix A.6. □

Note that $BL\gamma QP \leq 1$ trades off γ and B, i.e. the larger B is, the smaller γ must be. The major difficulties are similar but not identical to those of Theorem 3. We apply a similar idea but treating both B and $\gamma_t = \gamma$ as variables to obtain an upper bound in closed form for the difference of the first two partial gradients in step 16. Then Lipschitz continuity of $\nabla F(\omega)$ leads to a supermartingale relationship for the sequence of the loss function errors $F(\omega^t) - F(\omega^*)$. As a consequence, claims in (1.6) and (1.7) follow according to the supermartingale convergence theorem. The only distinction between Theorem 3 and Theorem 5 is caused by the property of the learning rate. The error exists in each iteration, which is a function of the learning rate γ_t, however, in Theorem 3, the error function goes to 0 as the number of iterations increases, which is not the case when the learning rate is a constant. Therefore, we can only ensure a relatively high-quality solution.

Theorem 5 guarantees that SODDA finds good quality solutions when using an appropriate learning rate γ and batch size B. Notice that although methods of type SVRG achieve linear convergence to the exact solution in expectation under a constant step size, SVRG performs

the exact full gradient after every certain number of iterations which would trigger emergence of communication under the doubly distributed setting and is unnecessary especially in early iterations. In addition, based on (1.7), there is a trade-off between the accuracy and the convergence rate. Although reducing the learning rate γ or batch size B narrows down the error bound $\frac{C_4 dB^3\gamma}{2\xi}$ and contributes significantly to a more accurate convergence, the constant convergence rate $1 - \frac{2\xi B}{d}\gamma$ suffers greatly since it increases and gets closer to 1, which leads to a slower convergence rate.

To address this problem, in the following theorem, by considering not only the loss function errors $F(\omega^t) - F(\omega^*)$ but also the errors $\|\omega^t - \omega^*\|^2$, we prove that the sequence of the loss function values $F(\omega^t)$ generated by SODDA converges to the optimal value $F(\omega^*)$ for any constant learning rate selected from a certain region. In addition, we are able to further assert that the sequence of ω^t converges to ω^* when taking Assumption 1 into account. Furthermore, since we employ an approximation of the exact full gradient for the sake of the efficiency of the algorithm in Theorem 5, the algorithm converges only to a neighborhood of an optimal solution under a constant learning rate. In the following theorem, if we are allowed to employ the exact full gradient in expectation, then the algorithm in Theorem 6 converges to the exact solution in expectation under a constant step size.

Theorem 6. *If Assumptions 1-4 hold, and the learning rate $\gamma_t = \gamma$ is a constant such that $\gamma \in (0, \min\left\{1, \frac{1}{BLQP}, \gamma_1, \gamma_2\right\})$, where both γ_1 and γ_2 are positive constants specified in Appendix H, and the sequence $(\mathcal{b}^t, c^t, \mathcal{d}^t)_{t=0}^{\infty} = (d, c^t, N)_{t=0}^{\infty}$ for arbitrary positive $c^t \leq d$, then the sequence of parameters ω^t generated by SODDA converges to ω^*, that is*

$$\lim_{t\to\infty} \left\|\omega^t - \omega^*\right\| = 0. \tag{1.8}$$

PROOF. See Appendix A.7. □

The Lyapunov analysis, which is a common strategy to deal with a constant learning rate (see e.g. (Schmidt et al., 2017)), fails for our algorithm due to the analogous reasons as those for Theorem 3. The success of the Lyapunov analysis heavily relies on the number of negative terms available when computing the loss function errors $F(\omega^t) - F(\omega^*)$ and the errors $\omega^t - \omega^*$. Unfortunately, the doubly distributed data setting results in lack of information in each iteration in step 16, which leads to a scarcity of negative terms to ensure the decrease of the loss function value.

Our steps to study the convergence analysis are as follows. We first establish either exact forms or upper bounds for all terms involving gradients. Then, from the update rule, we find a criteria for the constant learning rate γ so as to make the errors $\omega^t - \omega^*$ at least not increase as the number of iterations increases. In addition, given Lipschitz continuity of $\triangledown F(\omega)$, we find a recursive formula regarding the loss function error, which provides another constraint for γ such that the loss function error vanishes as t increases. Finally, the convergence of SODDA and the existence of γ are guaranteed by two cubic inequality constraints aforementioned.

1.5. Numerical Study

In this section, we compare the SODDA method with RADiSA-avg (Nathan and Klabjan, 2017), which is the best known optimization algorithm for solving problem (1) with doubly distributed data. All the algorithms are implemented in Scala with Spark 2.0. The experiments are conducted in a Hadoop cluster with 4 nodes, each containing 8 Intel Xeon 2.2GHz cores. We conduct experiments on three different-size synthetic datasets that are larger than the datasets in (Nathan and Klabjan, 2017) and two datasets used in (Wongchaisuwat and Klabjan, 2018)

extracted from SemMed Database. For all of these datasets, we train one of the most popular classification models: binary classification hinge loss support vector machines (SVM), and set the learning rate $\gamma_t = \frac{1}{(1+\sqrt{t-1})}$, which is also employed in (Nathan and Klabjan, 2017). Furthermore, we set the feature partition number $Q = 3$ and observation partition number $P = 5$, which is also one of the cases studied in (Nathan and Klabjan, 2017). We do not compare different learning rates and Q, P since these have been extensively studied in (Nathan and Klabjan, 2017).

1.5.1. SVM with Synthetic data

We first compare SODDA with RADiSA-avg (Nathan and Klabjan, 2017) using synthetic data. The datasets for these experiments are generated based on a standard procedure introduced in (Zhang et al., 2012), which is also used in (Nathan and Klabjan, 2017): the x_i's and z are sampled from the uniform distribution in $[-1, 1]$, and $y_i := \text{sgn}(x_i z)$ with probability 0.01 of flipping the sign. In addition, all the data is in the dense format and the features are standardized to have unit variance. The size of each partition from the small-size dataset is $50,000 \times 6,000$, the one from the mid-size data is $60,000 \times 7,000$ and the one from the large-size data is $60,000 \times 9,000$. The information about these three datasets is listed in Table 1.1.

data size	small	medium	large
$P \times Q$	5×3	5×3	5×3
size of each partition	$50,000 \times 6,000$	$60,000 \times 7,000$	$60,000 \times 9,000$
Number of Spark executors used	18	25	25

Table 1.1. Synthetic datasets for numerical experiments

First, we conduct $\mathcal{b}^t, c^t, \mathcal{d}^t$ subsequence related experiments. We justify the value of $(\mathcal{b}^t, c^t, \mathcal{d}^t)$ from the small-size dataset, since the other two datasets would take more computational time. We study the impact of $(\mathcal{b}^t, c^t, \mathcal{d}^t)$ to the performance of SODDA by varying one of the three parameters $(\mathcal{b}^t, c^t, \mathcal{d}^t)$ while keeping the other two parameters fixed.

The most important results are presented in Figure 1.2. In Figure 1.2(a), we study the cases where the number of total observations used to estimate the full gradient in step 8 varies from 60% to 90% with $\mathcal{b}^t = c^t = 100\%$. In Figure 1.2(b), we consider the cases when c^t varies from 40% to 80% given that every feature is involved to compute the approximated full gradient, i.e. $\mathcal{b}^t = 100\%$. Figure 1.2(c) represents the cases where only partial features are used in step 8 but everything available is fully used, i.e. $\mathcal{b}^t = c^t$. In Figures 1.2(d)-(f), we study three different $\mathcal{b}^t$ choices and for each one we vary c^t. Figure 1.2(g) is an extension of Figure 1.2(d) showing the long-time performance under the corresponding set of parameters.

In these plots, we observe that every set of parameters $(\mathcal{b}^t, c^t, \mathcal{d}^t)$ with the small-size dataset outperforms RADiSA-avg in early iterations, however, the benefits peak at certain points. More precisely, from Figure 1.2(a), we discover that the marginal benefit grows up dramatically when $\mathcal{d}^t$ increases from 60% to 80% and slows down from 80% to 90%, thus, $\mathcal{d}^t = 85\%$ seems to be most beneficial. When it comes to c^t, we observe that although the value of c^t does not influence the accuracy of the solution, a higher value of c^t leads to a faster convergence speed to a good quality solution in Figure 1.2(b). Thus, we set $c^t = 80\%$ as a good value. From Figures 1.2(c)-(g), we observe that the value of $\mathcal{b}^t$ affects the accuracy of the solution significantly, therefore, we set $\mathcal{b}^t = 85\%$ after taking both the accuracy of the solution and the computational time into consideration.

In these figures, we observe that SODDA always outperforms RADiSA-avg in early iterations on the small-size dataset, and there is a trade-off between the accuracy of the loss function value and the sampling sizes used in the algorithm. More precisely, using less data leads to a faster convergence speed but a less accurate solution, while using more data contributes to a more accurate solution but requires more time.

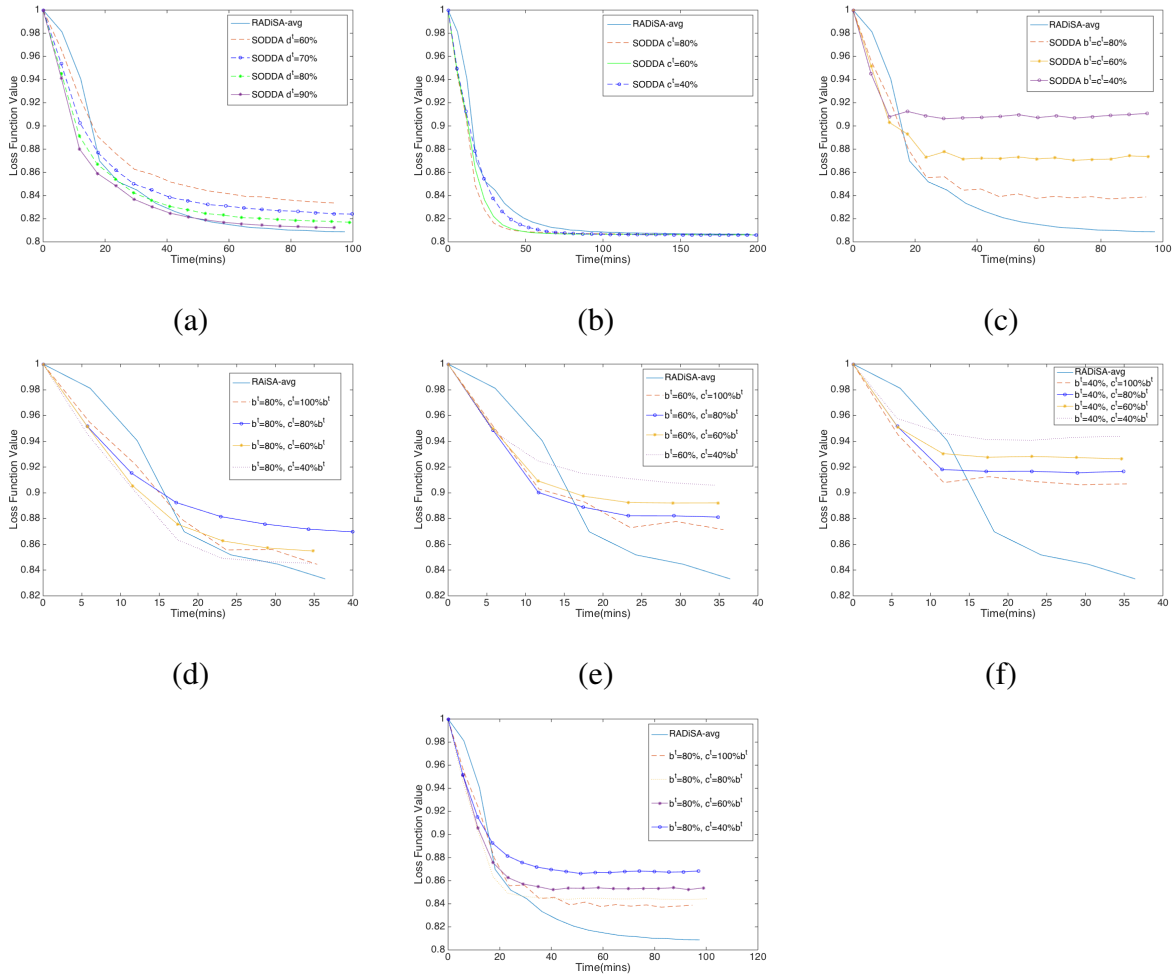
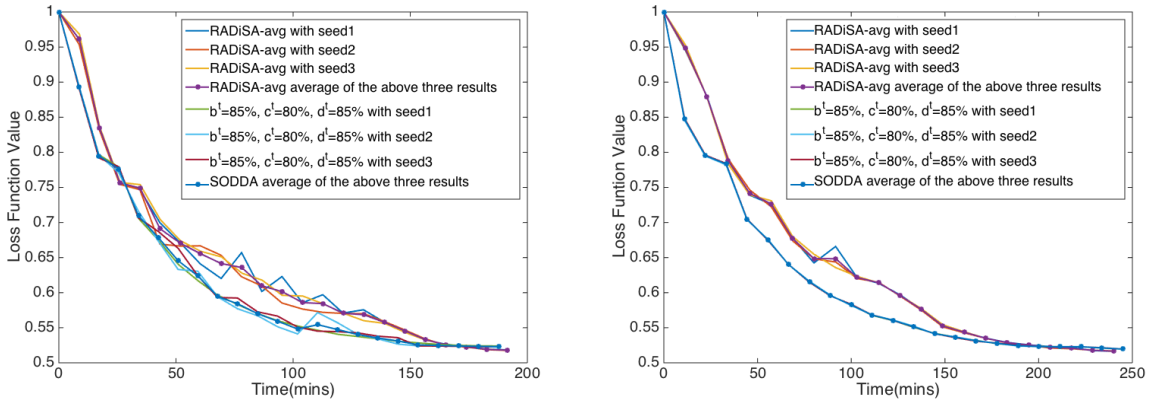


Figure 1.2. Comparison of SODDA and RADiSA-avg on small-size dataset

After specifying the values of $\mathcal{b}^t$, c^t and $\mathcal{d}^t$ to be $(85\%, 80\%, 85\%)$, we test both SODDA and RADiSA-avg on the mid- and large-size datasets with three different seeds. The results are presented in Figure 1.3. As we can observe, SODDA always exhibits a stronger and faster convergence than RADiSA-avg. It is interesting that as the size of the dataset increases, the intersection time of SODDA and RADiSA-avg comes later, which gives SODDA more advantages over RADiSA-avg when dealing with large datasets.



(a) (b)

Figure 1.3. Comparison of SODDA and RADiSA-avg for three different seeds on the mid- and large-size datasets

In the SODDA algorithm, we randomly choose a subset of observations and a block of features to estimate the full gradient in step 8. Moreover, both SODDA and RADiSA-avg utilize an observation randomly selected from a randomly chosen sub-matrix in the update step, where SODDA employs a sub-block of the approximated full gradient as a corrector but RADiSA-avg employs the exact full gradient. In order to eliminate the uncertainty about the choice of seeds, we conduct experiments on the large-size dataset under the same set of parameter $(\mathcal{b}^t, c^t, \mathcal{d}^t)$

with different seeds. Table 1.2 summarizes the influence of the change of the seed on the large-size dataset. For 10 different seeds, we run 40 iterations for each. The first two columns present the average of the difference of the maximum objective value and the average function value across the 10 seeds, and the average of the difference of the average function value and the minimum objective value across the 10 seeds, respectively. Similarly, the remaining terms are defined as the maximum of the difference of the maximum objective value and the average function value, and the maximum of the difference of the average function value and the minimum objective value. As we can see in Table 1.2, the perturbation caused by the change of the seed is negligible especially when compared to the objective function value, which is a positive characteristic. Thus, in the remaining experiments, we no longer need to consider the impact of the randomness caused by either SODDA or RADiSA-avg.

	avg(max-avg)	avg(avg-min)	max(max-avg)	max(avg-min)
SODDA	0.4600×10^{-4}	0.0251×10^{-4}	0.2500×10^{-3}	3.0000×10^{-3}
RADiSA-avg	1.6373×10^{-4}	1.2606×10^{-4}	1.8000×10^{-3}	2.3500×10^{-3}

Table 1.2. Variation of SODDA and RADiSA-avg by using different seeds

1.5.2. SVM with SemMed Database

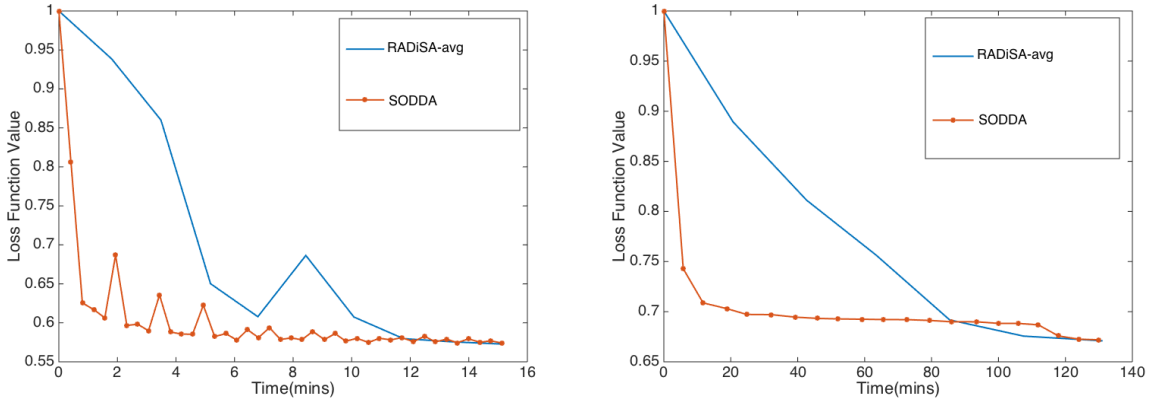
In the last set of experiments, we study the performances of SODDA with the $(\beta^t, c^t, \partial^t)$ selected in the previous section and RADiSA-avg on the Semantic MEDLINE Database (Kilicoglu et al., 2012) with SemRep, a semantic interpreter of biomedical text (Rindflesch and Fiszman, 2003) as an extraction tool to construct the knowledge graph (KG). Like the preprocessing done in (Wongchaisuwat and Klabjan, 2018), we apply the inference method, which is called the Path

Ranking Algorithm (PRA) (Lao and Cohen, 2010), to KG constructed from SemRep. The model under consideration is still linear SVM, and all the datasets considered are in the sparse format. The first dataset DIAG-neg10 is based on relationship "DIAGNOSES," while LOC-neg5 is created in a similar manner based on "LOCATION OF." The data is summarized in Table 1.3.

Figure 1.4 illustrates the convergence paths of the objective loss function $F(\omega)$ generated by SODDA and RADiSA-avg versus time. We observe that using SODDA is much better than RADiSA with respect to not only the running time but also the loss reduction in early iterations. Comparing Figure 1.4(a) with Figure 1.4(b), we discover that the superior behavior of RADiSA over RADiSA-avg is more apparent and robust when applied to larger datasets, which is expected since it is more beneficial for datasets with larger size to perform partial computation instead of full computation of gradients in step 8.

Dataset	Observations (N)	Features (d)	Size of each partition ($n \times m$)
DIAG-neg10	425,185	26,946	$85,037 \times 8,982$
LOC-neg5	5,638,696	26,966	$1,127,740 \times 8,989$

Table 1.3. Datasets extracted from SemMed database



(a) DIAG-neg10 (b) LOC-neg5

Figure 1.4. Comparison of SODDA and RADiSA-avg on SemMed database

1.5.3. Key Findings

From the first set of experiments conducted on different synthetic datasets in the dense format, we justify a good set of parameters $(\beta^t, c^t, \vartheta^t) = (85\%, 80\%, 85\%)$ and eliminate the potential impact of the randomness involved in SODDA and RADiSA to the performance of the convergence. Furthermore, we discover that SODDA always exhibits a stronger and faster convergence than RADiSA-avg for every dataset considered and parameter values chosen. In the second set of experiments, we observe the same dominance of SODDA when compared to RADiSA-avg on sparse datasets.

In conclusion, SODDA provides a faster, stronger and more robust convergence than RADiSA-avg for both dense and sparse datasets.

CHAPTER 2

Convergence Analyses of Online ADAM

2.1. Introduction

Online learning is the process of dynamically incorporating knowledge of the geometry of the data observed in earlier iterations to perform more informative learning in later iterations, as opposed to standard machine learning techniques which provide an optimal predictor after training over the entire dataset. Online learning is a preferred paradigm in situations where the algorithm has to dynamically adapt to new patterns in the dataset, or when the dataset itself is generated as a function of time, i.e. stock price prediction. Online learning is also used when the dataset itself is computationally infeasible to be trained over the entire dataset.

In standard online learning it is assumed that a finite number of samples is encountered however in real world streaming setting an infinite number of samples is observed (e.g., Twitter is streaming since inception and will continue to do so for foreseeable future). The performance of an online learning algorithm on early examples is negligible when measuring the performance or making predictions and decisions on the later portion of a dataset (the performance of an algorithm on tweets from ten years ago has very little bearing on its performance on recent tweets). The problem can be tackled by restarting, however, it is challenging to determine when to restart. For this reason we propose a metric which forgets about samples encountered a long time ago. Consequently, we introduce a performance metric, regret with rolling window, which measures the performance of an online learning algorithm over a possible infinite size dataset.

This metric also requires an adaptation of prior algorithms, because, for example, a diminishing learning rate has poor performance on an infinite data stream.

Stochastic gradient descent (SGD) (Zinkevich, 2003) is a widely used approach in areas of online machine learning, where the weights are updated each time a new sample is received. Furthermore, it requires a diminishing learning rate in order to achieve a high-quality performance. It has been empirically observed that, in order to reduce the impact of the choice of the learning rate and conduct stochastic optimization more efficiently, the adaptive moment estimation algorithm (ADAM) (Kingma and Ba, 2015) and its extensions ((Luo et al., 2019),(Reddi et al., 2018)) are another type of popular methods, which store an exponentially decaying average of past gradients and squared gradients and applies adaptive learning rate. (In standard gradient descent algorithms we use the term learning rate, while in adaptive learning rate algorithms we call stepsize the hyperparameter that governs the scale between the weights and the adjusted gradient.) In spite of this, no contribution has been made to the case where the regret is computed in a rolling window. Moreover, applying a diminishing learning rate or stepsize to regret with rolling window is not a good strategy, otherwise, the performance is heavily dependent on the learning rate or stepsize and the rank of a sample. Namely, regret with rolling window requires a constant learning rate or stepsize.

Standard online setting has been studied in the convex setting. With improvements in computational power resulting from GPUs, deep neural networks have been very popular in a variety of AI problems recently. A core application of online learning is online web search and recommender systems (Zoghi et al., 2017) where deep learning solutions have recently emerged. At the same time, online learning based on deep neural networks has become an integral role in many stages in finance, from portfolio management, algorithmic trading, to fraud detection, to

loan and insurance underwriting. To this end we focus not only on convex loss functions, but also on deep neural networks.

In this work, we propose a new family of efficient online subgradient methods for both general convex functions and a two-layer ReLU neural network based on regret with rolling window metric. More precisely, we first present an algorithm, namely convergent Adam (CONVGADAM), designed for general strictly convex functions based on gradient descent using adaptive learning rate and inspired by the work of (Reddi et al., 2018). CONVGADAM is a more general algorithm that can dynamically adapt to an arbitrary sequence of strictly convex functions. In the meanwhile, we experimentally show that CONVGADAM outperforms state-of-the-art, yet non-adaptive, online gradient descent (OGD) (Zinkevich, 2003). Then, we propose an algorithm, called deep neural network gradient descent (DNNGD), for a two-layer ReLU neural network. DNNGD takes standard gradient first, then it rescales the weights upon receiving each new sample. Lastly, we introduce a new algorithm, deep neural network Adam (DNNADAM), which uses an adaptive learning rate for the two-layer ReLU neural network. DNNADAM is first endowed with long-term memory by using gradient updates scaled by square roots of exponential decaying moving averages of squared past gradients and then it rescales weights with every new sample.

In this paper, we not only propose a new family of gradient-based online learning algorithms for both convex and non-convex loss functions, but also present a complete technical proof of regret with rolling window for each of them. For strongly convex functions, given a constant learning rate, we show that CONVGADAM attains regret with rolling window which is proportional to the square root of the size of the rolling window, compared to the true regret $\mathcal{O}(\log(T)\sqrt{T})$ of AMSGRAD (Reddi et al., 2018). Besides, we not only point out but also fix

the problem in the proof of regret for AMSGRAD later in this paper. Table B.1 in Appendix B.1.2 summarizes all regret bounds in various settings, including the previous flawed analyses. Furthermore, we prove that both DNNGD and DNNADAM attain the same regret with rolling window under reasonable assumptions for the two-layer ReLU neural network. The strongest assumption requires that the angle between the current sample and weight error is bounded away from $\pi/2$. Although DNNGD and DNNADAM require some assumptions, these two algorithms have a higher probability to converge than other flavors of ADAM due to the convergence analyses provided in Section 5. In summary, we make the following five contributions.

- We introduce regret with rolling window that is applicable in data streaming, i.e., infinite stream of data.
- We provide a proof of regret with rolling window which is proportional to the square root of the size of the rolling window when applying OGD to an arbitrary sequence of convex loss functions.
- We provide a convergent first-order gradient-based algorithm, i.e. CONVGADAM, employing adaptive learning rate to dynamically adapt to the new patterns in the dataset. Furthermore, given strictly convex functions and a constant stepsize, we provide a complete technical proof of regret with rolling window. Besides, we point out a problem with the proof of convergence of AMSGRAD (Reddi et al., 2018), which eventually leads to $\mathcal{O}(\log(T)\sqrt{T})$ regret in the standard online setting, and we provide a different analysis for AMSGRAD which obtains $\mathcal{O}(\sqrt{T})$ regret in standard online setting by using our proof technique.

- We propose a first-order gradient-based algorithm, called DNNGD, for the two-layer ReLU neural network. Moreover, we show that DNNGD shares the same regret with rolling window with OGD when employing a constant learning rate.
- We further develop an algorithm, i.e. DNNADAM, based on adaptive estimation of lower-order moments for the two-layer ReLU neural network. At the same time, we argue that DNNADAM shares the same regret with rolling window with CONVGADAM when employing a constant stepsize.
- We present numerical results showing that CONVGADAM outperforms state-of-art, yet not adaptive, OGD.

The paper is organized as follow. In the next section, we review several works related to ADAM, analyses of two-layer neural networks and regret in online convex learning. In Section 3, we state the formal optimization problem in streaming, i.e., we introduce regret with rolling window. In the subsequent section we propose the two algorithms in presence of convex loss functions and we provide the underlying regret analyses. In Section 5 we study the case of deep neural networks as the loss function. In Section 6 we present experimental results comparing CONVGADAM with OGD.

2.2. Related Work

ADAM *and its variants:* ADAM (Kingma and Ba, 2015) is one of the most popular stochastic optimization methods that has been applied to convex loss functions and deep networks which is based on using gradient updates scaled by square roots of exponential moving averages of squared past gradients. In many applications, e.g. learning with large output spaces, it has been empirically observed that it fails to converge to an optimal solution or a critical point in

nonconvex settings. A cause for such failures is the exponential moving average, which leads ADAM to forget about the influence of large and informative gradients quickly (Chen et al., 2019). To tackle this issue, AMSGRAD (Reddi et al., 2018) is introduced which has long-term memory of past gradients. ADABOUND (Luo et al., 2019) is another extension of ADAM, which employs dynamic bounds on learning rates to achieve a gradual and smooth transition from adaptive methods to stochastic gradient. Though both AMSGRAD (Reddi et al., 2018) and ADABOUND (Luo et al., 2019) provide theoretical proofs of convergence in a convex case, very limited further research related to ADAM has be done in a non-convex case while ADAM in particular has become the default algorithm leveraged across many deep learning frameworks due to its rapid training loss progress. Unfortunately, there are flaws in the proof of AMSGRAD, which is explained in a later section and articulated in Appendix A.

Two-layer neural network: Deep learning achieves state-of-art performance on a wide variety of problems in machine learning and AI. Despite its empirical success, there is little theoretical evidence to support it. Inspired by the idea that gradient descent converges to minimizers and avoids any poor local minima or saddle points ((Lee et al., 2016), (Lee et al., 2017), (Baldi and Hornik, 1989), (Goodfellow et al., 2016), (Kawaguchi, 2016)), Luo & Wu (Wu et al., 2018) prove that there is no spurious local minima in a two-hidden-unit ReLU network. However, Luo & Wu make an assumption that the 2^{nd} layer is fixed, which does not hold in applications. Li & Yuan (Li and Yuan, 2017) also make progress on understanding algorithms by providing a convergence analysis for SGD on special two-layer feedforward networks with ReLU activations, yet, they specify the 1^{st} layer as begin offset by "identity mapping" (mimicking residual connections) and the 2^{nd} layer as the ℓ_1-norm function. Additionally, based on their work (Du et al., 2018b), Du et al (Du et al., 2018a) give the 2^{nd} layer more freedom in the problem of

learning a two-layer neural network with a non-overlapping convolutional layer and ReLU activation. They prove that although there is a spurious local minimizer, gradient descent with weight normalization can still recover the true parameters with constant probability when given Gaussian inputs. Nevertheless, the convergence is guaranteed when the 1^{st} layer is a convolutional layer.

Online convex learning: Many successful algorithms and associated proofs have been studied and provided over the past few years to minimize regret in online learning setting. Zinkevich (Zinkevich, 2003) shows that the online gradient descent algorithm achieves regret $\mathcal{O}(\sqrt{T})$, for an arbitrary sequence of T convex loss functions (of bounded gradients) and given a diminishing learning rate. Then, Hazan et al (Hazan et al., 2007) improve regret to $\mathcal{O}(\log(T))$ when given strictly convex functions and a diminishing learning rate. The idea of adapting first order optimization methods is by no means new and is also popular in online convex learning. Duchi, Hazan & Singer (Duchi et al., 2011) present ADAGRAD, which employs very low learning rates for frequently occurring features and high learning rates for infrequent features, and obtain a comparable bound by assuming 1-strongly convex proximal functions. In a similar framework, Zhu & Xu (Zhu and Xu, 2015) extend the celebrated online gradient descent algorithm to Hilbert spaces (function spaces) and analyzed the convergence guarantee of the algorithm. The online functional gradient algorithm they propose also achieves regret $\mathcal{O}(\sqrt{T})$ when given convex loss functions. In all these algorithms, the loss function is required to be convex or strongly convex and the learning rate or step size must diminish. However, no work about regret analyses of online learning applied on deep neural networks (non-convex loss functions) has been done.

Adaptive regret: Recently, adaptive regret has been studied in the setting of prediction with expert advice (PEA) in online learning. Adaptive regret measures the maximum difference of

the performances of an online algorithm and the offline optimum for any consecutive τ samples in total T rounds, while our regret measures the maximum difference of the performances in the whole history. Existing online algorithms are closely related in the sense that adaptive algorithms designed are usually built upon the PEA algorithms. The concept of adaptive regret is formally introduced by Hazan and Seshadhri (Hazan and Seshadhri, 2007). They also propose a new algorithm named follow the leading history (FLH), which contains an expert-algorithm, a set of intervals and a meta-algorithm. Then Daniely in (Daniely et al., 2015) extends this idea by introducing strongly adaptive algorithms, which provide a regret bound $\mathcal{O}(\log(s+1)\sqrt{|I|}$ where s is the end point of the rolling window and $|I|$ is the size of the window. Later on, Zhang in (Zhang et al., 2019) applies the concept of the adaptive learning rate to adaptive regret and proposes adaptive algorithms for convex and smooth functions, and finally obtains a regret bound $\mathcal{O}(\sqrt{(\sum_{t=r}^{s} f_t(\omega))\log(s)\log(s-r)})$, where $f_t(\omega)$ is the loss function for the *t'th* sample given any ω in the corresponding domain, and r and s are the starting and ending data points of the interested sequence. Notice that in conjunction with infinite streaming, s blows up and eventually dominates the regret bound. Although the concept of the adaptive regret is similar to our regret with rolling window, adaptive regret relies on other existing online algorithms which not only use diminishing learning rate but also bring extra error. In regret with rolling window, we consider these two aspects together (infinite time stream and the issue of learning rates) and propose a new family of online algorithms which use a constant learning rate and achieve a more robust regret. Specifically, our regret bound is $\mathcal{O}(\sqrt{T})$, which does not depend of the position of the window.

2.3. Regret with Rolling Window

We consider the problem of optimizing regret with rolling window, inspired by the standard regret ((Zinkevich, 2003), (Abernethy et al., 2012), (Rakhlin and Tewari, 2009)). The problem with the traditional regret is that it captures the performance of an algorithm only over a fixed number of samples or loss functions. In most applications data is continuously streamed with an infinite number of future loss functions. The performance over any finite number of consecutive loss functions T is of interest. The concept of regret is to compare the optimal offline algorithm with access to T contiguous loss functions with the performance of the underlying online algorithm. Regret with rolling window is to find the maximum of all differences between the online loss and the loss of the offline algorithm for any T contiguous samples. More precisely, for an infinite sequence $\{z^t, y^t\}_{t=1}^{\infty}$, where each feature vector $z^t \in \mathbb{R}^d$ is associated with the corresponding label y^t, given fixed T and any p, we first define $\omega_p^* \in \operatorname{argmin}_\omega \sum_{t=p}^{p+T} f_t(\omega)$, which corresponds to the optimal solution of the offline algorithm. In general, $f_t(\omega) = \text{loss}(x^t, y^t; \omega)$, e.g. $f_t(\omega) = \left\|\omega^T x^t - y^t\right\|^2$ if the linear regression model is applied and the mean square error is used. Then, we consider

$$\max_{p \in \mathbb{N}} R_p(T) := \sum_{t=p}^{T+p} l_t(\omega_t) \tag{2.1}$$

with $l_t(\omega_t) = f_t(\omega_t) - f_t(\omega_p^*)$, where f_t is a function of sample z^t. The regret with rolling window metric captures regret over every T consecutive loss functions and it is aiming to assess the worst possible regret over every such sequence. Note that if we have only T loss functions corresponding only to $p = 1$, then this is the standard regret definition in online learning. The goal is to develop algorithms with low regret with rolling window. We prove that regret with

rolling window can be bounded by $\mathcal{O}(\sqrt{T})$. In other words, the average regret with rolling window approaches zero.

2.4. Convex Setting

In the convex setting, we propose two algorithms with a different learning rate or stepsize strategy and analyze them with respect to (2.1) in the streaming setting.

2.4.1. Algorithms

Algorithms in standard online setting are almost all based on gradient descent where the parameters are updated after each new loss function is received based on the gradient of the current loss function. A challenge is the strategy to select an appropriate learning rate. In order to guarantee good regret the learning rate is usually decaying. In the streaming setting, we point out that a decaying learning rate is improper since far away samples (very large p) would get a very small learning rate implying low consideration to such samples. In conclusion, the learning rate has to be a constant or it should follow a dynamically adaptive learning algorithm, i.e. ADAM. The algorithms we provide for solving (2.1) in the streaming setting are based on gradient descent and one of the just mentioned learning rate strategies.

In order to present our algorithms, we first need to specify notation and parameters. In each algorithm, we denote by η and g_t the learning rate or stepsize and a subgradient of loss function f_t associated with sample (z^t, y^t), respectively. Additionally, we employ $\odot$ to represent the element-wise multiplication between two vectors or matrices (Hadamard product). However, for other operations we do not introduce new notation, e.g., for element-wise division $(/)$ and

square root ($\sqrt{}$), since these two operations are written differently when representing standard matrix or vector operations.

We start with OGD which mimics gradient descent in online setting and achieves $\mathcal{O}(\sqrt{T})$ regret with rolling window when given constant learning rate. Algorithm 2 is a twist on Zinkevich's ONLINE GRADIENT DESCENT (Zinkevich, 2003). OGD updates its weight when a new sample is received in step 4. In addition, OGD uses constant learning rate in the streaming setting so as to efficiently and dynamically learn the geometry of the dataset. Otherwise, if a diminishing learning rate is applied, OGD misses informative samples which arrive late due to the extremely small learning rate and leads to $\mathcal{O}(T)$ regret with rolling window (this is trivial to observe if the loss functions are bounded). Regret of $\mathcal{O}(\sqrt{T})$ is achieved in the streaming setting if learning rate $\eta = 1/\sqrt{T}$.

Constant learning rates have a drawback by treating all features equally. Consequently, we adapt ADAM to online setting and further extend it to streaming. Algorithm 3, inspired by ADAM (Kingma and Ba, 2015) and AMSGRAD (Reddi et al., 2018), has regret with rolling window also of the order $\mathcal{O}(\sqrt{T})$ given constant stepsize η as shown in the next section. The key difference of CONVGADAM with AMSGRAD is that it maintains the same ratio of the past gradients and the current gradient instead of putting more and more weight on the current gradient and losing the memory of the past gradients fast. In Algorithm 3, CONVGADAM records exponential moving average of gradients and moments in step 5 and 6. Step 7 guarantees that $\hat{v}_t$ is the maximum of all v_t until the present time step. Then, step 8 gives the adaptive update rule by using the maximum value of v_t to normalize the running average of the gradient at time t. Besides, constant stepsize η is crucial to make CONVGADAM well-performed due to the aforementioned reason with a potential decaying learning rate or stepsize.

In step 8, we observe that $\hat{v}_{ti} = 0$ for a feature i implies $m_{ti} = 0$, therefore, we retain the foregoing weight ω_{ti} as the succeeding weight $\omega_{t+1,i}$. In other words, in this case we define $\frac{\eta}{\sqrt{\hat{v}_{ti}}} \cdot m_{ti} = \frac{\eta}{\sqrt{0}} \cdot 0 = 0$.

Algorithm 2 ONLINE GRADIENT DESCENT

1: *Positive parameter* η
2: **for** $t = 0, 1, 2, \cdots$ **do**
3: $g_t = \bigtriangledown f_t(w_t)$
4: $w_{t+1} = w_t - \eta g_t$
5: **end for**

Algorithm 3 CONVERGENT ADAM

1: *Positive parameters* $\eta, \beta_1 < 1, \beta_2 < 1$
2: *Set* $m_0 = 0, v_0 = 0,$ and $\hat{v}_0 = 0$
3: **for** $t = 0, 1, 2, \cdots,$ **do**
4: $g_t = \bigtriangledown f_t(w_t)$
5: $m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t$
6: $v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t \odot g_t$
7: $\hat{v}_t = \max(\hat{v}_{t-1}, v_t)$
8: $w_{t+1} = w_t - \frac{\eta}{\sqrt{\hat{v}_t}} \odot m_t$
9: **end for**

2.4.2. Analyses

In this section, we provide regret analyses of OGD and CONVGADAM showing that both of them attain regret with rolling window which is proportional to the square root of the size of the rolling window given a constant learning rate or stepsize in the streaming setting. For inner (scalar) products, given the fact that $\langle a, b \rangle = a^T b = b^T a$ for two vectors a and b, in the rest of the paper, for short expressions we use $a^T b$, but for longer we use $\langle a, b \rangle$.

We require the following standard assumptions.

Assumption 1:

(1) There exists a constant D_∞, such that $\|\omega_t\|_\infty \leq \frac{D_\infty}{2}$, for any $t \in \mathbb{N}$.

(2) The loss gradients $\nabla f_t(\omega_t)$ are bounded, i.e., for all ω_t such that $\|\omega_t\|_\infty \leq \frac{D_\infty}{2}$, we have $\|\nabla f_t(\omega_t)\|_\infty \leq G_\infty$.

(3) Functions $f_t(\omega)$ are convex and differentiable with respect to ω for every $t \in \mathbb{N}$.

(4) Functions $f_t(\omega)$ are strongly convex with parameter H, i.e., for all $\bar{\omega}$ and $\tilde{\omega}$, and for $t \in \mathbb{N}$, it holds $f_t(\bar{\omega}) + \bigtriangledown f_t(\bar{\omega})(\tilde{\omega} - \bar{\omega}) + \frac{H}{2}\|\tilde{\omega} - \bar{\omega}\|^2 \leq f_t(\tilde{\omega})$.

The first condition in Assumption 1 assumes that ω_t are bounded. This assumption can be removed by further complicating certain aspects of the upcoming proofs. This extension is discussed in Appendix B.1.1 for the sake of clarity of the algorithm. In 2 from Assumption 1, the gradient of the loss function is requested to be upper bounded. Notice that each loss function f_t is enforced to be differentiable and convex in 3, whereas f_t is required to be strictly convex with parameter H in 4. All these are standard assumptions.

We first provide the regret analysis of OGD.

Theorem 7. *If 1-3 in Assumption 1 hold, and $\eta = \frac{\eta_1}{\sqrt{T}}$ for any positive constant η_1, the sequence ω_t generated by* OGD *achieves* $\max_{p\in\mathbb{N}} R_p(T) \leq \mathcal{O}(\sqrt{T})$.

The proof is provided in Appendix B.2. Under the assumptions in Assumption 1, by finding a relationship for the sequence of the weight error $\omega_t - \omega_p^*$ and employing the property of convexity from condition 3 from Assumption 1, we prove that OGD obtains the regret with rolling window which is proportional to the square root of the size of the rolling window when given the constant learning rate. This is consistent with the regret of OGD in the standard online setting.

The analysis of OGD is not totally new but still has some important differences. Also, when using a diminishing learning rate, given strongly convex f_t, it has been proven in (Hazan et al.,

2007) by Hazan that OGD obtains logarithmic regret. However, this is not possible even given strongly convex f_t when using a constant learning rate, which should be clear after reading our regret analysis of OGD and comparing it with the regret analysis in (Hazan et al., 2007).

If OGD is an analogue to the Gradient Descent optimization method for the online setting, then CONVGADAM is an online analogue of ADAM, which dynamically incorporates knowledge of the characteristics of the dataset received in earlier iterations to perform more informative gradient-based learning. Next, we show that CONVGADAM achieves the same regret with rolling window given a constant stepsize.

Theorem 8. *If Assumption 1 holds, and β_1 and β_2 are two constants between 0 and 1 such that $\lambda := \frac{\beta_1}{\sqrt{\beta_2}} < 1$ and $\beta_1 \leq \frac{H\eta}{1+H\eta}$, then for $\eta = \frac{\eta_1}{\sqrt{T}}$ for any positive constant η_1, the sequence ω_t generated by* CONVGADAM *achieves* $\max_{p\in\mathbb{N}} R_p(T) \leq \mathcal{O}(\sqrt{T})$.

The proof is provided in Appendix B.3. The very technical proof follows the following steps. Based on the updating procedure in steps 4-8, we establish a relationship for the sequence of the weight error $\omega_t - \omega_p^*$. Meanwhile, considering condition 4 in Assumption 1, we obtain another relationship between the loss function error $f_t(\omega_t) - f_t(\omega_p^*)$ and $\langle \omega_t - \omega_*, \bigtriangledown f_t(\omega_t)\rangle$. Assembling these two relationships provide a relationship between the weight error $\omega_t - \omega_*$ and the loss function error $f_t(\omega_t) - f_t(\omega_*)$. By deriving upper bounds for all of the remaining terms based on conditions from Assumption 1, we are able to argue the same regret with rolling window of $\mathcal{O}(\sqrt{T})$.

In the regret analysis of AMSGRAD (Reddi et al., 2018), the authors forget that the stepsize is $\frac{1}{\sqrt{t}}$ and take the hyperparameter η to be exponentially decaying for granted without assumptions which eventually leads to $\mathcal{O}(T\sqrt{T})$ regret in standard online setting. Our analysis is

flexible enough to extend to AMSGRAD and a slight change to our proof yields the $\mathcal{O}(\sqrt{T})$ regret for AMSGRAD. The changes in our proof to accommodate standard online setting and AMSGRAD are stated in Appendix B.1.2. Moreover, the proof of convergence of AMSGRAD in (Reddi et al., 2018) uses a diminishing stepsize while our proof is valid for both constant and diminishing stepsizes. Likewise, for ADABOUND (Luo et al., 2019), the right scale of the stepsize is also missed and the regret should be $\mathcal{O}(T)$, which is discussed in more detail in Appendix B.1.2. In this section we also discuss how to amend our proof to provide the $\mathcal{O}(\sqrt{T})$ regret bound in standard online setting for ADABOUND.

Theorem 8 guarantees that CONVGADAM achieves the same regret with rolling window as OGD for convex loss functions. On the other hand, very limited work has been done about regret for nonconvex loss functions, e.g. the loss function of a two-layer ReLU neural network. In the following section, we argue that both DNNGD and DNNADAM attain the same regret with rolling window if the initial starting point is close to an optimal offline solution and by using a constant learning rate or stepsize. In addition to a favorable starting point, further assumptions are needed.

2.5. Two-Layer ReLU Neural Network

In this section we consider a two layer neural network with the first hidden layer having an arbitrary number of neurons and the second hidden layer has a single neuron. The underlying activation function is a probabilistic version of ReLU and minimum square error is considered as the loss function. First of all, the optimization problem of such a two-layer ReLU neural network is neither convex nor convex (and clearly non linear), therefore, it is very hard to find

a global minimizer. Instead, we show that our algorithms achieve $\mathcal{O}(\sqrt{T})$ regret with rolling window when the initial point is close enough to an optimal solution.

Neural networks as classifiers have a lot of success in practice, whereas a formal theoretical understanding of the mechanism of why they work is largely missing. Studying a general neural network is challenging, therefore, we focus on the proposed two-layer ReLU neural network. For a dataset $\{z^t, y^t\}_{t=1}^{\infty}$, the standard loss function of the two-layer neural network is $f_t\left(\omega_{1,t}, \omega_{2,t}\right) = \frac{\left\|\omega_{1,t}^T \sigma\left(\omega_{2,t} z^t\right) - y^t\right\|^2}{2}$, where σ represents the ReLU activation function applied element-wise, $\omega_{1,t}$ is the parameter vector, and $\omega_{2,t}$ is the parameter matrix. It turns out that ReLU is challenging to analyze since nesting them yields many combinations of the various values being below zeros. One way to get around this is to consider a probabilistic version of ReLU and capturing expected loss, Luo & Wu (Wu et al., 2018).

To this end we treat ReLU as a random Bernoulli variable in the sense that $\Pr(\sigma(x) = x) = \rho$, $\Pr(\sigma(x) = 0) = 1 - \rho$. Luo & Wu (Wu et al., 2018) in the standard offline setting analyze $f_t\left(\omega_{1,t}, \omega_{2,t}\right)$ for the probabilistic version of ReLU. For our online analyses we need to slightly alter the setting by introducing two independent identically distributed random variables σ_1 and σ_2 and the loss function as follows

$$f_t(\omega_{1,t}, \omega_{2,t}) = \frac{\left(\omega_{1,t}^T \sigma_1\left(\omega_{2,t} z^t\right) - y^t\right)\left(\omega_{1,t}^T \sigma_2\left(\omega_{2,t} z^t\right) - y^t\right)}{2}. \tag{2.2}$$

There is a crucial property of f_t which is positive-homogeneity. That is, for any $c > 0$, $f_t(c\omega_1, \frac{\omega_2}{c}) = f_t(\omega_1, \omega_2)$. This property allows the network to be rescaled without changing the function computed by the network.

For two-layer ReLU neural network, given $(\omega_{1,*}^p, \omega_{2,*}^p) \in \operatorname{argmin}_\omega \sum_{t=p}^{p+T} \mathbb{E}_{\sigma_1,\sigma_2} f_t(\omega_{1,t}, \omega_{2,t})$, we consider regret with rolling window as

$$\max_{p\in\mathbb{N}} \min_{\substack{(\omega_{1,t})_{t\in\mathbb{N}},(\omega_{2,t})_{t\in\mathbb{N}} \\ \|\omega_{1,t}\|=1}} R_p(T) := \sum_{t=p}^{T+p} \mathbb{E}_{\sigma_1,\sigma_2} \left[l_t \left(\omega_{1,t}, \omega_{2,t} \right) \right]. \tag{2.3}$$

Next, we propose two algorithms with different learning rates or stepsizes for the two-layer neural network and analyze them with respect to (2.3).

2.5.1. Algorithms

In order to present the algorithms, let us first introduce notation and parameters. For any matrix A and vector x, let $[A]_{ij}$ and $[x]_i$ denote the element in the i_{th} row and j_{th} column of matrix A and i_{th} coordinate of vector x, respectively. Similarly, we use $[A]_{\cdot j}$ ($[A]_{i\cdot}$) to represent the j_{th} column (i_{th} row) of matrix A. Next, in order to be consistent, we also denote η and $g_{1,t}, g_{2,t}$ as the learning rate or stepsize and a subgradient of loss function f_t associated with sample (z^t, y^t), respectively. Let ξ_1 and ξ_2 be constants. Lastly, in order to be consistent with the notation in the convex setting, we employ $\odot$ to represent the element-wise multiplication between vectors or matrices while using standard division and square root notation for the corresponding operations element-wise in vectors and matrices.

We start with DNNGD which is the algorithm with a fixed learning rate, Algorithm 4. We show later that its regret with rolling window is $\mathcal{O}(\sqrt{T})$. DNNGD is an analogue of the gradient descent optimization method for the online setting with the two-layer ReLU neural network, and at the same time it is an extension of OGD. Different from OGD, DNNGD not only modifies weights at a given iteration by following the gradient direction, but it also rescales weights

based on the domain constraint in step 6, i.e. $\omega_{1,t}$ has a fixed norm. Then, $\omega_{2,t}$ is rescaled at the same time to impose positive-homogeneity in step 7.

Algorithm 4 DEEP NN GRADIENT DESCENT

1: *Positive parameter* $\eta > 0$

2: **for** $t = 0, 1, 2, \cdots$ **do**

3: Sample $\sigma_1,\ \sigma_2$

4: $g_{1,t} = \frac{1}{2}\left(\omega_{1,t}^T \sigma_1\left(\omega_{2,t} z^t\right) - y^t\right)\sigma_2\left(\omega_{2,t} z^t\right) + \frac{1}{2}\left(\omega_{1,t}^T \sigma_2\left(\omega_{2,t} z^t\right) - y^t\right)\sigma_1\left(\omega_{2,t} z^t\right)$

5: $g_{2,t} = \frac{1}{2}\left(\omega_{1,t}^T \sigma_1\left(\omega_{2,t} z^t\right) - y^t\right)\omega_{1,t}\left(\sigma_2\left(z^t\right)\right)^T + \frac{1}{2}\left(\omega_{1,t}^T \sigma_2\left(\omega_{2,t} z^t\right) - y^t\right)\omega_{1,t}\left(\sigma_1\left(z^t\right)\right)^T$

6: $\omega_{1,t+1} = \frac{\omega_{1,t} - \eta g_{1,t}}{\|\omega_{1,t} - \eta g_{1,t}\| / \sqrt{\frac{1}{2} + \xi_1}}$

7: $\omega_{2,t+1} = \left(\omega_{2,t} - \eta g_{2,t}\right) \cdot \left[\|\omega_{1,t} - \eta g_{1,t}\| \,/\, \sqrt{\frac{1}{2} + \xi_1}\right]$

8: **end for**

Taking the drawbacks of a constant learning rate into consideration, we propose Algorithm 5, which is an extension of CONVGADAM for the two-layer ReLU neural network and likewise attains $\mathcal{O}(\sqrt{T})$ regret with rolling window. In DNNADAM, the stochastic gradients computed in steps 4 and 5 are different than those in DNNGD. This is due to challenges in establishing the regret bound. Nevertheless, the stochastic gradients are unbiased estimators of gradients of the loss function. An alternative is to have four samples, two per gradient group. This would also enable the regret analysis, however we only employ two of them so as to reduce the variance of the algorithm. DNNADAM records exponential moving average of gradients and moments in steps 6 - 9. Step 10 modifies $v_{2,t}$ to be a matrix with same value in the same column. This is a divergence from standard ADAM which does not have this requirement. The modification is required for the regret analysis. Then, steps 11 and 12 guarantee that $\{\hat{v}_{1,t}\}$ and $\{\hat{v}_{2,t}\}$ are nondecreasing sequences element-wise. Lastly, we update weights and also perform

the rescaling modification to DNNADAM in steps 13 and 14. Additionally, we apply the same strategy as we mention in CONVGADAM when $[\hat{v}_{1,t}]_i = 0$ or $[\hat{v}_{2,t}]_{i,j} = 0$. More precisely, if $[\hat{v}_{1,t}]_i = 0$ ($[\hat{v}_{2,t}]_{ij} = 0$), it implies $[g_{1,k}]_i = 0$ ($[g_{2,k}]_{ij} = 0$) for all k, which in turn yields $[m_{1,t}]_i = 0$ ($[m_{2,t}]_{ij} = 0$). Thus, we define $\left[m_{1,t}/\sqrt{\hat{v}_{1,t}}\right]_i = \frac{0}{0} = 0$ and $\left[m_{2,t}/\sqrt{\hat{v}_{2,t}}\right]_{ij} = \frac{0}{0} = 0$. Therefore, we maintain the weights from the last iteration.

Algorithm 5 DEEP NN ADAM

1: *Positive parameters* $\eta, \epsilon_1, \epsilon_2, \beta_{11t} \leq 1, \beta_{12t} \leq 1, \beta_{21} \leq 1, \beta_{22} \leq 1$

2: **for** $t = 0, 1, 2, \cdots$ **do**

3: Sample $\sigma_1,\ \sigma_2$

4: $g_{1,t} = \left(\omega_{1,t}^T \sigma_1\left(\omega_{2,t} z^t\right) - y^t\right) \sigma_2\left(\omega_{2,t} z^t\right)$

5: $g_{2,t} = \left(\omega_{1,t}^T \sigma_1\left(\omega_{2,t} z^t\right) - y^t\right) \omega_{1,t} \left(\sigma_2\left(z^t\right)\right)^T$

6: $m_{1,t} = \beta_{11t} m_{1,t-1} + \left(1 - \beta_{11t}\right) g_{1,t}$

7: $m_{2,t} = \beta_{12t} m_{2,t-1} + \left(1 - \beta_{12t}\right) g_{2,t}$

8: $v_{1,t} = \beta_{21} v_{1,t-1} + \left(1 - \beta_{21}\right) g_{1,t} \odot g_{1,t}$

9: $\dot{v}_{2,t} = \beta_{22} \dot{v}_{2,t-1} + \left(1 - \beta_{22}\right) g_{2,t} \odot g_{2,t}$

10: $[v_{2,t}]_{ij} = \max_k \left|[\dot{v}_{2,t}]_{kj}\right|$

11: $\hat{v}_{1t} = \max\left(v_{1t}, \hat{v}_{1,t-1}\right)$

12: $\hat{v}_{2t} = \max\left(v_{2t}, \hat{v}_{2,t-1}\right)$

13: $\omega_{1,t+1} = \frac{\omega_{1t} - \frac{\eta}{\sqrt{\hat{v}_{1t}}} \odot m_{1t}}{\left\|\omega_{1t} - \frac{\eta}{\sqrt{\hat{v}_{1t}}} \odot m_{1t}\right\|} \cdot \sqrt{\frac{\left[\frac{1}{2} + \xi_2\right]}{\left(1 - \beta_{121}\right)}}$

14: $\omega_{2,t+1} = \left(\omega_{2,t} - \frac{\eta}{\sqrt{\hat{v}_{2,t}}} \odot m_{2t}\right) \cdot \left\|\omega_{1t} - \frac{\eta}{\sqrt{\hat{v}_{1t}}} \odot m_{1t}\right\| / \sqrt{\frac{\left[\frac{1}{2} + \xi_2\right]}{\left(1 - \beta_{121}\right)}}$

15: **end for**

2.5.2. Analyses

In this section, we discuss regret with rolling window bounds of DNNGD and DNNADAM showing that both of them attain regret with rolling window proportional to the square root of the size of the rolling window. Before establishing the regret bounds, we first require the following assumptions.

Assumption 2:

(1) Activations σ_1, σ_2 are independent Bernoulli random variables with the same probability ρ of success, i.e. Pr$(\sigma(x) = x) = \rho$, Pr$(\sigma(x) = 0) = 1 - \rho$.

(2) There exists $\omega_{1,*}$ and $\omega_{2,*}$ such that $\mathbb{E}\left[\omega_{1,*}^T \sigma_1 \left(\omega_{2,*} z^t\right)\right] = \rho \omega_{1,*}^T \omega_{2,*} z^t = y^t$ for all t.

(3) Quantities $\omega_{1,t}, \omega_{2,t}$, z^t and y^t are all bounded for any t. In particular, let $\left\|\omega_{2,t}\right\|_F \leq \alpha$ and $\left|\left[g_{2,t}\right]_{ij}\right| \leq G_{2,\infty}$ for any t, i, j.

(4) There exists $0 < \epsilon < \pi/2$ such that $\frac{\left|\left\langle \omega_{1,t}^T \omega_{2,t} - \omega_{1,*}^T \omega_{2,*}, z^t \right\rangle\right|}{\left\|\omega_{1,t}^T \omega_{2,t} - \omega_{1,*}^T \omega_{2,*}\right\| \|z^t\|} \geq \cos(\epsilon)$ for all t when $\left(\omega_{1,t}^T \omega_{2,t} - \omega_{1,*}^T \omega_{2,*}\right)^T z^t \neq 0$.

(5) There exits a positive constant μ such that $\mu \leq \min_{i,t,[\hat{v}_{1,t}]_i \neq 0} \left|\left[\hat{v}_{1,t}\right]_i\right|$.

As Kawaguchi assumed in (Kawaguchi, 2016) and other works ((Dauphin et al., 2014), (Choromanska et al., 2015a), (Choromanska et al., 2015b)), we also assume that σ's are Bernoulli random variables with the same probability of success and are independent from input z^t's and

weights ω's in 1 from Assumption 2. [1] Condition in 2 from Assumption 2 states that the optimal expected loss is zero. This is also assumed in other prior work in offline, e.g. (Wu et al., 2018), (Du et al., 2018a). The 3^{rd} condition in Assumption 2 is an extension of 1 in Assumption 1. Likewise, the constraints on $\omega_{1,t}$ and $\omega_{2,t}$ can be removed by further introducing technique discussed in Appendix B.1.1, and consequently, $g_{1,t}$ and $g_{2,t}$ are bounded due to steps 4 and 5. The next to the last condition in Assumption 2 requires that a new coming sample has to be beneficial to improve current weights. More precisely, we interpret the difference between the current weights and optimal weights as an error that needs to be corrected. Then, a new sample which is not relevant to the error vector is not allowed. In other words, we assume that the algorithm does not receive any uninformative samples. Condition 5 from Assumption 2 assumes that any nonzero $|[\hat{v}_{1,t}]_i|$ is lower bounded by a constant μ for all t and i. It is a weak constraint since $[\hat{v}_{1,t}]_i \geq [\hat{v}_{1,t-1}]_i$ for any t and i. In practice, we can modify it by only memorizing the first nonzero value in each coordinate and finding the smallest among these values. Otherwise, if all of $[\hat{v}_{1,t}]_i = 0$, then we can set $\mu = 1$ by default.

The regret statement for DNNGD is as follows.

Theorem 9. *If 1-4 in Assumption 2 hold, $\xi_1 = \frac{\alpha}{\cos(\epsilon)}$, and $\eta = \frac{\eta_1}{\sqrt{T}}$ for any positive constant η_1, the sequence $\omega_{1,t}$ and $\omega_{2,t}$ generated by* DNNGD *for a 2-layer ReLU neural network achieves* $\max_{p\in\mathbb{N}} \mathbb{E}\left[R_p(T)\right] \leq \mathcal{O}(\sqrt{T})$.

[1] In general, the distribution of the Bernoulli random variable representing the ReLU activation function is not required to be stationary for all t. Since all loss functions are considered separately, we only need to assume that for every z^t, there is a corresponding ρ^t such that $\mathrm{E}\left[\omega_{1,*}^T\omega_{2,*}z^t\right] = \rho^t\omega_{1,*}^T\omega_{2,*}z^t = y^t$, then, later in the proof, those ρ^t's are absorbed into $\mathrm{E}\left[l_t(\omega_{1,t},\omega_{2,t}|\mathrm{F}^t\right]$. Therefore, the algorithms can dynamically adapt to the new patterns in the dataset. In the proof, we simplify this process by using a constant ρ. Then, given $\sigma, \sigma_1, \sigma_2$ are i.i.d, $\mathrm{E}_\sigma\left[\left\|\omega_{1,t}^T\sigma\left(\omega_{2,t}z^t\right) - y^t\right\|^2/2\right] = \mathrm{E}_\sigma\left[\left\|\omega_{1,t}^T\sigma\left(\omega_{2,t}z^t\right) - \omega_{1,*}^T\sigma\left(\omega_{2,*}z^t\right)\right\|^2/2\right] = \frac{\rho}{2}\left(\omega_{1,t}^T\omega_{2,t}z^t - \omega_{1,*}^T\omega_{2,*}z^t\right)^2$. At the same time, the new loss function is $\mathrm{E}_{\sigma_1,\sigma_2}\left[\left(\omega_{1,t}^T\sigma_1\left(\omega_{2,t}z^t\right) - y^t\right)\left(\omega_{1,t}^T\sigma_2\left(\omega_{2,t}z^t\right) - y^t\right)/2\right] = \mathrm{E}_{\sigma_1}\left[\omega_{1,t}^T\sigma_1\left(\omega_{2,t}z^t\right) - y^t\right]\mathrm{E}_{\sigma_2}\left[\omega_{1,t}^T\sigma_2\left(\omega_{2,t}z^t\right) - y^t\right]/2 = \frac{\rho^2}{2}\left(\omega_{1,t}^T\omega_{2,t}z^t - \omega_{1,*}^T\omega_{2,*}z^t\right)^2$. Therefore, minimizing our new loss function is the same as minimizing the original loss given that ρ is a positive constant.

The proof is in Appendix B.5. Based on the fact that the loss function is nonconvex, i.e., we no longer have a direct relationship between the loss function error $f_t(\omega_{1,t}, \omega_{2,t}) - f_t(\omega_{1,*}, \omega_{2,*})$ and $\left\langle \omega_{1,t}^T \omega_{2,t} - \omega_{1,*}^T \omega_{2,*}, \omega_{1,t}^T \bigtriangledown_{\omega_2} f_t(\omega_{1,t}, \omega_{2,t}) + \left(\bigtriangledown_{\omega_1} f_t(\omega_{1,t}, \omega_{2,t})\right)^T \omega_{2,t} \right\rangle$, any technique that relies on the property of convexity is inappropriate. The main challenges are coming from building a bridge between the loss function error $f_t(\omega_{1,t}^T \omega_{2,t}) - f_t(\omega_{1,*} \omega_{2,*})$ and the weight error $\omega_{1,t}^T \omega_{2,t} - \omega_{1,*}^T \omega_{2,*}$. To address this problem, we explore the difference between $\omega_{1,t+1}^T \omega_{2,t+1} - \omega_{1,*}^T \omega_{2,*}$ and $\omega_{1,t}^T \omega_{2,t} - \omega_{1,*}^T \omega_{2,*}$ in detail.

The steps to study the regret with rolling window are as follows. Based on steps 4 - 7, we expand $\omega_{1,t+1}^T \omega_{2,t+1} - \omega_{1,*}^T \omega_{2,*}$ to establish a relationship for the sequence of the weight error $\omega_{1,t}^T \omega_{2,t} - \omega_{1,*}^T \omega_{2,*}$. In association with explicit formulas of gradients and condition 4 in Assumption 2, we obtain the loss function error $f_t(\omega_{1,t}, \omega_{2,t}) - f_t(\omega_{1,*}, \omega_{2,*})$. Meanwhile, all of the remaining terms are bounded due to condition 3 from Assumption 2. Combined with the fact that $\omega_{1,t}$ has a constant norm, the regret with rolling window bound of DNNGD is achieved by applying the law of iterated expectation.

At the same time, our proof is flexible enough to extend to standard online setting. For a constant learning rate, Appendix B.5 provides the necessary details for the standard case. In summary, regret of $\mathcal{O}(\sqrt{T})$ is achieved. We note that such a result has only been known for the diminishing learning rate and thus we extend the prior knowledge by covering the constant learning rate case.

The adaptive learning setting algorithm DNNADAM has the same regret bound as stated in the following theorem.

Theorem 10. *If Assumption 2 holds,* $\eta = \frac{\eta_1}{\sqrt{T}}$ *for any positive constant* η_1, $\beta_{111}, \beta_{121}, \beta_{21}, \beta_{22}$ *are constants between* 0 *and* 1 *such that* $\lambda_1 := \frac{\beta_{111}}{\beta_{21}} \leq 1$ *and* $\lambda_2 := \frac{\beta_{121}}{\beta_{22}} \leq 1$, $\beta_{11t} = \beta_{111}\gamma_1^t$ *and*

$\beta_{12t} = \beta_{121}\gamma_2^t$ *with* $0 < \gamma_1, \gamma_2 < 1$*, and* $\xi_2 = \frac{\alpha G_{2,\infty}}{\mu \cos(\epsilon)}$*, then, the sequence* $\omega_{1,t}$ *and* $\omega_{2,t}$ *generated by* DNNADAM *for the 2-layer ReLU neural network achieves* $\max_{p\in\mathbb{N}} \mathbb{E}\left[R_p(T)\right] \leq \mathcal{O}(\sqrt{T})$.

The proof is in Appendix B.7. Similar to the difficulty faced in the proof of Theorem 9, we do not possess a relationship between the loss function error $f_t(\omega_{1,t}, \omega_{2,t}) - f_t(\omega_{1,*}, \omega_{2,*})$ and $\left\langle \omega_{1,t}^T\omega_{2,t} - \omega_{1,*}^T\omega_{2,*}, \omega_{1,t}^T \bigtriangledown_{\omega_2} f_t(\omega_{1,t}, \omega_{2,t}) + \left(\bigtriangledown_{\omega_1} f_t(\omega_{1,t}, \omega_{2,t})\right)^T \omega_{2,t} \right\rangle$. Even worse, the variance of the algorithm caused by merging all previous information and normalizing the stepsize makes the relationship between the loss function error $f_t(\omega_{1,t}, \omega_{2,t}) - f_t(\omega_{1,*}, \omega_{2,*})$ and the weight error $\omega_{1,t}^T\omega_{2,t} - \omega_{1,*}^T\omega_{2,*}$ more ambiguous. The way we deal with this is by treating $\frac{m_t}{\sqrt{v_t}}$ together as the gradient first and then extracting the effective gradient out from it and bounding the remaining terms.

The structure of the technical proof is similar to that of Theorem 9. We first establish a relationship for the sequence of the weight error $\omega_{1,t}^T\omega_{2,t} - \omega_{1,*}^T\omega_{2,*}$ by multiplying $\sqrt[4]{\hat{v}_{2,t}}$. Then, using the definitions of β's, λ's and γ's, we bound all the terms without the stepsize by constants except those which potentially can contribute to the loss function. To this end, we obtain a relationship between the weight error $\omega_{1,t}^T\omega_{2,t} - \omega_{1,*}^T\omega_{2,*}$ and the loss function. Finally, combined with step12 and the law of iterated expectation, we are able to argue $\mathcal{O}(\sqrt{T})$ regret with rolling window for DNNADAM.

Likewise, we are able to extend the proof of Theorem 10 to the standard online setting for DNNADAM. We do not need to make any change to establish $\mathcal{O}(\sqrt{T})$. For diminishing stepsize μ, a slight change to the proof is indeed. Details are provided in Appendix B.1.1.

2.6. Numerical Study

In this section, we compare the CONVGADAM method with OGD (Zinkevich, 2003) for solving problem (2.1) with a long sequence of data points (mimicking streaming). We conduct experiments on the MNIST8M dataset and two other different-size real datasets from the Yahoo! Research Alliance Webscope program. For all of these datasets, we train multi-class hinge loss support vector machines (SVM) (Shalev-Shwartz and Ben-David, 2014) and we assume that the samples are streamed one by one based on a certain random order. For all the figures provided in this section, the horizontal axis is in 10^5 scale. Moreover, we set $\beta_1 = 0.8$ and $\beta_2 = 0.81$ in CONVGADAM. We mostly capture the log of the loss function value which is defined as $\max_{p\in\mathbb{N}} \min_{(\omega_t)_{t\in\mathbb{N}}} \sum_{t=p}^{T+p} f_t(\omega_t)$.

2.6.1. Multiclass SVM with Yahoo! Targeting User Modeling Dataset

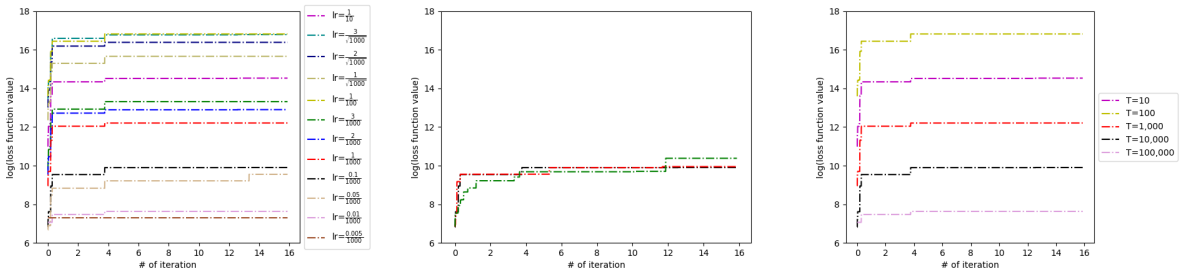
We first compare CONVGADAM with OGD using the Yahoo! user targeting and interest prediction dataset consisting of Yahoo user profiles[2]. It contains 1,589,113 samples (i.e., user profiles), represented by a total of 13,346 features and 380 different classification problems (called labels in the supporting documentation) each one with 3 classes.

First, we pick the first label out and conduct a sequence of experiments with respect to this label. The most important results are presented in Figure 2.2.1 for OGD and Figure 2.2.2 for CONVGADAM. In Figures 2.2.1(a) and 2.2.2(a), we consider the cases when the learning rate or step size varies from 0.1 to $5 \cdot 10^{-6}$ while keeping the order and T fixed at 1,000. Figures 2.2.1(b) and 2.2.2(b) provide the influence of the order of the sequence. Figures 2.2.1(c) and 2.2.2(c) represent the case where T varies from 10 to 10^5 with a fixed learning rate or step

[2]https://webscope.sandbox.yahoo.com/catalog.php?datatype=a

size. Lastly, in Figure 2.2.2(d), we compare the performance of CONVGADAM and OGD with certain learning rates and step sizes.

In these plots, we observe that CONVGADAM outperforms OGD for most of the learning rates and step sizes, and definitely for promissing choices. More precisely, in Figure 2.2.1(a) and 2.2.2(a), we discover that 0.1/1000 and $3/\sqrt{1000}$ are two high-quality learning rate and stepsize values which have relatively low error and are learning for OGD and CONVGADAM, respectively. Therefore, we apply those two learning rates for the remaining experiments on this dataset. In Figures 2.2.1(b) and 2.2.2(b), we observe that the perturbation caused by the change of the order is negligible especially when compared to the loss value, which is a positive characteristic. Thus, in the remaining experiments, we no longer need to consider the impact of the order of the sequence. From Figure 2.2.1(c) and Figure 2.2.1(d), we discover that the loss and T have a significantly positive correlation as we expect. Notice that changing T but fixing the learning rate or stepsize essentially means containing more samples in the regret, in other words, the regret for $T = 100$ is roughly 10 times the regret for $T = 10$. Since the pattern in the figures is preserved for the different T values for OGD and CONVGADAM, in the remaining experiments we fix T. In Figure 2.2.2(c), we discover that too big T or too small T causes poor performance and therefore, for the remaining experiments, we set $T = 1,000$ whenever T is fixed. From Figure 2.2.2(d), we observe that CONVGADAM outperforms OGD.



(a) (b) (c)

Figure 2.1. Comparison of OGD for different orders, learning rates and T

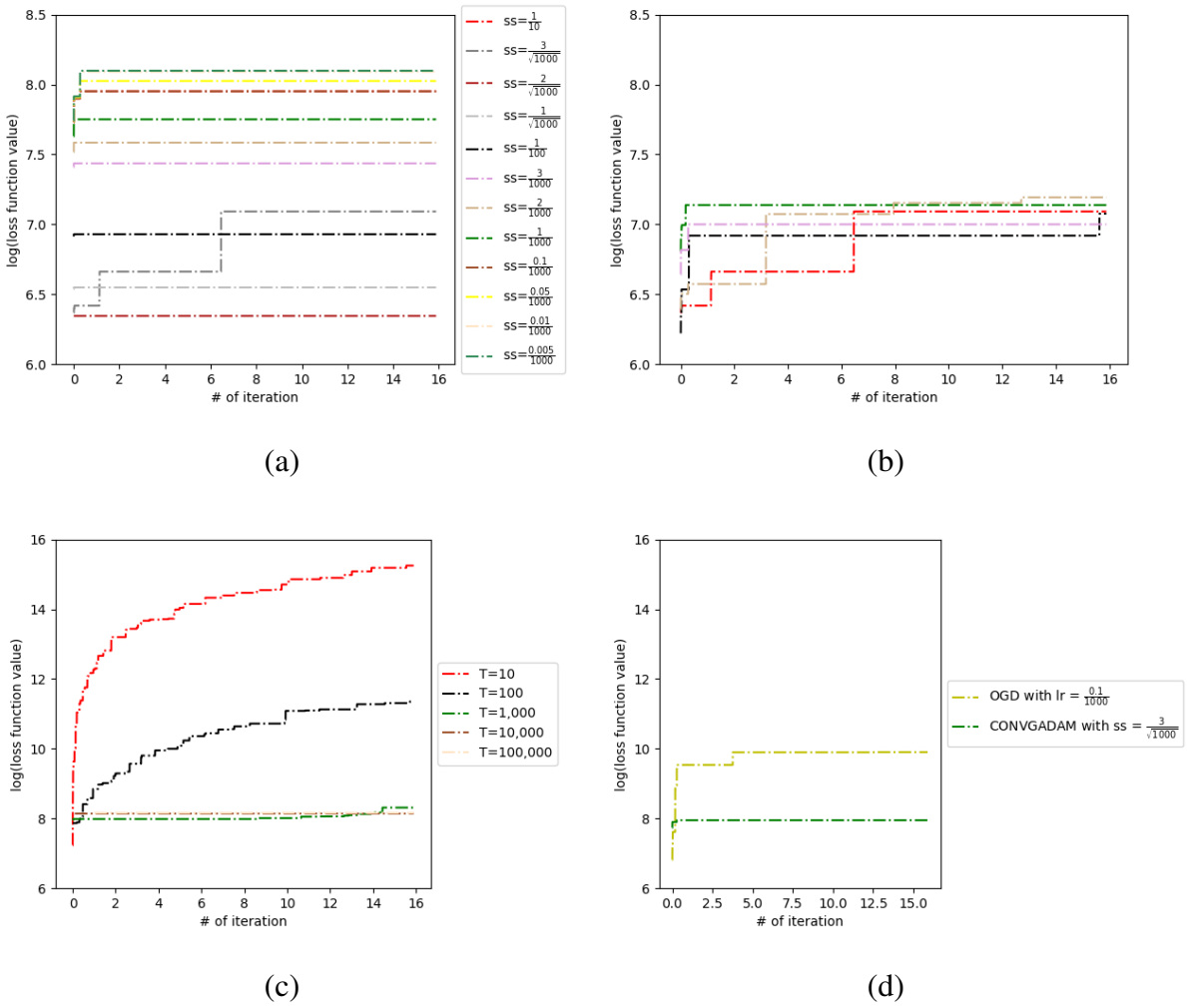
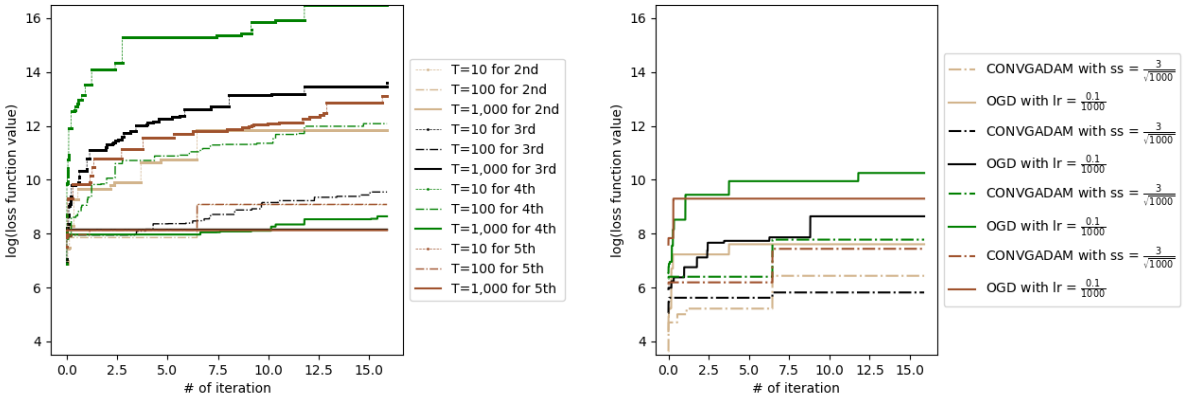


Figure 2.2. Comparison of CONVGADAM for different orders, stepsizes and T

After studying the algorithms on the first label, we test them on the next four labels. In Figure 2.2.3, we compare the performance of CONVGADAM for different T and the difference with OGD on the four labels. In these plots, we observe that $T = 1000$ provides a more stable and better performance than the other two values. Moreover, CONVGADAM outperforms OGD for all considered learning rates and step sizes.



(a) Comparison of CONVGADAM for different T

(b) Comparison of OGD and CONVGADAM

Figure 2.3. Performance of CONVGADAM and OGD on the remaining labels

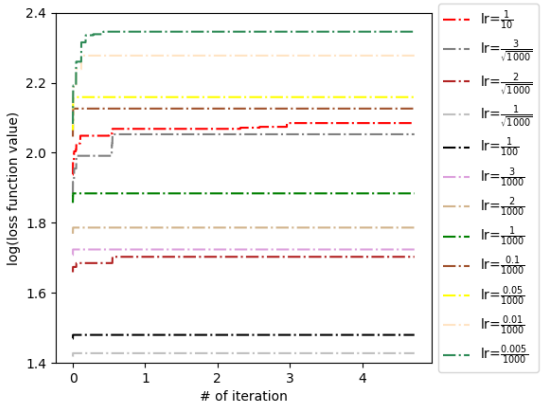
2.6.2. Multiclass SVM with Yahoo! Learn to Rank Challenge Dataset

In this set of experiments, we study the performances of CONVGADAM and OGD on Yahoo! Learn to Rank Challenge Dataset[3]. The dataset contains 473,134 samples, represented by a total of 700 features and 5 classes.

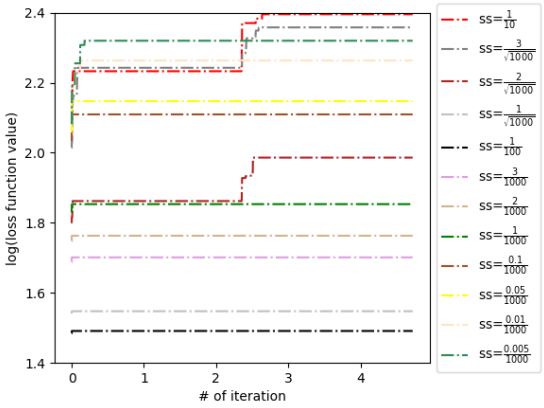
Figures 2.2.4(a) and 2.2.4(b) show the performances of OGD and CONVGADAM for different learning rates and stepsizes. Figure 2.2.4(c) provides the performance of CONVGADAM for different T. Lastly, Figure 2.2.4(d) compares the performance of CONVGADAM and OGD for a set of good learning rates but same T.

From Figures 2.2.4(a) and 2.2.4(b), we select the learning rate and stepsize $3/\sqrt{1000}$ and $2/\sqrt{1000}$ for CONVGADAM and OGD, respectively. From Figure 4(d), we discover the superior behavior of CONVGADAM over OGD as we expect.

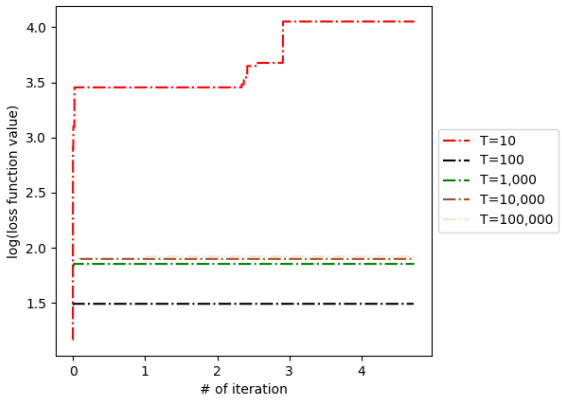
[3]https://webscope.sandbox.yahoo.com/catalog.php?datatype=c



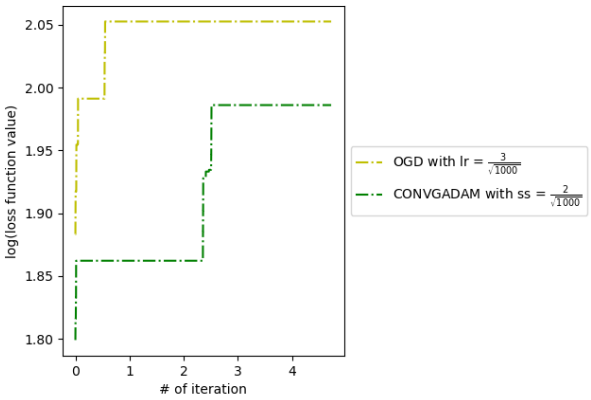
(a) Comparison of OGD for different learning rates



(b) Comparison of CONVGADAM for different stepsizes



(c) Comparison of CONVGADAM for different T



(d) Comparison of CONVGADAM and OGD

Figure 2.4. Performance of CONVGADAM on Learn to Rank Challenge dataset

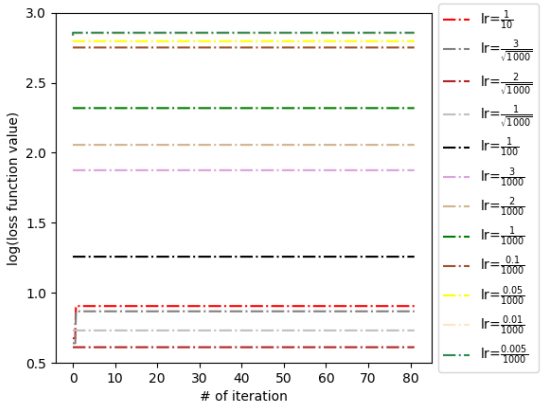
2.6.3. Multiclass SVM with MNIST8M Dataset

In this set of experiments, we study the performances of CONVGADAM and OGD on MNIST8M Dataset[4]. The dataset is generated on the fly by performing careful elastic deformation of the

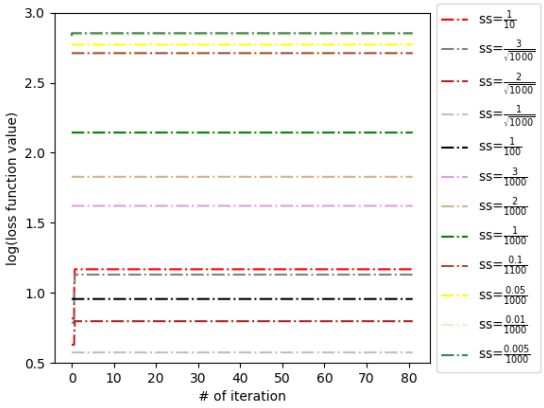
[4]https://www.csie.ntu.edu.tw/ cjlin/libsvmtools/datasets/

original MNIST training set. The dataset contains 8,100,000 samples, represented by a total of 784 features and 10 classes.

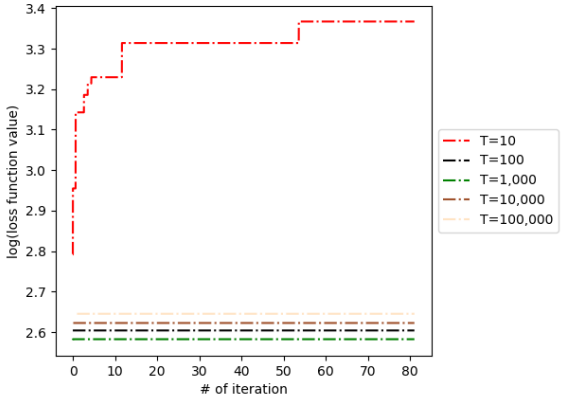
In Figures 2.2.5(a) and 2.2.5(b), we compare the performances of OGD and CONVGADAM for different learning rates and stepsizes. Figure 2.2.5(c) shows that performance of CONVGADAM for different T. Lastly, Figure 2.2.5(d) depicts the comparison of CONVGADAM and OGD. From Figures 2.2.5(a) and 2.2.5(b), we select the stepsize $2/\sqrt{1000}$ and the learning rate of $1/1000$. As we observe, CONVGADAM always exhibits a better performance than OGD.



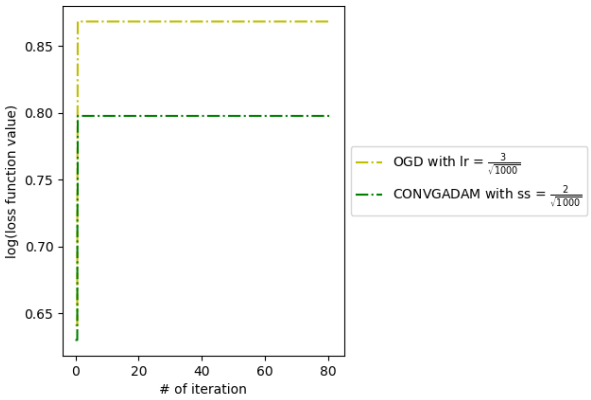
(a) Comparison of OGD for different learning rates



(b) Comparison of CONVGADAM for different stepsizes



(c) Comparison of CONVGADAM for different T



(d) Comparison of CONVGADAM and OGD

Figure 2.5. Performance of CONVGADAM on MINST8M dataset

CHAPTER 3

Topic Analysis for Text with Side Data

3.1. Introduction

As the conjoint and massive knowledge in the forms of news, blogs, web pages, etc., continues to be digitized and stored, discovery becomes more and more challenging and together with it the main underlying topics. A Bayesian multinomial mixture model, latent Dirichlet allocation (LDA) (Blei et al., 2003), has recently gained much popularity due to its simplicity, and usefulness in stratifying a large collection of documents by projecting every document to a low dimensional space which is spanned by a set of bases capturing the semantic aspects of the collection. However, as the acquisition of information becomes more convenient, text data can be accompanied by extra side data. For example, when customers post their comments for products or restaurants, they usually associate a comment with a rating score or a thumb-up or thumb-down opinion, and retailers usually provide categorical labels for the products in question. In addition, customer loyalty data can be pulled in. Taking such side data into account improves the ability of LDA to discover patterns and topics among the documents. Currently, there are two types of existing models that combine side data: (1) downstream topic models and (2) upstream topic models. The downstream models assume that the text content and side data are generated simultaneously given latent topics, while upstream models assume that the text content is generated conditioned on the side data as well. Our model belongs to the family of upstream topic models where we accommodate much more complex interactions between

side data and text by means of deep neural networks, when compared to other upstream topic models, i.e. DMR (Mimno and McCallum, 2008).

In this paper, we propose a new LDA-style topic model, namely hybrid neural network LDA (nnLDA), based on LDA and a neural network. Our model captures not only text content of the dataset, but also useful high-level content and secondary, non-dominant, and more salient statistical patterns from side data. Formally, the model represents the document-topic distribution as mixtures of feature-specific distributions. The prior distribution over topics is the output of a neural network whose input is side data, therefore, it is specific to each distinct combination of side data. Moreover, the neural network is optimized together with the rest of the model in a stochastic EM sampling scheme to better interpret the collection of the documents. The expectation step corresponds to finding the optimal word group and topic group while the maximization step aims to find the optimal neural network parameters and the topic-word distribution. In standard LDA, the prior is fixed while in nnLDA, it depends on a sample.

In this paper, we not only propose a more general model, nnLDA, but also present a complete technical proof confirming that nnLDA performs at least as well as plain LDA in terms of log likelihood. Furthermore, we provide an efficient variational EM algorithm for nnLDA. Lastly, we demonstrate our approach on a few real-world datasets. In summary, we make the following contributions.

- We provide a new topic model for text datasets with side data.
- We prove that the lower bound of log likelihood of nnLDA is greater than or equal to the lower bound of log likelihood of LDA for any dataset.
- We provide an efficient variational EM algorithm for nnLDA.

- We present numerical results showing that nnLDA outperforms LDA and DMR in terms of topic grouping, model perplexity, classification and text generation.

The paper is organized as follow. In the next section, we review several related works about incorporating side data in generative topic models. In Section 3, we present the hybrid neural network model, followed by our analyses of log likelihood of nnLDA and an efficient variational EM algorithm. In Section 4, we present experimental results comparing nnLDA with plain LDA and DMR.

3.2. Related Work

There are a large amount of extensions of the plain LDA model, however, a full retrospection of this immense literature exceeds the scope of this work. In this section, we state several kinds of variations of LDA which are most related to our new model and interpret the relationships among them.

LDA: Plain LDA, a probabilistic latent aspect model, has been widely applied on text documents (Blei et al., 2003) and (Wang et al., 2020), images (Li and Perona, 2005), and network entities (Airoldi et al., 2008) on account of its convenience and functionality in reducing the dimensionality of the data and generating interpretable and semantically coherent topics. It is an unsupervised model which is typically constructed on a distinct bag of words from input contents. Nevertheless, in many practical applications, besides the document contents, useful side information can be easily obtained. Furthermore, such side information often provides useful high-level or direct summarization of the content, while it is not directly involved in the plain LDA model to affect topic inference. In contrast, nnLDA incorporates such information

into latent aspect modeling by applying a neural network to discover secondary, non-dominant and more salient statistical patterns that may be more interesting and related to the user's goal.

Downstream Topic Models: One approach of incorporating side data in generative topic models is to generate both the text content and side data simultaneously given latent topics. More precisely, for this type of models, each hidden topic not only has a distribution over words but also has another distribution over side data. As a consequence, in training, the loss function in optimization-based learning is the joint likelihood of the content and side data. Examples of such "downstream" models are correspondence LDA (Corr-LDA) (Blei and Jordan, 2003), mixed-membership model for authorship (Erosheva et al., 2004), Group-Topic model (Wang et al., 2005), Topics over Time model (TOT) (Wang and McCallum, 2006), and maximum entropy discrimination LDA (MedLDA) (Zhu et al., 2012).

One of the most flexible downstream models is the supervised LDA (sLDA) model (Blei and McAuliffe, 2007), which has variants including multi-class sLDA (Wang et al., 2009) and TOT (Wang and McCallum, 2006). sLDA generates side data such as customers' ratings by maximizing the joint likelihood of the content data and the responses, where the likelihood-based objective is a generalized linear model (GLM) incorporating a proper link function with an exponential family dispersion function specified by the modeler for different types of side data. Compared to our model, in order to explicitly estimate probability distributions over all different side data, sLDA has to fully specify the link function and dispersion functions for the GLM, which increases the modeling complexity significantly. In essence, only a relatively small number of distinct side data vectors are allowed. Our model has no such restriction by applying a completely different approach.

Upstream Topic Model: In a "downstream" model, the side data is predicted based on the latent topics of the dataset, whereas in an "upstream" topic model, the side data is being conditioned on to generate the latent topics of the dataset. Another distinct difference from "downstream" topic models is the choice of the likelihood-based loss in optimization-based learning. More precisely, instead of maximizing the joint likelihood of the content and side data, an "upstream" topic model maximizes the conditional likelihood. Examples of such "upstream" topic models are Discriminative LDA (DiscLDA) (Lacoste-Julien et al., 2008), the scene understanding models (Sudderth et al., 2005) and the author-topic model (Rosen-Zvi et al., 2004). In the author-topic model, words are generated by first selecting one author uniformly from an observed author list and then selecting a topic from the topic distribution with respect to that specific author. Then, given a topic, words are selected from the topic-word distribution of that topic. This model assumes that each word is generated only by one author. There are a few extensions of the author-topic model which allow a mixture of latent topics for one document and one author, i.e. (Rosen-Zvi et al., 2004), (McCallum et al., 2007), (Dietz et al., 2007). However, these aforementioned models cannot accommodate combinations of modalities of side data, for example, the aforementioned models cannot handle categorical data and continuous data at the same time. In addition, the side data used in these models are either ratings or labels which essentially is the final intention of learning. Different from the previous models, DMR can handle combinations of modalities of side data, which uses the dot product to project the impact of side data onto the prior. An advanced version introduced in (Benton et al., 2016) is collective supervision of topic models, where aggregate-level labels are provided for groups of documents instead of individual documents, followed by a deterministic relationship between the labels and the priors. Similar to DMR, although the advanced version in (Benton et al., 2016) uses

group-level labels and can handle missing side data, it also employs dot product to directly project the impact of the side data onto the prior. Compared to DMR and collective supervision of topic models, nnLDA provides a more comprehensive and flexible learning of side data by applying a neural network when compared to the dot product employed in DMR and the normal distribution assumption in collective supervision of topic models.

3.3. Model and Algorithm

We first present notation and the setting. We use the language of text collections throughout the paper, referring to terminologies such as "words," "documents" and "corpus" since it makes the concepts more intuitive to understand. In general, similar to plain LDA, nnLDA is not restricted to text datasets, and can also be applied on other kinds of datasets, i.e. image datasets.

- A *word*, defined as an item from a vocabulary indexed by $\{1, \cdots, V\}$, is applied one-hot encoding. More precisely, using superscripts to denote components, the v'th word in the vocabulary is represented by a V-vector w such that $w^v = 1$ and $w^u = 0$ for all $u \neq v$.
- A *document* is a set of N words denoted by $d = \mathbf{w} = \{w_1, w_2, \cdots, w_N\}$ if it only contains textual data. Similarly, if a document contains q different kinds of side data together with the aforementioned textual data, we denote it by $d = (\mathbf{w}, \mathbf{s}) = (\{w_1, w_2, \cdots, w_N\}, \{s_1, s_2, \ \cdots, s_q\})$ where $\mathbf{s} \in \mathrm{R}^q$.
- A *corpus* is a collection of M documents denoted by $D = \{\mathbf{w}_1, \mathbf{w}_2, \cdots, \mathbf{w}_M\}$ for textual only documents and $(D, S) = \{(\mathbf{w}_1, \mathbf{s}_1), (\mathbf{w}_2, \mathbf{s}_2), \cdots, (\mathbf{w}_M, \mathbf{s}_M)\}$ for documents containing both side and textual data.

The main goal of nnLDA is to find a probabilistic model of a corpus that, by involving high-level summarization from side data, not only assigns high probability to documents in this corpus but also assigns high probability to other similar documents based on side data.

3.3.1. Generative Model

We propose the nnLDA model to explain the generative process of a document d with textual data **w** (containing N words) and side data (structural data) **s**, the steps of which can be summarized as follows.

(1) Choose $N \sim \mathsf{Poisson}(\xi)$.

(2) Choose $\mathbf{s} \sim \mathcal{N}(\mu, \sigma^2 I)$.

(3) Choose $\alpha_d = g(\gamma; \mathbf{s})$

(4) Choose $\theta \sim \mathsf{Dir}(\alpha_d)$.

(5) For each of the N words w_n:

  (a) Choose a topic $z_n \sim \mathsf{Multinomial}(\theta)$.

  (b) Choose a word w_n from $p(w_n \mid z_n, \beta)$, a multinomial probability conditioned on the topic z_n.

Notation "Poisson," "Dir" represents the Poisson and Dirichlet distribution, respectively. In step 3, g refers to a parametric model to generate α. In summary, the model has two trainable parameters: γ, the parameters of g for side info **s**; β, the topic-word distribution. In the meanwhile, there are three hyper parameters: μ and σ^2, the mean and the variance of the probability distribution for side data **s**; and K, which does not explicitly appear in the generative process, the number of topics.

Step 1 is independent of the remaining steps, which determines the number of words in the document. Then, for each document, step 2 provides a representation of side data **s** by using a normal distribution with mean μ and variance σ^2. Then, applying a model with input **s** in step 3 provides the prior α_d for the Dirichlet distribution. Next, the random parameter of a multinomial distribution over topics, θ, is generated by the Dirichlet distribution. Finally, for the n'th word in the document, step 5(a) first selects a topic z_n among the K different topics by the multinomial distribution with parameter θ, and then step 5(b) generates a word w_n based on the topic-word distribution β specific to topic z_n. Step 5 follows standard LDA.

3.3.2. Analysis

Note that a Dirichlet random vector $\theta = (\theta_1, \theta_2, \cdots, \theta_K)$ has the following probability density:

$$p(\theta \mid \alpha) = \frac{\Gamma\left(\sum_{i=1}^{K} \alpha_i\right)}{\prod_{i=1}^{K} \Gamma(\alpha_i)} \theta_1^{\alpha_1 - 1} \cdots \theta_K^{\alpha_K - 1},$$

where K is the number of topic groups, α is the prior of the Dirichlet distribution and θ takes values in the $(K-1)$-simplex. Then, the generative process implies that the conditional distribution of the nnLDA model of a document $d = (\mathbf{w}, \mathbf{s})$ is

$$\begin{aligned} P_1(\mathbf{w} \mid \mu, \sigma, \gamma, \beta) &= \tilde{\tilde{P}}_1(\mathbf{w} \mid \mathbf{s}, \gamma, \beta) \\ &= \int \tilde{\tilde{p}}(\theta \mid \mathbf{s}, \gamma) \left(\prod_{n=1}^{N} \sum_{z_k} \tilde{p}(z_k \mid \theta) \tilde{p}(w_n \mid z_k, \beta) \right) \mathrm{d}\theta \\ &= \int \tilde{p}(\theta \mid \mu, \sigma, \gamma) \left(\prod_{n=1}^{N} \sum_{z_k} \tilde{p}(z_k \mid \theta) \tilde{p}(w_n \mid z_k, \beta) \right) \mathrm{d}\theta \\ &= \int \tilde{p}(\theta \mid \mu, \sigma, \gamma) \left(\prod_{n=1}^{N} \sum_{i=1}^{K} \prod_{j=1}^{V} (\theta_i \beta_{ij})^{w_n^j} \right) \mathrm{d}\theta, \end{aligned}$$

which in turn yields

$$\begin{aligned} P_1(D \mid \mu, \sigma, \gamma, \beta) &= \mathbb{E}\left[\int \tilde{p}(\theta_d \mid \mu, \sigma, \gamma) \left(\prod_{n=1}^{N} \sum_{z_{d_k}} \tilde{p}(z_{d_k} \mid \theta_d) \tilde{p}(w_{d_n} \mid z_{d_k}, \beta) \right) \mathrm{d}\theta_d \right] \\ &= \mathbb{E}\left[\int \tilde{p}(\theta_d \mid \mu, \sigma, \gamma) \left(\prod_{n=1}^{N} \sum_{i=1}^{K} \prod_{j=1}^{V} (\theta_i \beta_{ij})^{w_n^j} \right) \mathrm{d}\theta_d \right], \end{aligned}$$

where $\tilde{p}(\theta_d \mid \mu, \sigma, \gamma) = \tilde{\tilde{p}}(\theta_d \mid \mathbf{s}, \gamma) = p(\theta_d \mid g(\gamma; \mathbf{s})) = p(\theta_d \mid \alpha_d)$ for a corpus D.

The nnLDA model represented above is a probabilistic graphical model with three levels. Parameters μ, σ, γ and β are corpus-level parameters, which are assumed to be sampled once in the generative process of a corpus. Variables α_d and θ_d are document-level variables, which are sampled once per document. Finally, w_{d_n} and z_{d_k} are word-level variables, sampled once for each word in each document.

In the rest of this section, we provide an analytical comparison of standard LDA and nnLDA.

Compared to standard LDA, nnLDA employs an extra neural network g to generate document-level variable α_d. Since nnLDA is "richer" than LDA, we expect that it should produce a higher likelihood. Without assumptions on $g(\gamma;\cdot)$ this does not hold since, for example, $g(\gamma;\cdot)$ can map everything to a constant vector different from the prior used by LDA. As a result, in order for the statement to hold the network must be expressive. The question to consider is whether a neural network is capable of memorizing arbitrary side data of a given size. We tackle this question by introducing the concept of finite sample expressivity which is an extension of a similar definition in (Yun et al., 2019). Given the definition, if $g(\gamma;\cdot)$ has finite sample expressivity, nnLDA at least can find the optimal α^* used in standard LDA.

Definition 1. *Function $g(\gamma;\cdot)$ has finite sample expressivity if for all inputs $x_i \in \mathrm{R}^{d_x}, 1 \leq i \leq N$ and for all $y_i \in [-M,+M]^{d_y}, 1 \leq i \leq N$ for some constant $M > 0$, there exists a parameter γ such that $g(\gamma;x_i) = y_i$ for every $1 \leq i \leq N$.*

Based on Definition 1, Theorem 3.1 shown in (Yun et al., 2019) provides a specific set of constraints, i.e. any 3-layer (i.e., 2-hidden-layer) ReLU FCNN with hidden layer widths d_1 and d_2 can fit any arbitrary dataset if $d_1 d_2 \geq 4Nd_y$, where d_y and N are the dimension of the label and the number of samples, respectively. By extending the aforementioned theorem, Proposition 3.4 and Theorem 4.1 in (Yun et al., 2019) argue that any FCNN given constraints on the number of neurons in each layer is able to have finite sample expressivity.

In the following, we assume that $g(\gamma;\cdot)$ has finite sample expressivity. Therefore, given K and any α^* representing the number of topic groups and optimal parameters in LDA, since $\alpha^* \in [-M,+M]^K$ for some constant M, there exists a γ_1 such that, for all inputs $\mathbf{s}_i$ and α^*, $g(\gamma_1;\mathbf{s}_i) = \alpha^*$ for all $1 \leq i \leq N$.

We next prove that the optimized probability of nnLDA is at least as good as that of plain LDA. Let α^* and β^* be optimal solutions to $P_2 = \max_{\alpha,\beta} P(\, D|\, \alpha, \beta)$ of LDA, meanwhile, let μ^*, σ^* and γ^* be optimal solutions to $P_1 = \max_{\mu,\sigma,\gamma} P_1(\, D|\, \mu, \sigma, \gamma, \beta^*)$ of nnLDA (see Appendix C.1 for formal definitions).

Theorem 11. *If α^*, β^* are optimal solutions to LDA, then there exists optimal solutions μ^*, σ^* and γ^* to nnLDA such that*

$$P_1(D \mid \mu^*, \sigma^*, \gamma^*, \beta^*) \geq P_2(D \mid \alpha^*, \beta^*).$$

PROOF. See Appendix D.1. □

While Theorem 13 asserts that when it comes to model fit nnLDA fits the data better than LDA, it does not provide a gap statement. If the side data provides positive influence during the learning process by a constant C, then, due to the independence of words, topics and documents, we are able to argue that the optimized probability is at least improved by $C - 1$.

Theorem 12. *For any document $(\boldsymbol{w}, \boldsymbol{s}) \in (D, S)$, if $\hat{p}(w_i \mid \alpha^*, \beta^*) \neq 0$ for all i, and there exists a positive constant $C > 1$ such that $\prod_{i=1}^{N} \tilde{p}(w_i \mid \gamma^*, \beta^*, \mu^*, \sigma^*) \geq C \prod_{i=1}^{N} \hat{p}(w_i \mid \alpha^*, \beta^*)$ for every $w_i \in \boldsymbol{w}$, and if D in P_1 and D in P_2 follow the same distribution, then*

$$\frac{P_1(D \mid \mu^*, \sigma^*, \gamma^*, \beta^*) - P_2(D \mid \alpha^*, \beta^*)}{P_2(D \mid \alpha^*, \beta^*)} \geq C - 1.$$

PROOF. See Appendix C.3 for a formal proof. □

The assumption on $\hat{p}(w_i \mid \alpha^*, \beta^*)$ in Theorem 12 is reasonable since it indicates that all documents are not randomly generated. The positive constant C in the assumption captures

the improvement given by the side data. In other words, as long as the side data has positive impact on the text data, this assumption holds. Next, we link the existence of C to lift from data mining. Let us define lift as

$$l(d) = \frac{P(\mathbf{w})P(\mathbf{s})}{P(\mathbf{w}, \mathbf{s})}$$

with $d = (\mathbf{w}, \mathbf{s})$. Lift measures the dependency level of words $\mathbf{w}$ and side data $\mathbf{s}$. If $l(d) < 1$ for d with N words and $P(\mathbf{s}) > 0$, we have

$$P(\mathbf{s}) \prod_{n=1}^{N} P(w_n) = P(\mathbf{w})P(\mathbf{s}) < P(\mathbf{w}, \mathbf{s}) = P(\mathbf{s}) \prod_{n=1}^{N} P(w_n|\mathbf{s}),$$

and in turn

$$\prod_{n=1}^{N} P(w_n) < \prod_{n=1}^{N} P(w_n|\mathbf{s}),$$

and

$$\prod_{n=1}^{N} \hat{p}(w_n \mid \alpha^*, \beta^*) < \prod_{n=1}^{N} \tilde{p}(w_n \mid \gamma^*, \beta^*, \mu^*, \sigma^*).$$

This implies that there exists $C > 1$. In summary, when $l(d) < 1$ and $P(\mathbf{s}) > 0$ for each d in the corpus, Theorem 12 holds. Lift essentially measures the dependency of $\mathbf{w}$ and $\mathbf{s}$, which is widely used in data mining. The condition indicates that the side data helps to link the words to the documents they are more likely to be in.

Informally, in the proof, due to the independence assumption of words, topics and documents in nnLDA, the generative probability of nnLDA for a corpus can be reformulated as a product of $\tilde{p}(\theta_d \mid \mu^*, \sigma^*, \gamma^*)$ and conditional probability of words $\tilde{p}(w_n \mid \theta_d, \beta^*)$. Likewise,

the same property holds for plain LDA. Lastly, given a relationship between documents $d = \mathbf{w}$ and $d = (\mathbf{w}, \mathbf{s})$ as an expression of the conditional probability of words, we are able to build a connection of the optimized probabilities between nnLDA and LDA.

3.3.3. Variational Inference with EM Algorithm

We train the nnLDA model using a stochastic EM sampling scheme, in which we alternate between sampling topic assignments from the current prior distribution conditioned on the observed words and side data, and optimizing the parameters given the topic assignments.

Details are similar to those in (Blei et al., 2003). In this section, instead of showing all the details, we only point out the differences from the derivation of plain LDA. By applying the Jensen's inequality and KL divergence between the variational posterior probability and the true posterior probability, which is a formally stated technique in (Blei et al., 2003), a lower bound of log likelihood reads

$$\begin{aligned} L(\xi, \phi; \gamma, \beta) = {} & \mathbb{E}_q \left[\log p(\theta \mid g(\gamma; \mathbf{s}))\right] + \mathbb{E}_q \left[\log p(z \mid \theta)\right] + \mathbb{E}_q \left[\log p(\mathbf{w} \mid z, \beta)\right] \\ & - \mathbb{E}_q \left[\log q(\theta)\right] - \mathbb{E}_q \left[\log q(z)\right], \end{aligned} \tag{3.1}$$

where ξ, ϕ are variational parameters of θ and z, respectively, and $q(\cdot)$ represents the variational distribution. Then, the iterative algorithm is

(1) (E-step) For each document, find the optimizing values of the variational parameters ξ and ϕ of z and θ, respectively.

(2) (M-step) Maximize the resulting lower bound of log likelihood with respect to the model parameters γ and β.

The E-step is similar to the E-step in (Blei et al., 2003) except replacing prior α by $g(\gamma; \mathbf{s})$. We run the E-step until it converges for each document. The M-step is finding a maximum likelihood estimation with expected sufficient statistics for each document under the approximate posterior parameters ξ and ϕ, which are computed in the E-step. Likewise, since the log likelihood objective related to β does not involve $g(\gamma; \mathbf{s})$, we are allowed to directly borrow the update rule of β from (Blei et al., 2003), which is

$$\beta_{ij} \propto \sum_{d=1}^{M} \sum_{n=1}^{N} \phi^*_{dni} w^j_{dn}.$$

In contrast, for the neural network parameter γ, we resort to log likelihood objective related to γ as follows,

$$\begin{aligned} L_{[\gamma]} = \sum_{d=1}^{M} \Bigg(& \log \Gamma(\sum_{j=1}^{K} [g(\gamma; \mathbf{s}_d)]_j) - \sum_{i=1}^{K} \log \Gamma([g(\gamma; \mathbf{s}_d)]_i) \\ & + \sum_{i=1}^{K} \left(([g(\gamma; \mathbf{s}_d)]_i - 1) \left(\Psi(\xi_{di}) - \Psi(\sum_{j=1}^{K} \xi_{dj}) \right) \right) \Bigg), \end{aligned}$$

where M is the number of documents in the corpus, and Ψ is the digamma function, the first derivative of the log Gamma function. Then, applying the backpropagation approach provides the derivative and the update rule for parameter γ.

3.4. Experimental Study

In this section, we compare the nnLDA model with standard LDA and the DMR model introduced in (Blei et al., 2003) and (Mimno and McCallum, 2008), respectively. We conduct experiments on five different-size datasets among which one is a synthetic dataset and the remaining four are real-world datasets. For these datasets, we study the performance of topic

grouping, perplexity, classification and comment generation for nnLDA, plain LDA and DMR models. For each of the tasks, some datasets are not eligible to be examined due to lack of information. The synthetic dataset is publically available at `https://github.com/biyifang/nnLDA/blob/main/syn_file.csv` while the real-world datasets are proprietary.

3.4.1. Datasets and Training Details

The first dataset we use is a synthetic dataset of 2,000 samples. Each sample contains a customer's feedback with respect to his or her purchase along with the characteristics of the product. More precisely, there are two different categories, which are product and description. In the product category, it can either be TV or burger; similarly, in the description category, the word can either be price or quality. In order to generate comments, we assign a bag of words to each combination of product and description as shown in Table 3.1.

Category combination	Bag of words
(burger, price)	value, pricey, ouch, steep, cheap, value, reason, accept, unreason, unacceptable
(burger, quality)	nasty, fantastic, delicious, tasty, juicy, unreason, unacceptable, reason, accept, fresh
(TV, price)	promotion, affordable, value, increase, expensive, tasty, economical, fancy, okay
(TV, quality)	fabulous, fantastic, promising, sharp, large, clear, eco friendly, fresh, pixilated

Table 3.1. Synthetic Dataset

After randomly selecting one category combination from the four combinations, a comment is generated containing at least one word and at most five words with an average 2.97 words, by selecting a certain number of words at random from the corresponding bag.

The second dataset is a real-world dataset, PTS for short, which has 795 samples. Each sample contains a customer's short feedback and rating with respect to his or her purchase along with the characteristics of the product. Additionally, the category (side data) selected for nnLDA corresponds to sectors, which are generalizations of products. In this dataset, there is only only 1 word in the shortest comment, while the longest comment in the dataset contains 49 words. Overall, the average length of the comments is 10.6 words. For example, a customer, who bought a product belonging to sector Baby, leaves a comment "Cheap& Soft" with a rating of 3.

The third dataset WIP is a medium-size dataset with 3,451 samples. Each sample contains a customer's short feedback and rating with respect to his or her purchase along with the characteristics of the product. The sector attribution is again side data when training models with one feature. The other attribution counted for models with two features is channel. The most concrete comment in the dataset has 138 words, while the briefest comment has only 1 word. In the meanwhile, the average length of the comments in the dataset is 8.9 words.

DCL is another medium-size dataset of 5,427 samples. Different from the PTS and WIP datasets, each sample in DCL contains a customer's long feedback and rating with respect to his or her purchase along with the characteristics of the product. Additionally, the side data selected for nnLDA corresponds to groups of products. The smallest number of words for a comment in this dataset is 1, while the largest is 988. Overall, the average length of the comments is 61.7

words. A short sample comment is “quick points that will be all that matters to a buyer wanting accurate metrics to buy by tinny sound but plenty of audio hookups.”

The last dataset is RR, which has 100,000 samples, from which we randomly select 10,000 samples. Each sample contains a customer’s feedback with respect to his or her purchase along with the characteristics of the product. Additionally, the side data for nnLDA corresponds to the category, which can be grocery, health and personal care, furniture, kitchen, etc. The longest comment has 418 words while the shortest comment has only 1 word as the previous datasets, and the average length of the comments is 69.5 words. A short sample comment reads “great tasting oil and made the most excellent gluten free chocolate cake.”

Due to the lack of some information from the certain datasets, we are unable to study all tasks of interest for all of these datasets. For the topic grouping task, we examine the ability of nnLDA, plain LDA and DMR to assign the comments from the same topic group into the same correct topic group. For this task, we only conduct experiments on the synthetic dataset since only the topic groups of the synthetic dataset are clear. For the perplexity task, we compute the logarithm of the perplexity of all the words in the corresponding dataset. We do not study the performance of perplexity for the synthetic dataset since we know the true number of topic groups. For the classification task, we use the probability vector generated by the topic models to predict the rating for that comment. Since the RR dataset does not have ratings, we are unable to examine the classification ability of the topic models on the RR dataset. The last task tests the performance of the topic models on generating new comments. For this task, we only conduct experiments on the two smallest real world datasets since it is of interest how topic models perform given a small number of samples. Table 3.2 presents the tasks of interest for each dataset.

Dataset	Topic grouping	Perplexity	Classification	Comment generation
Synthetic dataset	Yes	No	No	No
PTS	No	Yes	Yes	Yes
WIP	No	Yes	Yes	Yes
DCL	No	Yes	Yes	No
RR	No	Yes	No	No

Table 3.2. Tasks of Interest

For all of these datasets, we employ a two-layer fully connected neural network as $g(\gamma; \cdot)$ in nnLDA. Furthermore, we set the number of neurons to be 20 in the first layer, the number of neurons of the second layer to be the number of topic groups assigned in the beginning and the batch size to be 64. All features of the side data are categorical and are one-hot encoded. Additionally, all weights in $g(\gamma; \cdot)$ are initialized by Kaiming Initialization (He et al., 2015). We apply the ADAM algorithm with the learning rate of 0.001 and weight decay being 0.1. Meanwhile, we train all the models using EM with exactly the same stopping criteria of stopping E-step and M-step when the average change over the whole training dataset in the expected log likelihood becomes less than 0.01%. We vary the number of topic groups from 4 to 30. For DMR, we use the same values for the parameters as those in (Mimno and McCallum, 2008). All the algorithms are implemented in Python with Pytorch and trained on a single GPU card.

3.4.2. Experimental Results

In this section, we present all the results based on the tasks of interest.

Overall, nnLDA outperforms plain LDA and DMR in all datasets in terms of topic grouping, classification, perplexity and comment generation. Meanwhile, based on the fact that the last two datasets have many more words and more intrinsic concepts in their comments when compared to the first three datasets, nnLDA exceeds the performance of plain LDA and DMR dramatically when a document contains several topics or it is more comprehensive.

3.4.2.1. Topic Grouping. Table 3.3 shows the most frequent 5 words in each topic group generated by plain LDA, DMR and nnLDA when setting the number of topic groups to be 4 in the synthetic dataset. The topic groups generated by plain LDA and DMR are very vague and it is very hard to distinguish which topic group is describing what combination of product and description, while the topic groups given by nnLDA are very distinguishable, i.e. topic group 1 is about (burger, quality), topic group 2 is about (TV, price), topic group 3 is about (TV, quality) and topic group 4 is about (burger, price). It identifies correctly the seed topics. Therefore, nnLDA outperforms plain LDA in grouping.

	plain LDA	DMR	nnLDA
Topic group 1	promising, rebate, sharp, increase, outstanding	pricey, unacceptable, juicy, pixilated	unreason, unacceptable, juicy delicious, nasty
Topic group 2	unreason, value, okay, steep, ecofriendly	ouch, steep, tasty, unreason, promotion	promotion, increase, tasty, economical, okay
Topic group 3	reason, accept, promotion, large, unacceptable	accept, fantastic, value reason, affordable	fresh, promising, fantastic, large, eco friendly
Topic group 4	fresh, reason, outstanding, ecofriendly, fantastic	sharp, delicious, accept, fresh, clear	reason, accept, value, steep, cheap

Table 3.3. Top words of groups generated by LDA, DMR and nnLDA

	macro-precision	macro-recall	macro-F1	micro-F1
LDA	0.7238	0.7272	0.7211	0.7240
DMR	0.7238	0.7460	0.7313	0.7392
nnLDA	0.7401	0.7919	0.7536	0.7905
relative improvement from LDA	2.25%	8.90%	4.51%	9.19%
relative improvement from DMR	2.25%	6.15%	3.05%	6.94%

Table 3.4. Precision, recall and relative improvement of the synthetic dataset generated by LDA, DMR and nnLDA

Additionally, based on the top words of topics generated by LDA, DMR and nnLDA, we are able to assign the most related category combination to a comment with respect to a model. Since we have the category combination of each comment, Table 3.4 shows the macro-recall,

macro-precision and macro-F1 scores and micro-F1 of LDA, DMR and nnLDA, respectively, when training on the synthetic dataset, and the overall relative improvement of nnLDA. As the table shows, nnLDA outperforms plain LDA and DMR, which implies that nnLDA assigns more samples correctly to the right topic group. Therefore, in general, nnLDA improves the recall, precision and F1 scores.

In conclusion, nnLDA outperforms standard LDA and DMR in terms of the ability of topic grouping.

3.4.2.2. Perplexity. Figures 3.1 and 3.2 represent the log(perplexity) of plain LDA, DMR and nnLDA on the PTS and WIP datasets, respectively. Additionally, in Figure 3.2, for DMR and nnLDA, we not only conduct experiments on the dataset with the single feature (sector) as the side data, denoted as "DMR with single feature" and "nnLDA with single feature," but also on the dataset with two features (sector and channel) as side data, denoted as "DMR with two features" and "nnLDA with two features," respectively. The smallest log(perplexity) values generated by plain LDA and DMR are competitive to those of nnLDA for these two datasets. In Figure 3.1, the log(perplexity) value generated by plain LDA increases as the number of topic groups grows, while the log(perplexity) values generated by DMR and nnLDA decrease first and then increase as the number of topic groups increases on the PTS dataset. As it is shown in Figure 3.2, the log(perplexity) values generated by plain LDA and DMR increase as the number of topic groups grows on the WIP dataset. However, the log(perplexity) values generated by nnLDA decrease first and then increase as the number of topic groups increases on both of the aforementioned datasets. Moreover, we examine DMR and nnLDA models with two features on the WIP dataset, which take both sector and channel attributions as side data into account, in Figure 3.2. As we can observe, the minimum log(perplexity) generated by nnLDA with two

features (sector and channel attributions) is better than that of nnLDA with the single feature (sector attribution), although the optimal number of topic groups occurs at a different point since more side data is provided. Consequently, plain LDA does not learn the datasets, and DMR is able to learn the small datasets. In contrast, nnLDA starts learning the datasets as the log(perplexity) value decreases in the beginning and finds an optimal number of topic groups, then it gets confused since the number of topic groups are more than needed. Furthermore, nnLDA with two features provides better log(perplexity) than nnLDA with the single feature. Therefore, nnLDA is more capable of understanding the datasets; both small and medium-size datasets with short comments.

When learning more complex datasets, the advantage of the nnLDA model becomes more pronounced. Figure 3.3 represents the log(perplexity) of plain LDA, DMR and nnLDA on the DCL dataset, while Figure 3.4 shows the same on the RR dataset. In these figures we observe that the log(perplexity) generated by plain LDA and DMR blows up as the number of topic groups increases, while the log(perplexity) generated by nnLDA decreases first and then increases as the number of topic groups grows. Furthermore, the log(perplexity) values of nnLDA are much smaller than those of plain LDA and DMR. Consequently, nnLDA performs as well as plain LDA and DMR in small and medium-size datasets with short comments, and at the same time, nnLDA explains the datasets better than plain LDA and DMR in medium and large size datasets with long comments. There is also a trade-off between the accuracy and running time as shown in Table 3.5. In the table, we compare the running time of plain LDA, DMR with one feature and nnLDA with one feature on three different datasets. We observe that nnLDA spends more time than both DMR and plain LDA on training. In conclusion, nnLDA

performs better on learning while it requires a slightly longer training time. It is less than 10% slower than DMR.

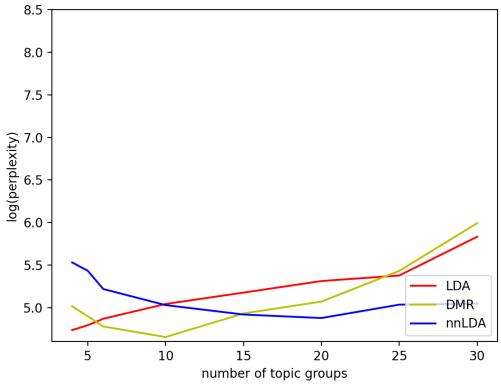


Figure 3.1. PTS dataset

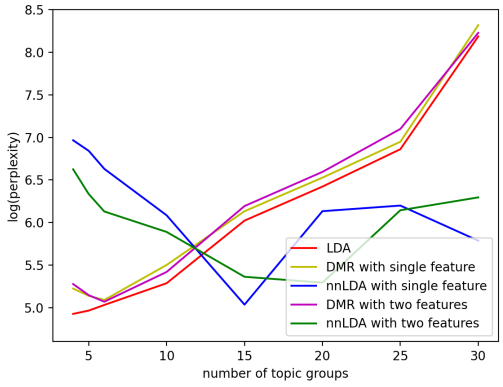


Figure 3.2. WIP dataset

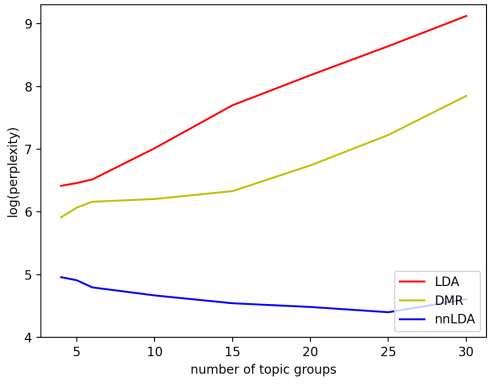


Figure 3.3. DCL dataset

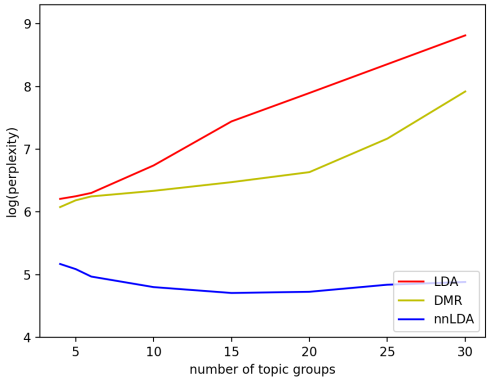


Figure 3.4. RR dataset

running time(s)	plain LDA	DMR	nnLDA
PTS	3	4	4
WIF	19	24	26
DCL	138	179	191

Table 3.5. Running time of different models on different datasets

In the following section, we study the classification problem of predicting the rating of each sample. In all the cases, we use 10-fold cross validation, which holds out 10% of the data for test purposes and trains the models on the remaining 90%. We apply nnLDA, plain LDA and DMR to find the probability of each sample to be assigned to each topic group and treat it as the feature matrix. Lastly, we train a classification model (xgboost (Chen and Guestrin, 2016)) on the feature matrix with the rating labels as the ground truth.

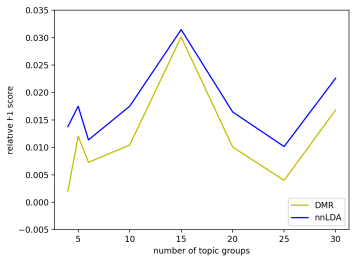


Figure 3.5. PTS

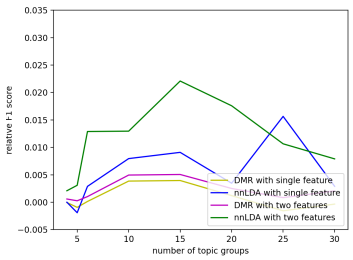


Figure 3.6. WIP

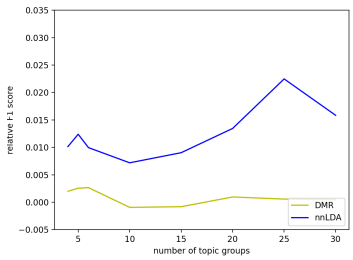


Figure 3.7. DCL

3.4.2.3. Classification. Figures 3.5, 3.6 and 3.7 depict the relative F1 scores of DMR and nnLDA with respect to plain LDA on the PTS, WIP, and DCL datasets, respectively. In Figure 3.5, the most distinguishable difference of F1 scores occurs when the number of topic groups is 15, where nnLDA has a gap of 0.032. In the meanwhile, DMR achieves its best performance at the same point with a gap of 0.030. Moreover, this chart shows that nnLDA outperforms

plain LDA and DMR no matter what the number of topic groups is. In Figure 3.6, when using the single feature (sector attribution), the biggest gaps of F1 scores happen when the number of topic groups is 15 for DMR and 25 for nnLDA. The biggest gap between nnLDA and plain LDA is 0.016, while the largest gap between DMR and plain LDA is 0.003. Considering models using two features (sector and channel attributions) as the side data, the highest relative F1 score given by nnLDA with two features is 0.022 with 15 topic groups, compared with 0.004 produced by DMR with 10 topic groups. Although plain LDA provides a slightly higher F1 score than nnLDA when applying 5 topic groups, nnLDA outperforms plain LDA and DMR significantly given any other number of topic groups. In Figure 3.7, the highest relative F1 score given by nnLDA is 0.022 with 25 topic groups, compared with 0.003 given by DMR for 6 topic groups. Moreover, this figure shows that nnLDA outperforms plain LDA dramatically whatever the number of topic groups is.

Therefore, nnLDA performs better than plain LDA and DMR when predicting the rating given customer's comments and product information in all datasets.

3.4.2.4. Comment Generation. In this section, we compare the comments generated by nnLDA with plain LDA and DMR. We set the number of topic groups to be 5 since all of plain LDA, DMR and nnLDA have relatively low perplexity scores based on Figures 3.1 and 3.2, and comparable F1 scores based on Figures 3.5 and 3.6 on the PTS and WIP datasets. A comment is generated based on the topic-document probability of the sample and the topic-word distribution. More precisely, for DMR and LDA, the prior α is generated based on the side data (sector) first while α is fixed in plain LDA. Next, a comment is created by selecting the top words which have the highest score computed by adding the products of the topic-document probability and topic-word for each word. Then, we randomly pick 50 comments that contain a certain level of

information, for example, we rule out comments like "N/A." Meanwhile, in order to evaluate the quality of comment generation, we employed 50 PhD students. Each one of them assessed a pair of comments (one based on plain LDA or DMR, and the other one based on nnLDA) for the same side data and they provided an assessment as to which one is better.

	Number of generated comments	
	PTS	WIP
plain LDA ¡ nnLDA	15	22
plain LDA ¿ nnLDA	11	9
plain LDA $\sim$ nnLDA	24	19
DMR ¡ nnLDA	16	20
DMR ¿ nnLDA	11	10
DMR $\sim$ nnLDA	23	20

Table 3.6. Comparison of the generated comments on different datasets

The upper left three values in Table 3.6 show the comparison of the generated comments given by plain LDA and nnLDA on the PTS dataset. Based on the table, among all these 50 samples, nnLDA generates more accurate comments in 15 samples, while plain LDA does better in 11 samples, and the two are tied for the remaining 24 samples. The lower left three values in Table 3.6 show the comparison of the generated comments given by DMR and nnLDA on the PTS dataset. Based on the table, among all these 50 samples, nnLDA generates more accurate comments in 16 samples, while DMR does better in 11 samples, and the two are tied for the remaining 23 samples. On the PTS dataset, nnLDA generates in $\frac{15-11}{50} = 8\%$ more reasonable comments compared to plain LDA, and in $\frac{16-11}{50} = 10\%$ more comparing to DMR.

The right column in Tables 3.6 shows the comparison of the generated comments given by plain LDA and nnLDA, and DMR and nnLDA on the WIP dataset, respectively. The observations and conclusions are similar. Furthermore, the advantage in number is more obvious on the WIP dataset, i.e. the improvement of nnLDA compared to plain LDA is as large as $\frac{22-9}{50} = 26\%$ and the improvement from DMR to nnLDA is $\frac{20-10}{50} = 20\%$. Therefore, taking generated comments into consideration, nnLDA generates more reasonable comments than plain LDA and DMR for both small and medium-sized datasets.

CHAPTER 4

Tricks and Plugins to GBM on Images and Sequences

4.1. Introduction

Deep convolutional neural networks (CNNs) and transformers such as BERT have had great recent success in learning image representations for vision tasks and NLP, respectively. Given the outstanding results produced by these networks, they have been widely applied in image classification ((He et al., 2016), (rizhevsky et al., 2017), (Lin et al., 2015)), object detection ((Girshick et al., 2014), (Iandola et al., 2014), (Ren et al., 2015)), speech recognition, ((Nakatani, 2019), (Yeh et al., 2019), (Dong et al., 2018)), and language translation ((Li et al., 2019), (Di Gangi et al., 2019), (Zhou et al., 2018)). However, an optimal image or text representation for each task is unique and finding an optimal deep neural network structure is a challenging problem. There are some approaches (neural architecture search) for designing these deep networks such as AutoML for Model Compression (AMC) in (He et al., 2018) and LEAF in (Liang et al., 2019), however, these methods require weeks of training on thousands of GPUs. In the meanwhile, ensemble methods for classification and regression have gained a lot of attention in recent years, which perform, both theoretically and empirically, substantially better than single models in a wide range of tasks, i.e. boosting decision trees (Quinlan, 2004). In order to tackle the design challenge specifically for CNNs, an idea of combining boosting and shallow CNNs is proposed in (Brahimi et al., 2016). The idea is to simplify the complicated design process of deep neural networks by employing the boosting strategy which combines the

strengths of multiple CNNs. However, the memory requirement and running time become challenging when the weak learner is not extremely simple. Moreover, very limited contribution has been made to the case when the weak learner is a transformer. Furthermore, no work has been conducted around the idea of only using partial data with weak learners.

In this paper, we propose a family of boosting algorithms for images, namely subgrid BoostCNN, and another family of boosting algorithms for sequences, namely BoostTransformer, which are both based on boosting, deep CNNs and transformers. We select a subset of features for each weak learner, where the concepts are borrowed from random forests. This strategy requires new ideas in order to accommodate unstructured data. Moreover, we apply the concept of importance sampling to the combination of boosting and a transformer, which assigns a probability to each sample.

Subgrid BoostCNN is aimed to solve the same problem as deep CNNs but it provides higher accuracy with lower running time and memory requirements. Subgrid BoostCNN builds on the previous boosting Deep Convolutional Neural Networks (BoostCNN) (Brahimi et al., 2016). One important new aspect in subgrid BoostCNN is that it does not require a full image for training a weak learner; instead, it only selects important pixels based on the gradient from each image together with the corresponding residual to train the current weak learner. Although, this might amplify the error by breaking the original relationship between a pixel and its neighborhood, subgrid BoostCNN focuses on important pixels. Another technique aiming to reduce the running time is omitting the optimization process for the original full CNN, which is employed to find the important pixels for the weak learner; instead, we borrow the CNN portion from the last weak learner concatenated with the fully connected layer used in the 1st iterate

to compute the importance value of each pixel, and train the CNN concatenated with an appropriate fully connected layer. Consequently, subgrid BoostCNN does the optimization process once in each iteration, which is the same as BoostCNN, while subgrid BoostCNN has fewer parameters when compared with BoostCNN. This subgrid trick is essential especially when the training process for the weak learner is computationally demanding. Furthermore, we demonstrate subgrid BoostCNN on three different image datasets and argue that subgrid BoostCNN outperforms both BoostCNN and deep CNNs. More precisely, subgrid BoostCNN improves the accuracies by $1.16\%, 0.82\%, 12.10\%$ on CIFAR-10, SVHN and ImageNet datasets, respectively, when compared to standard CNN models. In addition, subgrid BoostCNN obtains accuracies $0.34\%, 0.50\%, 4.19\%$ higher than those generated by BoostCNN on the aforementioned datasets, respectively.

BoostTransformer is an algorithm which combines the merits of boosting and transformers. BoostTransformer incorporates boosting weights with transformers based on least squares objective functions. Motivated by the successful combination of BoostCNN and the subgrid trick, we propose subsequence BoostTransformer, which does not require the full data for training weak learners. In subsequence BoostTransformer, important tokens, which are from the input, are selected for each weak learner based on the attention distribution (Vaswani et al., 2017). Similarly, we might loss the connections between consecutive words, while informative words are emphasized during learning. Consequently, subsequence BoostTransformer takes less time to achieve a better accuracy when compared to vanilla BoostTransformer. Moreover, motivated by the phenomenon that overfitting in BoostTransformer appears early, we propose a new algorithm, namely importance-sampling-based BoostTransformer, which combines

the merits of BoostTransformer and importance sampling. Importance-sampling-based BoostTransformer first computes a probability distribution for all the samples in the dataset; then in each iteration, it randomly chooses a subset of samples based on the pre-computed probability distribution; lastly, similar to BoostTransformer, it trains the weak learner on the selected samples. This algorithm not only delays overfitting, but also improves the accuracy and significantly reduces the running time. In the meanwhile, we present a complete technical proof for importance-sampling-based BoostTransformer showing that the optimal probability distribution is proportional to the norm of the residuals. Lastly, we conduct computational experiments demonstrating a superior performance of the proposed algorithms. More precisely, BoostTransformer provides higher accuracy and more stable solutions when compared to transformers. Moreover, subsequence BoostTransformer and importance-sampling-based BoostTransformer not only provide better and more robust solutions but also dramatically reduce the running time when compared to transformers. Compared to standard transformers, BoostTransformer, subsequence BoostTransformer and importance-sampling-based BoostTransformer provide an average of $0.87\%, 0.55\%, 0.79\%$ accuracy improvements, respectively, on IMDB, Yelp and Amazon datasets. Furthermore, subsequence BoostTransformer and importance-sampling-based BoostTransformer take only two thirds and one half of time transformers need to learn the datasets, respectively.

In summary, we make the following contributions.

- We provide a better boosting method for deep CNNs, i.e. subgrid BoostCNN, which only requires important pixels from the image dataset where such pixels are selected dynamically for each weak learner.

- We provide a boosting method for sequences, i.e. BoostTransformer, which combines the merits of boosting and transformers.
- We provide a better boosting method for transformers, i.e. subsequence BoostTransformer, which does not require the full sequences but only important tokens.
- We provide another enhancement for BoostTransformer, i.e. importance-sampling-based BoostTransformer, which combines importance sampling and BoostTranformer. Moreover, we provide a proof showing that the optimal probability distribution for the samples is proportional to the norm of the residuals.
- We present numerical results showing that subsequence BoostTransformer and importance-sampling-based BoostTransformer outperform vanilla transformers on select tasks and datasets.

The rest of the paper is organized as follows. In the next section, we review several related works in gradient boosting machine, CNN and transformers. In Section 3, we state the formal optimization problem and provide the exposition of the subgrid BoostCNN. In the subsequent section, we propose BoostTransformer, subsequence BoostTransformer and importance-sampling-based BoostTransformer, followed by the analysis of the optimal probability distribution for importance-sampling-based BoostTransformer. In Section 5, we present experimental results comparing the different algorithms.

4.2. Related Work

There are many extensions of Gradient Boosting Machine (GBM) (Natekin and Knoll, 2013), however, a full retrospection of this immense literature exceeds the scope of this work. In this section, we mainly state several kinds of variations of GBM which are most related to

our new algorithms, together with the two add-ons to our optimization algorithms, i.e. subgrid and importance sampling.

Boosting for CNNs: Deep CNNs, which have recently produced outstanding performance in learning image representations, are capable of learning complex features that are highly invariant and discriminant (Gu et al., 2018). The success of deep CNNs in recognizing objects has encouraged recent works to combine boosting together with deep CNNs. Brahimi & Aoun (Brahimi et al., 2019) propose a new Boosted Convolutional Neural Network architecture, which uses a very deep convolutional neural network reinforced by adding Boosted Blocks. The Boosted Blocks employed consist of a succession of convolutional layers boosted by using a Multi-Bias Nonlinear Activation function. Nevertheless, the architecture of the proposed Boosted convolutional neural network is fixed; it can not dynamically change the number of Boosted Blocks based on a given dataset. Another attempt at combining deep CNNs and boosting is boosted sampling (Berger et al., 2018), which uses posterior error maps, generated throughout training, to focus sampling on different regions, resulting in a more informative loss. However, boosted sampling applies boosting on selecting samples and treats deep CNN as a black box to make a prediction. To enrich the usage of the information generated by deep CNNs, Lee & Chen (Lee et al., 2018) propose a new BoostCNN structure which employs a trained deep convolutional neural network model to extract the features of the images, and then, use the AdaBoost algorithm to assemble the Softmax classifiers. However, how to combine different sets of the features extracted is unclear and the computational cost is expensive when training several deep CNNs at the same time. To tackle this problem, Han & Meng (Han et al., 2016) propose Incremental Boosting CNN (IB-CNN) to integrate boosting into the CNN via

an incremental boosting layer that selects discriminative neurons from a lower layer and is incrementally updated on successive mini-batches. Different from IB-CNN which only involves one deep CNN, BoostCNN (Brahimi et al., 2016) incorporates boosting weights into the neural network architecture based on least squares objective functions, which leads to the aggregation of several CNNs. However, the computational and memory demand of BoostCNN is high when the weak learner is not simple. All these works assume all features for training weak learners, which is a big difference with our work.

Boosting for Recurrent Neural Network (RNN) and Transformer: RNN, long short-term memory (LSTM) and transformers have been firmly established as state of the art approaches in sequence modeling and transduction problems such as language modeling and machine translation ((Bahdanau et al., 2015), (Cho et al., 2014), (Sutskever et al., 2014), (Vaswani et al., 2017)). Some efforts have been made in combining boosting with RNN or LSTM. Chen & Lundberg (Chen et al., 2018) present feature learning via LSTM networks and prediction via gradient boosting trees (XGB). More precisely, they generate features by performing supervised representation learning with an LSTM network, then augment the original XGB model with these new generated features. However, the selection of the features from LSTM is not determined by XGB, which leads to a disconnect between LSTM and XGB. Another attempt at combining boosting and RNN is the boosting algorithm for regression with RNNs (Assaad et al., 2008). This algorithm adapts an ensemble method to the problem of predicting future values of time series using RNNs as base learners, and it is based on the boosting algorithm where different points of the time series are emphasized during the learning process by training different base learners on different subsets of time points. However, no work has been done about combing boosting and transformers. Although, analyses of attention in Transformer have been explored

(Clark et al., 2019), very limited work is launched about the usage of the attention distribution in token selection.

Given the fact that importance sampling improves the performance by prioritizing training samples, importance sampling has been well studied, both theoretically and empirically, in standard stochastic gradient descent settings (Needell et al., 2014) (Zhao and Zhang, 2015), in deep learning settings (Katharopoulos and Fleuret, 2018), and in minibatches (Csiba and Richtárik, 2018). As stated in these papers, importance sampling theoretically improves the convergence rate and is experimentally effective in reducing the training time and training loss. However, no generalization work has been done in a boosting setting.

4.3. Algorithms for CNN as Weak Learner

In this section, we provide a summary of BoostCNN and propose a new algorithm, subgrid CNN, which combines BoostCNN and the subgrid trick.

4.3.1. Background: Standard BoostCNN

We start with a brief overview of multiclass boosting. Given a sample $x_i \in \mathcal{X}$ and its class label $z_i \in \{1, 2, \cdots, M\}$, multiclass boosting is a method that combines several multiclass predictors $g_t : \mathcal{X} \to \mathbb{R}^d$ to form a strong committee $f(x)$ of classifiers, i.e. $f(x) = \sum_{t=1}^{N} \alpha_t g_t(x)$ where g_t and α_t are the weak learner and coefficient selected at the t^{th} boosting iteration. There are various approaches for multiclass boosting such as (Hastie et al., 2009), (Mukherjee and Schapire, 2013), (Saberian and Vasconcelos, 2011); we use the GD-MCBoost method of (Saberian and Vasconcelos, 2011), (Brahimi et al., 2016) herein. For simplicity, in the rest of the paper, we assume that $d = M$.

Standard BoostCNN (Brahimi et al., 2016) trains a boosted predictor $f(x)$ by minimizing the risk of classification

$$\mathcal{R}[f] = \mathrm{E}_{X,Z}\left[L(z, f(x))\right] \approx \frac{1}{|\mathcal{D}|} \sum_{(x_i,z_i)\in\mathcal{D}} L(z_i, f(x_i)), \tag{4.1}$$

where $\mathcal{D}$ is the set of training samples and

$$L(z, f(x)) = \sum_{j=1, j\neq z}^{M} e^{\frac{1}{2}[\langle y_z, f(x)\rangle - \langle y_j, f(x)\rangle]},$$

given $y_k = \mathbb{1}_k \in \mathbb{R}^M$, i.e. the k^{th} unit vector. The minimization is via gradient descent in a functional space. Standard BoostCNN starts with $f(x) = \mathbf{0} \in \mathbb{R}^d$ for every x and iteratively computes the directional derivative of risk (4.1), for updating $f(x)$ along the direction of $g(x)$

$$\begin{aligned}\delta\mathcal{R}[f; g] &= \left.\frac{\partial\mathcal{R}[f + \epsilon g]}{\partial\epsilon}\right|_{\epsilon=0} = -\frac{1}{2\,|\mathcal{D}|} \sum_{(x_i,z_i)\in\mathcal{D}} \sum_{j=1}^{M} g_j(x_i) w_j(x_i, z_i) \\ &= -\frac{1}{2\,|\mathcal{D}|} \sum_{(x_i,z_i)\in\mathcal{D}} g(x_i)^T w(x_i, z_i),\end{aligned} \tag{4.2}$$

where

$$w_k(x, z) = \begin{cases} -e^{-\frac{1}{2}[f_z(x) - f_k(x)]}, & k \neq z \\ \sum_{j=1, j\neq k}^{M} e^{-\frac{1}{2}[f_z(x) - f_j(x)]}, & k = z. \end{cases} \tag{4.3}$$

Then, standard BoostCNN selects a weak learner g^* that minimizes (4.2), which essentially measures the similarity between the boosting weights $w(x_i, z_i)$ and the function values $g(x_i)$. Therefore, the optimal network output $g^*(x_i)$ has to be proportional to the boosting weights, i.e.

$$g^*(x_i) = \beta w(x_i, z_i), \tag{4.4}$$

for some constant $\beta > 0$. Note that the exact value of β is irrelevant since $g^*(x_i)$ is scaled when computing α^*. Consequently, without loss of generality, we assume $\beta = 1$ and convert the problem to finding a network $g(x) \in \mathbb{R}^M$ that minimizes the square error loss

$$\mathcal{L}(w, g) = \sum_{(x_i, z_i) \in \mathcal{D}} \|g(x_i) - w(x_i, z_i)\|^2 . \tag{4.5}$$

After the weak learner is trained, BoostCNN applies a line search to compute the optimal step size along g^*,

$$\alpha^* = \operatorname*{argmin}_{\alpha \in \mathbb{R}} \mathcal{R}[f + \alpha g^*]. \tag{4.6}$$

Finally, the boosted predictor $f(x)$ is updated as $f = f + \alpha^* g^*$.

4.3.2. Subgrid BoostCNN

When considering full-size images, BoostCNN using complex CNNs as weak learners is time-consuming and memory hungry. Consequently, we would like to reduce the size of the images to lower the running time and the memory requirement. A straightforward idea would be downsizing the images directly. A problem of this approach is that the noise would possibly spread out to later learners since a strong signal could be weakened during the downsize process. Another candidate for solving the aforementioned problem is randomly selecting pixels from the original images, however, the fluctuation of the performance of the algorithm would be significant especially when the images are sharp or have a lot of noise. In this paper, we apply the subgrid trick to each weak learner in BoostCNN. The remaining question is how to select a subgrid for each weak learner. Formally, a subgrid is defined by deleting a subset of rows and columns.

Moreover, the processed images may not have the same size between iterations, which in turn requires that the new BoostCNN should allow each weak learner to have a different architecture. Furthermore, how to pass the weak learner model parameters from the previous weak learner to the current weak learner is unclear if they have different architectures.

In order to address these issues, we first separate a standard deep CNN into two parts. We name all the stacks of layers such as convolutional layers and pooling layers, except the last fully-connected (FC) layers, as the feature extractor. In the meanwhile, we name the last FC layers as the classifier. Furthermore, we refer to g_0 as the basic weak learner and all the succeeding g_t's as the additive weak learners. Subgrid BoostCNN defines an importance index for each pixel (j, k) in the image as

$$I_{j,k} = \frac{1}{|\mathcal{D}|} \sum_{(x_i,z_i)\in\mathcal{D}} \sum_{c\in C} \left| \frac{\partial \mathcal{L}(w, g)}{\partial x_i^{j,k,c}} \right|, \tag{4.7}$$

where $x_i^{j,k,c}$ denotes the pixel (j, k) in channel c from the sample x_i and C represents the set of all channels, i.e., for general image datasets, $C = \{1, 2, 3\}$. The importance index of a row, column is a summation of the importance indexes in the row, column divided by the number of columns, rows, respectively. This importance index is computed based on the residual of the current predictor. Therefore, a larger importance value means a larger adjustment is needed for this pixel at the current iterate. The algorithm uses the importance index generated based on the feature extractor of the incumbent weak learner and the classifier from g_0 to conduct subgrid selection. The selection strategy we apply in the algorithm is deleting less important columns and rows, which eventually provides the important subgrid. After the subgrid is selected, subgrid BoostCNN creates a new tensor x_i^t at iterate t, and then feeds it into an appropriate feature

extractor followed by a proper classifier. The modified minimization problem becomes

$$\mathcal{L}(w, g) = \sum_{(x_i, z_i) \in \mathcal{D}} \left\| g(x_i^t) - w(x_i, z_i) \right\|^2, \tag{4.8}$$

where the modified boosting classifier is

$$f(x) = \sum_{t=1}^{N} \alpha_t g_t(x^t). \tag{4.9}$$

In this way, subgrid BoostCNN dynamically selects important subgrids based on the updated residuals. Moreover, subgrid BoostCNN is able to deal with inputs of different sizes by applying different classifiers. Furthermore, we are allowed to pass the feature extractor's parameters from the previous weak learner since the feature extractor is not restricted to the input size. The proposed algorithm (subgrid BoostCNN) is summarized in Algorithm 6.

Algorithm 6 subgrid BoostCNN

1: **Inputs:**

number of classes M, number of boosting iterations N_b, shrinkage

parameter ν, dataset $\mathcal{D} = \{(x_1, z_1), \cdots, (x_n, z_n)\}$ where

$z_i \in \{1, \cdots, M\}$ is the label of sample x_i, and $0 < \sigma < 1$

2: **Initialize:**

set $f(x) = \mathbf{0} \in \mathbb{R}^M$, $P_0 = \{(j,k)|(j,k) \text{is a pixel in } x_i\}$

3: compute $w(x_i, z_i)$ for all (x_i, z_i), using (4.3)

4: train a deep CNN g_0^* to optimize (4.5)

5: $f(x) = g_0^*$

6: **for** t = $1, 2, \cdots, N_b$ **do**

7: update importance index $I_{j,k}$ for $(j,k) \in P_{t-1}$, using (4.7)

8: select the subgrid based on σ fraction of rows and columns with highest importance index and let P_t be the set of selected pixels; form a new tensor x_i^t for each sample i

9: construct a new proper weak learner architecture

10: compute $w(x_i, z_i)$ for all i, using (4.3) and (4.9)

11: train a deep CNN g_t^* to optimize (4.8)

12: find the optimal coefficient α_t, using (4.6) and (4.9)

13: $f(x) = f(x) + \nu\alpha_t g_t^*$

14: **end for**

Subgrid BoostCNN starts by initializing $f(x) = \mathbf{0} \in \mathbb{R}^M$. The algorithm first generates a full-size deep CNN as the basic weak learner, which uses the full image in steps 3-4. After the basic weak learner g_0^* is generated, in each iteration, subgrid BoostCNN first updates

the importance index $I_{j,k}$ for each pixel (j, k), which has been used in the preceding iterate at step 7. In order to mimic the loss of the full-size image, although we only update the importance indexes for the pixels which have been used in the last iterate, we feed the full-size tensor to the deep CNN g to compute the importance index. The deep CNN g used in (4.7) to compute the importance value is constructed by copying the feature extractor from the preceding weak learner followed by the classifier in the basic weak learner g_0^*. Next, by deleting less important rows and columns based on $I_{j,k}$, which contain $1 - \sigma$ fraction of pixels, it finds the most important subgrid having σ fraction of pixels at position P_t based on the importance index $I_{j,k}$, and forms a new tensor x_i^t in step 8. Note that P_t is not necessary to be a subset of P_{t-1} and actually is rarely to be a subset of P_{t-1}. This only happens when the highest importance index at iterate t is also the highest score at iterate $t - 1$. Next, a new additive weak learner is initialized by borrowing the feature extractor from the preceding weak learner g_{t-1}^* followed by a randomly initialized FC layer with the proper size in step 9. Once the additive weak learner is initialized, subgrid BoostCNN computes the boosting weights, $w(x) \in \mathrm{R}^M$ according to (4.3) and (4.9), trains a network g_t^* to minimize the squared error between the network output and boosting weights using (4.8), and finds the boosting coefficient α_t by minimizing the boosting loss (4.6) in steps 10-12. Lastly, the algorithm adds the network to the ensemble according to $f(x) = f(x) + \nu\alpha_t g_t^*$ for $\nu \in [0, 1]$ in step 13.

4.4. Algorithms for Transformer as Weak Learner

In this section, we propose three algorithms combining boosting and transformers from different perspectives. We assume BERT like bidirectional transformer classifier (Devlin et al., 2019) (Liu et al., 2019). The first token of each sequence is a special classification token, and

the corresponding final hidden state output of this token is used as the aggregated representation for the classification.

4.4.1. Standard BoostTransformer

Inspired by BoostCNN, we propose BoostTransformer which combines boosting and transformers (encoder) together. For a sequence classification problem, we are given a sample $x_i \in \mathcal{X}$, which contains a sequence of tokens, and its class label $z_i \in \{1, 2, \cdots, M\}$. The risk function, the functional gradient and the optimal boosting coefficient α^t are exactly the same as those in (4.1), (4.2), and (4.6), respectively. The algorithm follows standard gradient boosting machine.

4.4.2. Subsequence BoostTransformer

Combining the subgrid trick and BoostTransformer means applying the subgrid trick to each weak learner in BoostTransformer. Different from deep CNNs, transformers are able to deal with sequences of any length, thus, there is no issue when transferring information from the current weak learner to the succeeding weak learner. Similar to subgrid BoostCNN, we denote g_0 as the basic weak learner, which deals with the whole dataset, and all the succeeding g_t's as the additive weak learners. Moreover, subsequence BoostTransformer defines an importance index for each token w in the vocabulary based on the attention distribution. More precisely, the importance value of token w is computed by adding two parts; the first part is the importance of the token w itself, and the second part is the importance of token w to the remaining tokens in the same sample. In an L-layer transformer for a sequence x of length s (following (Devlin et al., 2019) we assume that the first token in x is a bogus token, which indicates that the corresponding token in the final layer is used as the embedding for classification), and positions

$1 \leq i, j \leq s$, and layer k for $1 \leq k \leq L$, let the attention from position i to position j between layer $k-1$ and k be donated by $a(i, j; k; x)$. We have $\sum_{j=1}^{s} a(j, i; k; x) = 1$ for every i, k, x. Then, given a transformer with L layers, the self-importance of token w in position p in a sample x_i is

$$I^S(w, x_i) = \left(\prod_{k=1}^{L-1} a(p, p; k; x_i) \right) \cdot a(p, 1; L; x_i) \approx a(p, 1; L; x_i), \tag{4.10}$$

The importance of token w to others is

$$I^R(w, x_i) = \left[\prod_{k=1}^{L-1} \max_{j, j \neq p} a(p_{k-1}, j; k; x_i) \right] \cdot a(p_{L-1}, 1; L; x_i), \tag{4.11}$$

where $p_{k-1} = \text{argmax}_{j, j \neq p} a(p_{k-2}, j; k-1; x_i)$ for $k = 2, 3, \cdots, L-1$, and $p_0 = p$. The first term computes the product of the maximum attention values through the path which does not contain p until the second to last layer. For the second term, as it has been shown in (Devlin et al., 2019), the classification layer only takes the 1st position of the last transformer layer which is corresponding to the classification token, therefore, the formula in (4.11) does not check all possible attention distributions; instead, it counts the attention value from the position p_{L-1} to the 1st position in the last transformer layer directly. After the aforementioned importance values are computed, the importance value of the word w is

$$I(w) = \sum_{x_i, w \in x_i} \left(I^S(w, x_i) + I^R(w, x_i) \right). \tag{4.12}$$

Then, the algorithm uses the importance index to select the most important tokens. After the tokens are selected, subsequence BoostTransformer creates a new sample x_i^t at iterate t, which contains only the important tokens, and is used by the weak learner. The modified minimization

problem and the boosting weak learner are explicitly presented in (4.8) and (4.9), respectively. The proposed algorithm (subsequence BoostTransformer) is summarized in Algorithm 7.

Different from standard BoostTransformer, subsequence BoostTransformer first reviews the whole dataset in steps 3-4 and generates the basic weak learner g_0^*. Once the basic weak learner is created, in each iteration, subsequence BoostTransformer first updates the attention-based importance vector I_w for any $w \in V_{t-1}$ in step 7, and selects σ fraction of the tokens to form the vocabulary set V_t, and lastly constructs a new sample x_i^t by deleting any tokens not in V_t in step 8. After the new sample x_i^t is constructed, subsequence BoostTransformer initializes the weights of the current transformer by using the weights in g_{t-1}^* and trains the transformer with x_i^t to minimize the squared error in (4.8) in steps 9-10. Lastly, the algorithm finds the boosting coefficient α_t by minimizing (4.6) in step 11 and adds the additive weak learner to the ensemble in step 12.

Algorithm 7 subsequence BoostTransformer

Inputs:

number of classes M, number of boosting iterations N_b, shrinkage parameter ν, dataset $\mathcal{D} = \{(x_1, z_1), \cdots, (x_n, z_n)\}$ where $z_i \in \{1, \cdots, M\}$ is the label of sample x_i, and $0 < \sigma < 1$

Initialize:

set $f(x) = \mathbf{0} \in \mathbb{R}^M$, $V_0 = \{w | w \in x_i \text{ for some } x_i\}$

compute $w(x_i, z_i)$ for all (x_i, z_i), using (4.3)

train a transformer g_0^* to optimize (4.5)

$f(x) = g_0^*$

for t = $1, 2, \cdots, N_b$ **do**

    update importance values I_w for $w \in V_{t-1}$, using (4.10), (4.11) and (4.12)

    select σ fraction of the tokens based on I_w, update vocabulary set V_t, and form a new sample x_i^t for each sample i

    compute $w(x_i, z_i)$ for all i, using (4.3) and (4.9)

    train a transformer g_t^* to optimize (4.8)

    find the optimal coefficient α_t, using (4.6) and (4.9)

    $f(x) = f(x) + \nu \alpha_t g_t^*$

end for

4.4.3. Importance-sampling-based BoostTransformer

Importance sampling, a strategy for preferential sampling of more important samples capable of accelerating the training process, has been well studied in stochastic gradient descent (SGD)

(Alain et al., 2015). However, there is virtually no existing work combining the power of importance sampling with the strength of boosting. Motivated by the phenomenon that overfitting appears early in standard BoostTransformer, we propose importance-sampling-based BoostTransformer, which combines importance sampling and BoostTransformer. Importance-sampling-based BoostTransformer mimics importance sampling SGD by introducing a new loss function and computing a probability distribution for drawing samples. Similarly, importance-sampling-based BoostTransformer computes a probability distribution in each iteration, and draws a subset of samples to train the weak learner based on the distribution. The probability distribution is

$$P(I = i) = \frac{\|w(x_i, z_i)\|}{\sum_{(x_j,z_j)\in\mathcal{D}} \|w(x_j, z_j)\|}, \tag{4.13}$$

which yields the new loss function for a subset of samples $\mathcal{J}$ to be

$$\bar{\mathcal{L}}_{\mathcal{J}}(w, g) = \sum_{(x_i,z_i)\in\mathcal{J}} \frac{1}{|\mathcal{D}|\, P(I = i)} \|g(x_i) - w(x_i, z_i)\|^2 . \tag{4.14}$$

We then apply any optimization algorithm with respect to (4.14) (by further using mini-batches or importance sampling). The entire algorithm is exhibited in Algorithm 8.

Algorithm 8 importance-sampling-based BoostTransformer

1: **Inputs:**

number of classes M, number of boosting iterations N_b, shrinkage

parameter ν, dataset $\mathcal{D} = \{(x_1, z_1), \cdots, (x_n, z_n)\}$ where

$z_i \in \{1, \cdots, M\}$ is the label of sample x_i, and $0 < \sigma < 1$

2: **Initialize:**

set $f(x) = \mathbf{0} \in \mathbb{R}^M$

3: compute $w(x_i, z_i)$ for all x_i, using (4.3)

4: train a transformer g_0^* to optimize (4.5)

5: $f(x) = g_0^*$

6: **for** t = $1, 2, \cdots, N_b$ **do**

7: compute probability distribution P_t, using (4.13)

8: draw independently $|\mathcal{I}^t|$ samples, which is σ fraction of the samples, based on P_t

9: compute $w(x_i, z_i)$ for $(x_i, z_i) \in \mathcal{I}^t$, using (4.3)

10: train a transformer g_t^* to optimize (4.14) on $\mathcal{I}^t$

11: find the optimal coefficient α_t, using (4.6) on $\mathcal{I}^t$

12: $f(x) = f(x) + \nu\alpha_t g_t^*$

13: **end for**

Importance-sampling-based BoostTransformer starts with learning the full-size dataset and training a basic weak learner in steps 3-4. In each iteration, the algorithm first computes the probability distribution P_t in step 7 and selects a subset $\mathcal{I}^t$ of samples based on the distribution in step 8. Once the dataset is created, it computes the weights and trains a transformer by

using the unbiased loss function (4.14), following by finding an optimal boosting coefficient in steps 9-12.

In the rest of this section, we provide all analysis of the optimal probability distribution in importance-sampling-based BoostTransformer. Given current aggregated classifier $f_{t-1} = \sum_{i=1}^{t-1} g_i^*$, let us define the expected training progress attributable to iteration t as

$$\mathbb{E}_{P_t}\left[\Delta^{(t)}\right] = \|f_{t-1} - f^*\|^2 - \mathbb{E}_{P_t}\left[\|f_t - f^*\|^2 \middle| \mathcal{F}^{t-1}\right].$$

Here f^* denotes the solution to (4.1), and the expectation is taken over the probability distribution P_t, and $\mathcal{F}^{t-1}$ contains the whole history of the algorithm up until iterate $t-1$. We assume that gradient sampling is unbiased. Inspired by the work in (Zhao and Zhang, 2015), we prove that the optimal probability distribution is proportional to the boosting weight at each iteration.

Theorem 13. *In* $\max_{P_t} \mathbb{E}_{P_t}\left[\Delta^{(t)}\right]$*, the optimal distribution for importance-sampling-based BoostTransformer to select each sample* i *is proportional to its "boosting weight norm:"*

$$P(I = i) = \frac{\|w(x_i, z_i)\|}{\sum_{(x_j, z_j)\in\mathcal{D}} \|w(x_j, z_j)\|}. \tag{4.15}$$

PROOF. See Appendix A. □

Based on the fact that the new loss function with respect to the probability distribution is unbiased, we discover that maximizing the improvement of the boosting algorithm is equivalent to minimizing the functional gradient variance. By applying Jensen's inequality, the optimal probability distribution is essentially proportional to the boosting weights, which are easy to obtain, in boosting algorithms.

4.5. Experimental Study

In this section, we first compare subgrid BoostCNN with standard BoostCNN and deep CNNs, next, we compare the standard transformer, BoostTransformer, subsequence BoostTransformer and importance-sampling-based BoostTransformer in the second half of the section. We conduct experiments on three different datasets for both CNN related and transformer related algorithms. From all of these datasets, we study the performance of the boosting technique, the subgrid trick and the importance sampling strategy. All the algorithms are implemented in Python with PyTorch (Paszke et al., 2017). Training is conducted on an NVIDIA Titan XP GPU.

4.5.1. Image

In this subsection, we illustrate properties of the proposed subgrid BoostCNN and compare its performance with other methods on several image classification tasks. In subgrid BoostCNN, the risk function (4.1) we employ is cross entropy, and the input of each weak learner is an image with 3 channels which can be handled by standard Conv2d functions in PyTorch. Meanwhile, we implement the subgrid strategy based on (4.7) with respect to each pixel (j, k). We delete approximately 10% of the rows and columns, which leads to $\sigma = 81\%$, and fix the shrinkage parameter ν to be 0.02. In each weak learner, we apply the ADAM algorithm with the learning rate of 0.0001 and weight decay being 0.0001.

We consider CIFAR-10 (Krizhevsky et al., 2009), SVHN (Netzer et al., 2011) and ImageNetSub (Deng et al., 2009) datasets as shown in Table 4.1. For the last dataset, since the original ImageNet dataset is large and takes significant amount of time to train, we select a subset of samples from the original ImageNet dataset. More precisely, we randomly pick 100

labels and select the corresponding samples from ImageNet, which consists of $124,000$ images for training and $10,000$ images for testing. We denote it as ImageNetSub. Data preprocessing consists of three steps: 1. random resizing and cropping with output size 224×224, scale uniformly sampled from [0.08, 1.0] and make the aspect ratio uniformly sampled from [0.75, 1.33]; 2. random horizontal flipping with flipping probability 0.5; 3. normalization for each channel.

	Number of Training/Testing Samples	Number of Classes
CIFAR-10	50k/10k	10
SVHN	73k/26k	10
ImageNetSub	124k/10k	100

Table 4.1. Image Datasets

For training, we employ three different deep CNNs, which are ResNet-18, ResNet-50 and ResNet-101. For each combination of dataset/CNN, we first train the deep CNN for a certain number of epochs, and then initialize the weights in the basic weak learner for the boosting algorithms as the weights in the deep CNN. In the subgrid BoostCNN experiments, we use 10 CNN weak learners. We train each weak learner for 15 epochs which has been calibrated. For comparison, we train BoostCNN, the ensemble method (without boosting weight update and always using all features) denoted by e-CNN and the subgrid ensemble method named as subgrid e-CNN (without boosting weight update in step 10 in Algorithm 6) for 10 iterates as well. Notice that subgrid e-CNN essentially mimics random forests. We also train the single deep CNN for the same number of 150 epochs.

We start by applying ResNet-18 as our weak learner for all different ensemble methods. Figures 4.1, 4.3 and 4.5 compare the relative performances with respect to single ResNet-18 vs the running time. The solid lines in green and yellow show the relative performances of BoostCNN and subgrid BoostCNN, respectively, while the dotted lines in green and yellow represent the relative performances of e-CNN and subgrid e-CNN, respectively. As shown in these figures, taking the same amount of time, subgrid BoostCNN outperforms all of the remaining algorithms. Furthermore, we observe that subgrid BoostCNN outperforms BoostCNN, and subgrid e-CNN has the same behavior when compared with e-CNN. In conclusion, the subgrid technique improves the performance of the boosting algorithm. Moreover, Figures 4.2, 4.4 and 4.6 depict subgrid BoostCNN and subgrid e-CNN using three different seeds with respect to their averages. The solid and dotted lines in the same color represent the same seed used in corresponding subgrid BoostCNN and subgrid e-CNN. As the figures show, the solid lines are closer to each other than the dotted lines, which indicates that subgrid BoostCNN is more robust with respect to the variation of the seed when compared with subgrid e-CNN. Furthermore, the standard deviations of the accuracy generated by subgrid e-CNN and subgrid BoostCNN are shown in Table 4.2. The standard deviations of the accuracy generated by subgrid e-CNN are significant compared to those of subgrid BoostCNN, which in turn indicates that subgrid BoostCNN is less sensitive to the choice of the seed. Therefore, subgrid BoostCNN is more robust than subgrid e-CNN.

	subgrid BoostCNN	subgrid e-CNN
CIFAR-10	0.478	2.519
SVHN	0.385	0.891
ImageNetSub	2.489	7.915

Table 4.2. Standard deviation times 10^3 of the accuracy results by different seeds

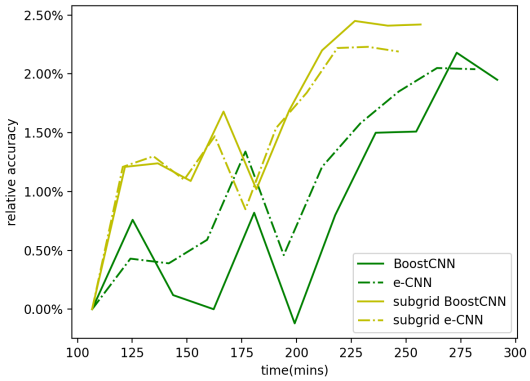


Figure 4.1. ResNet-18 on CIFAR-10

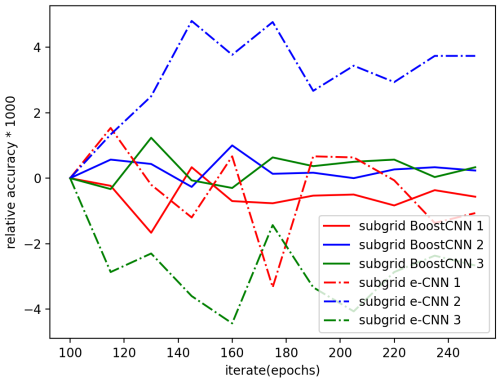


Figure 4.2. Different Seeds

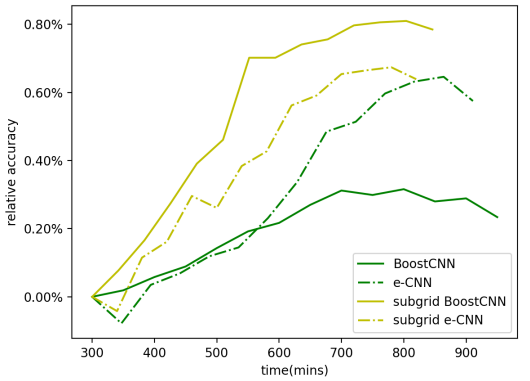


Figure 4.3. ResNet-18 on SVHN

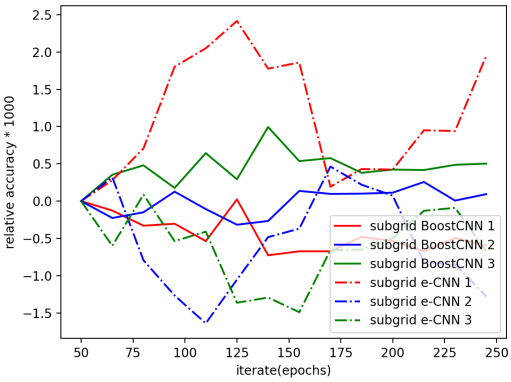


Figure 4.4. Different Seeds

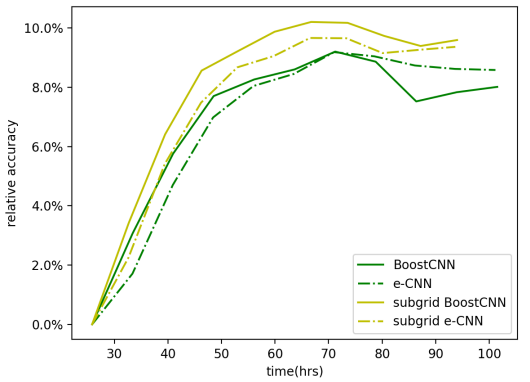


Figure 4.5. ResNet-18 on ImageNetSub

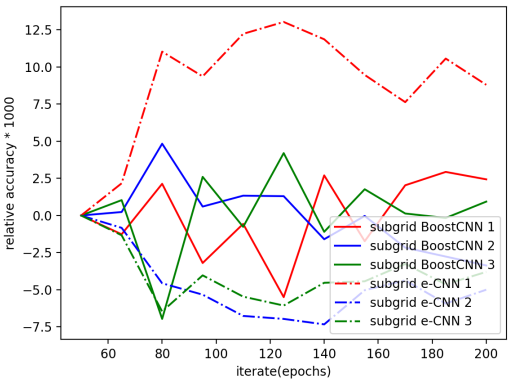


Figure 4.6. Different Seeds

Next, we evaluate relative performances of subgrid BoostCNN using ResNet-50 as the weak learner on CIFAR-10 and ImageNetSub datasets with respect to the single ResNet-50. We do not evaluate the relative performances on the SVHN dataset since the accuracy of the single ResNet-50 on the SVHN dataset is over 98%. From Figures 4.7 and 4.9, we also observe the benefits of the subgrid technique. Besides, Figures 4.8 and 4.10 confirm that subgrid BoostCNN is more stable than subgrid e-CNN since the solid series are closer to each other compared with the dotted series. Furthermore, we establish the relative performances of subgrid BoostCNN using ResNet-50 as the weak learner with respect to the single ResNet-101 in Figure 4.11. Although single ResNet-101 outperforms single ResNet-50, subgrid BoostCNN using ResNet-50 as the weak learner outperforms single ResNet-101 significantly in Figure 4.11, which indicates that subgrid BoostCNN with a simpler CNN is able to exhibit a better performance than a single deeper CNN. Lastly, we conduct experiments with ResNet-101 on the ImageNetSub dataset. From Figure 4.12, we not only discover the superior behaviors of BoostCNN, e-CNN, subgrid BoostCNN and subgrid e-CNN over ResNet-101 as we expect, but also observe the benefit of the subgrid technique.

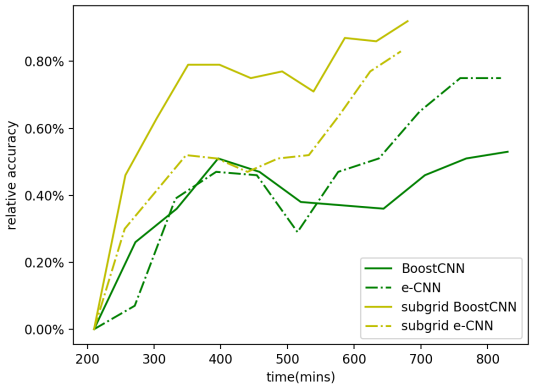


Figure 4.7. ResNet-50 on CIRFAR-10

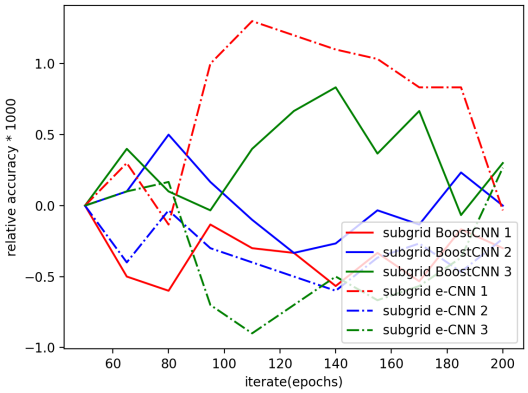


Figure 4.8. Different Seeds

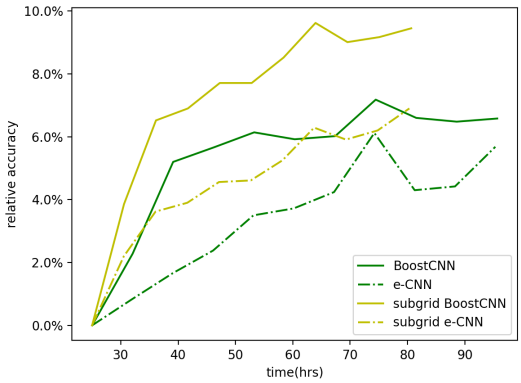


Figure 4.9. ResNet-50 on ImageNetSub

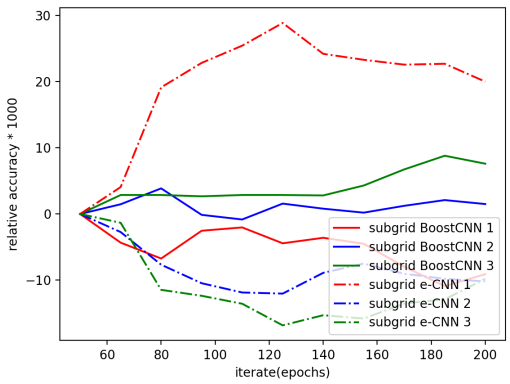


Figure 4.10. Different Seeds

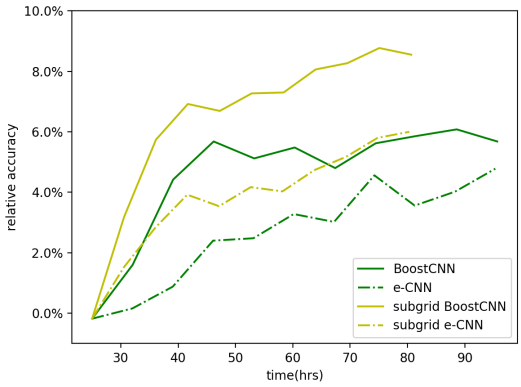


Figure 4.11. ResNet-50 on ImageNetSub compared to ResNet-101

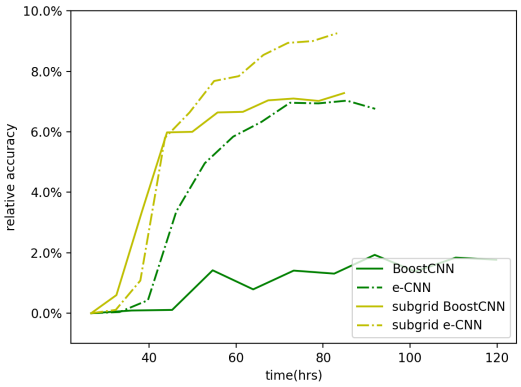


Figure 4.12. ResNet-101 on ImageNetSub

4.5.2. Text

In this section, we explore properties of the proposed Boost Transformer, subsequence Boost Transformer and importance-sampling-based Boost Transformer, and compare their performances with other methods on several text classification tasks. In the following experiments, the weak learner used is Roberta-based (Liu et al., 2019) from the HuggingFace library with only word embeddings to be pre-trained weights. Using transformer based boosting algorithms, we train an ensemble of 6 transformers each with 5 epochs (these numbers yield good performance). In subsequence BoostTransformer, we pick the most important 80% of the tokens in the vocabulary and reconstruct the dataset based on this new vocabulary. In importance-sampling-based BoostTransformer, the first flavor, in each iteration, we select 80% of the samples based on the probability distribution in (4.13) without further subsequence technique. In subsequence importance-sampling-based BoostTransformer, we first select 80% of the samples based on the probability distribution in (4.13), and then pick the most important 80% of the tokens in the current vocabulary given by the selected 80% samples, after that, we reconstruct the dataset based on this modified vocabulary. For comparison, we train the vanilla transformer and subsequence transformer, which randomly removes 20% of the tokens and trains the network on the dataset for 30 epochs. To train the model, we use AdamW (Loshchilov and Hutter, 2019) with learning rate 10^{-5}, weight decay 0.01 and batch size 16. We use linear learning rate decay with warmup ratio 0.06.

We start by presenting the three public datasets used: IMDB (Maas et al., 2011), Yelp polarity reviews and Amazon polarity reviews (McAuley and Leskovec, 2013). The IMDB dataset, which is for binary sentiment classification, contains a set of 25,000 highly polar movie reviews for training, and 25,000 for testing. The Yelp polarity reviews dataset, which is a subset

of the dataset obtained from the Yelp Dataset Challenge in 2015, consists of $100,000$ training samples and $38,000$ testing samples. The classification task for this dataset is predicting a polarity label by considering stars 1 and 2 negative, and 3 and 4 positive for each review text. The last dataset we use is the Amazon polarity reviews dataset, which is a subset of the original Amazon reviews dataset from the Stanford Network Analysis Project (SNAP). Dealing with the same classification task as the Yelp polarity review dataset, the Amazon polarity reviews dataset contains $100,000$ training samples and $25,000$ testing samples. The subsampled datasets are standard, i.e. we did not create our own subsamples. Empirically we found that a weak learner with 6 heads and 6 layers achieves good robust performance.

Given the architecture of the weak learner, we start by discussing experiments on IMDB. In Figure 4.13, we compare the relative performances of the algorithms with respect to the vanilla transformer. As it shows, all versions of BoostTransformer do not perform as good as the standard transformer and subsequence transformer in the first few epochs. However, they catch up quickly and dominate the performance in the remaining training epochs. Even more, based on Figure 4.14, which represents each model's relative improvement with respect to its initial weights, all versions of BoostTransformer maintain their performances as the number of epochs increases, while the performances of the standard transformer and the subsequence transformer start decreasing and fluctuating dramatically after the first few epochs, which implies that all versions of BoostTransformer are more robust than the standard and subsequence transformer.

Next, we evaluate relative performances with respect to the vanilla transformer on the Yelp and Amazon polarity review datasets. From Figures 4.15-4.18, we discover that the superior and more robust behavior of boosting algorithms over transformer is vigorous.

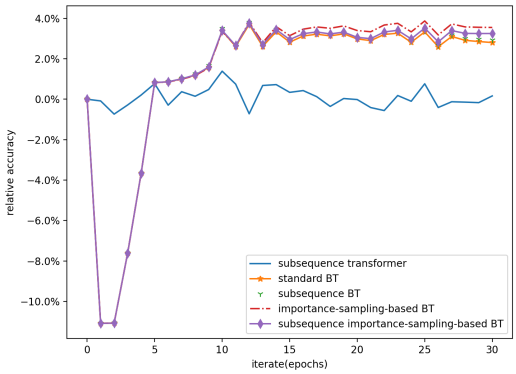


Figure 4.13. Relative Accuracy on IMDB

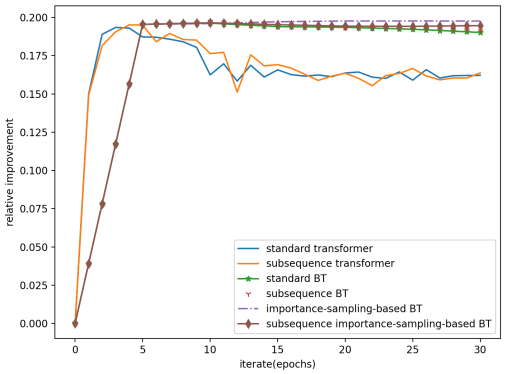


Figure 4.14. Improvement on IMDB

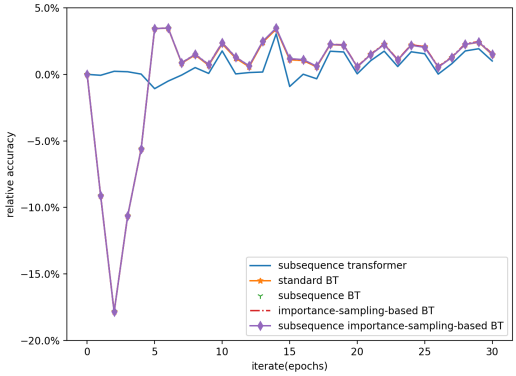


Figure 4.15. Relative Accuracy on Yelp

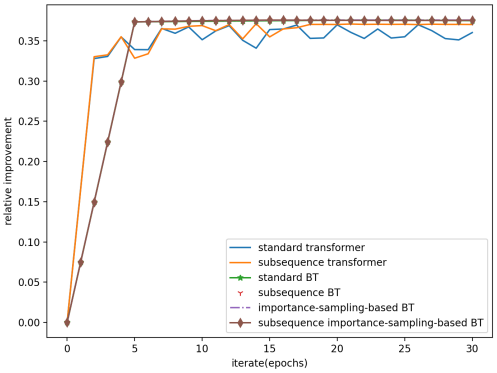


Figure 4.16. Improvement on Yelp

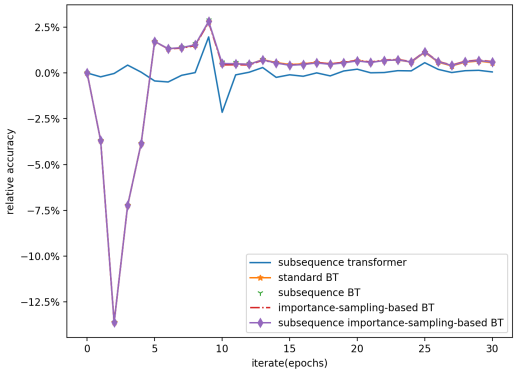


Figure 4.17. Relative Accuracy on Amazon

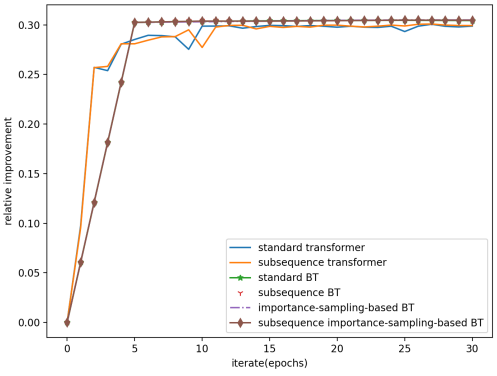


Figure 4.18. Improvement on Amazon

Furthermore, we zoom in on the performances at iterates larger than 2. In Figures 4.19-4.21, compared with the standard BoostTransformer, we observe that the subsequence BoostTransformer, importance-sampling-based BoostTransformer and subsequence importance-sampling-based BoostTransformer demonstrate a superior performance. Therefore, we conclude that the subsequence and importance sampling techniques are beneficial for the boosting algorithms. Moreover, we observe that the importance-sampling-based BoostTransformer gradually improves its performance and maintains its performance later on, while the subsequence BoostTransformer hits its best accuracy in early epochs and then starts fluctuating and decaying. The gap between the importance-sampling-based BoostTransformer and the subsequence BoostTransformer is more significant on the IMDB dataset, which has a much smaller size than the Yelp and Amazon polarity review datasets. For the subsequence importance-sampling-based BoostTransformer, compared to the subsequence BoostTransformer, although the subsequence importance-sampling-based BoostTransformer does not fluctuate and decrease as much as the subsequence BoostTransformer, which is more obvious in a small dataset (i.e. the IMDB dataset), its best accuracy is lower than that of the subsequence BoostTransformer, which is more obvious in larger datasets (i.e. the Yelp and Amazon datasets). On the other hand, compared to the importance-sampling-based BoostTransformer, although the subsequence importance-sampling-based BoostTransformer obtains its best accuracy earlier than the importance-sampling-based BoostTransformer, its overall performance fluctuates while the importance-sampling-based BoostTransformer keeps increasing and maintains its high-quality performance in all of the datasets, which implies that the subsequence importance-sampling-based BoostTransformer is less stable than the importance-sampling-based BoostTransformer. In conclusion, the

subsequence BoostTransformer fits well for datasets with enough samples and the importance-sampling-based BoostTransformer is more suitable for datasets with a limited number of samples.

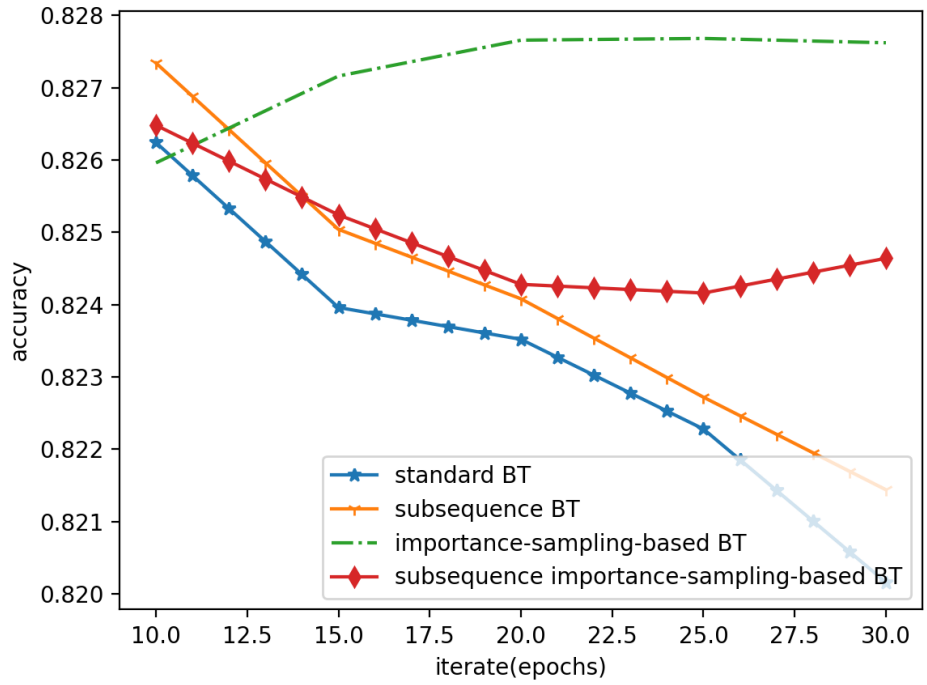


Figure 4.19. IMDB

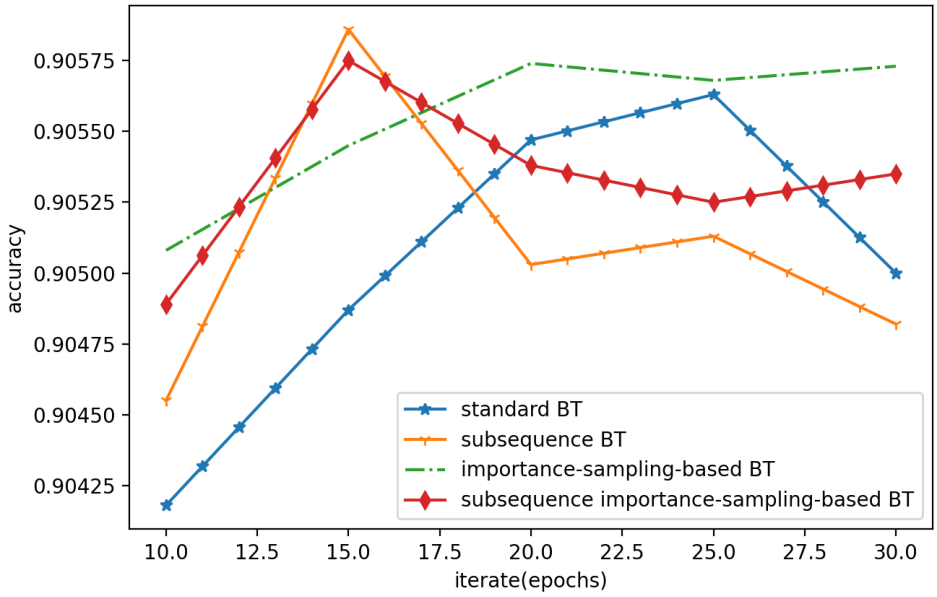


Figure 4.20. Yelp

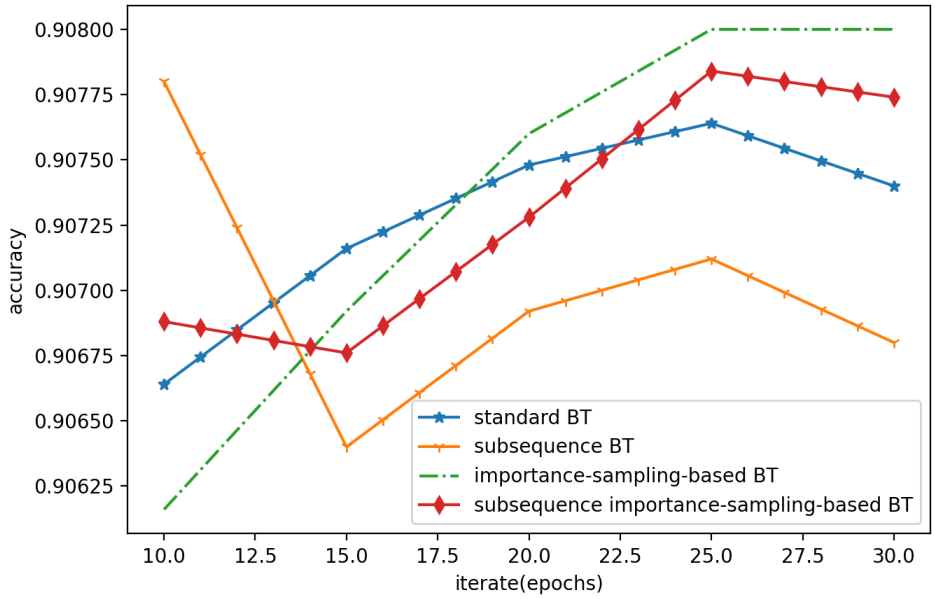


Figure 4.21. Amazon

Table 4.3 illustrates the running time of each algorithm on the different datasets in minutes. As we see in the table, the subsequence technique not only improves the performance of the boosting algorithms but also reduces the running time. Furthermore, the importance sampling technique reduces the running time significantly without hurting the performance.

	Trans.	subsequence Trans.	BT	subsequence BT	imp.-samp. -based BT	subsequence imp.-samp. -based BT
IMDB	21	14	23	17	13	12
Yelp	76	52	84	66	52	46
Amazon	75	53	79	58	46	41

Table 4.3. Running time for different algorithms

In conclusion, a subsequence transformer is a good choice if the running time cost is the most important concern, however, if accuracy performance is as crucial as the running time, then the subsequence BoostTransformer is the go-to option since it requires a slight increase in the running time but provides superior and more robust performance when compared to the subsequence transformer. In addition, if a dataset has a limited number of samples, i.e., it is easy to cause overfitting, then the importance-sampling-based BoostTransformer can outperform.

Bibliography

Jacob D Abernethy, Elad Hazan, and Alexander Rakhlin. 2012. Interior-point methods for full-information and bandit online learning. *IEEE Transactions on Information Theory*, 58(7):4164–4175.

Edoardo M. Airoldi, David M. Blei, Stephen E. Fienberg, and Eric P. Xing. 2008. Mixed membership stochastic blockmodels. *Journal of machine learning research*, 9:1981–2014.

Guillaume Alain, Alex Lamb, Chinnadhurai Sankar, Aaron C. Courville, and Yoshua Bengio. 2015. Variance reduction in SGD by distributed importance sampling. *ArXiv*, abs/1511.06481.

Mohammad Assaad, Romuald Boné, and Hubert Cardot. 2008. A new boosting algorithm for improved time-series forecasting with recurrent neural networks. *Information Fusion*, 9:41–55.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473.

Pierre Baldi and Kurt Hornik. 1989. Neural networks and principal component analysis: Learning from examples without local minima. *Neural Networks*, 2(1):53–58.

Adrian Benton, Michael J. Paul, Braden Hancock, and Mark Dredze. 2016. Collective supervision of topic models for predicting surveys with social media. In *AAAI*.

Albert Benveniste, Michel Métivier, and Pierre Priouret. 2012. *Adaptive algorithms and stochastic approximations*, volume 22. Springer Science & Business Media.

Lorenz Berger, Eoin Hyde, Matt Gibb, Nevil Pavithran, Garin Kelly, Faiz Mumtaz, and Sébastien Ourselin. 2018. Boosted training of convolutional neural networks for multi-class segmentation. *ArXiv*, abs/1806.05974.

Dimitri P Bertsekas and John N Tsitsiklis. 1989. *Parallel and distributed computation: numerical methods*, volume 23. Prentice Hall Englewood Cliffs.

Dimitri P Bertsekas and John N Tsitsiklis. 2000. Gradient convergence in gradient methods with errors. *SIAM Journal on Optimization*, 10(3):627–642.

David M. Blei and Michael I. Jordan. 2003. Modeling annotated data. In *SIGIR*.

David M. Blei and Jon D. McAuliffe. 2007. Supervised topic models. In *NIPS*.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet allocation. *Journal of machine learning research*, 3:993–1022.

Avrim Blum. 1998. On-line algorithms in machine learning. In *Online algorithms*, pages 306–325. Springer.

Vivek S Borkar. 2008. *Stochastic approximation: a dynamical systems viewpoint*. Baptism's 91 Witnesses.

Sourour Brahimi, Najib Ben Aoun, and Chokri Ben Amar. 2016. Boosted convolutional neural networks. In *BMVC*.

Sourour Brahimi, Najib Ben Aoun, and Chokri Ben Amar. 2019. Boosted convolutional neural network for object recognition at large scale. *Neurocomputing*, 330:337–354.

Hugh Chen, Scott Lundberg, and Su-In Lee. 2018. Hybrid gradient boosting trees and neural networks for forecasting operating room data. *ArXiv*, abs/1801.07384.

Jianmin Chen, Xinghao Pan, Rajat Monga, Samy Bengio, and Rafal Jozefowicz. 2016. Revisiting distributed synchronous SGD. *arXiv preprint arXiv:1604.00981*.

Tianqi Chen and Carlos Guestrin. 2016. XGBoost: A scalable tree boosting system. *Proceedings of the 22nd ACM SIGKDD*.

Xiangyi Chen, Sijia Liu, Ruoyu Sun, and Mingyi Hong. 2019. On the convergence of a class of ADAM-type algorithms for non-convex optimization. In *International Conference on Learning Representations*.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN Encoder-Decoder for statistical machine translation. In *EMNLP*.

Anna Choromanska, Mikael Henaff, Michael Mathieu, Gérard Ben Arous, and Yann LeCun. 2015a. The loss surfaces of multilayer networks. In *Artificial Intelligence and Statistics*, pages 192–204.

Anna Choromanska, Yann LeCun, and Gérard Ben Arous. 2015b. Open problem: The landscape of the loss surfaces of multilayer networks. In *Conference on Learning Theory*, pages 1756–1760.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What does BERT look at? An analysis of BERT's attention. *ArXiv*, abs/1906.04341.

Dominik Csiba and Peter Richtárik. 2018. Importance sampling for minibatches. *ArXiv*, abs/1602.02283.

Amit Daniely, Alon Gonen, and Shai Shalev-Shwartz. 2015. Strongly adaptive online learning. In *International Conference on Machine Learning*, pages 1405–1411.

Yann N Dauphin, Razvan Pascanu, Caglar Gulcehre, Kyunghyun Cho, Surya Ganguli, and Yoshua Bengio. 2014. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. In *Advances in Neural Information Processing Systems*, pages 2933–2941.

Soham De and Tom Goldstein. 2016. Efficient distributed SGD with variance reduction. In *2016 IEEE 16th International Conference on Data Mining*, pages 111–120. IEEE.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. ImageNet: A large-scale hierarchical image database. In *CVPR*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*.

Mattia A Di Gangi, Matteo Negri, and Marco Turchi. 2019. Adapting transformer to end-to-end spoken language translation. In *INTERSPEECH*.

Laura Dietz, Steffen Bickel, and Tobias Scheffer. 2007. Unsupervised prediction of citation influences. In *ICML*.

Linhao Dong, Shuang Xu, and Bo Xu. 2018. Speech-Transformer: A no-recurrence sequence-to-sequence model for speech recognition. In *ICASSP*.

Simon Du, Jason Lee, Yuandong Tian, Aarti Singh, and Barnabas Poczos. 2018a. Gradient descent learns one-hidden-layer CNN: Don't be afraid of spurious local minima. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1339–1348, Stockholmsmässan, Stockholm Sweden. PMLR.

Simon S. Du, Jason D. Lee, and Yuandong Tian. 2018b. When is a convolutional filter easy to learn? In *International Conference on Learning Representations*.

John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159.

Elena A. Erosheva, Stephen E. Fienberg, and John Lafferty. 2004. Mixed-membership models of scientific publications. *Proceedings of the National Academy of Sciences of the United States of America*, 101:5220 – 5227.

Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT Press.

Jiuxiang Gu, Zhenhua Wang, Jason Kuen, Lianyang Ma, Amir Shahroudy, Bing Shuai, Ting Liu, Xingxing Wang, Gang Wang, and Jianfei Cai. 2018. Recent advances in convolutional neural networks. *Pattern Recognition*, 77:354–377.

Shizhong Han, Zibo Meng, Ahmed-Shehab Khan, and Yan Tong. 2016. Incremental boosting convolutional neural network for facial action unit recognition. In *NIPS*.

Reza Harikandeh, Mohamed Osama Ahmed, Alim Virani, Mark Schmidt, Jakub Konečnỳ, and Scott Sallinen. 2015. Stop wasting my gradients: Practical SVRG. In *Advances in Neural Information Processing Systems*, pages 2251–2259.

Trevor Hastie, Saharon Rosset, Ji Zhu, and Hui Zou. 2009. Multi-class AdaBoost. *Statistics and Its Interface*, 2:349–360.

Elad Hazan, Amit Agarwal, and Satyen Kale. 2007. Logarithmic regret algorithms for online convex optimization. *Mach. Learn.*, 69(2-3):169–192.

Elad Hazan and Comandur Seshadhri. 2007. Adaptive algorithms for online decision problems. In *Electronic colloquium on computational complexity (ECCC)*, volume 14.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *Proceedings of the IEEE international conference on computer vision*.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *CVPR*.

Yihui He, Ji Lin, Zhijian Liu, Hanrui Wang, Li-Jia Li, and Song Han. 2018. AMC: AutoML for model compression and acceleration on mobile devices. In *ECCV*.

Mingyi Hong, Meisam Razaviyayn, Zhi-Quan Luo, and Jong-Shi Pang. 2015. A unified algorithmic framework for block-structured optimization involving big data: With applications in machine learning and signal processing. *IEEE Signal Processing Magazine*, 33(1):57–77.

Forrest Iandola, Matt Moskewicz, Sergey Karayev, Ross Girshick, Trevor Darrell, and Kurt Keutzer. 2014. DenseNet: Implementing efficient ConvNet descriptor pyramids. *ArXiv*, abs/1404.1869.

Martin Jaggi, Virginia Smith, Martin Takác, Jonathan Terhorst, Sanjay Krishnan, Thomas Hofmann, and Michael I Jordan. 2014. Communication-efficient distributed dual coordinate ascent. In *Advances in Neural Information Processing Systems*, pages 3068–3076.

Angelos Katharopoulos and François Fleuret. 2018. Not all samples are created equal: Deep learning with importance sampling. *ArXiv*, abs/1803.00942.

Kenji Kawaguchi. 2016. Deep learning without poor local minima. In *Advances in Neural Information Processing Systems*, pages 586–594.

Halil Kilicoglu, Dongwook Shin, Marcelo Fiszman, Graciela Rosemblat, and Thomas C Rindflesch. 2012. SemMedDB: a PubMed-scale repository of biomedical semantic predications. *Bioinformatics*, 28(23):3158–3160.

Diederik P. Kingma and Jimmy Ba. 2015. ADAM: A method for stochastic optimization. *CoRR*, abs/1412.6980.

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

Simon Lacoste-Julien, Fei Sha, and Michael I. Jordan. 2008. DiscLDA: Discriminative learning for dimensionality reduction and classification. In *NIPS*.

Ni Lao and William W Cohen. 2010. Relational retrieval using a combination of path-constrained random walks. *Machine Learning*, 81(1):53–67.

Jason D. Lee, Ioannis Panageas, Georgios Piliouras, Max Simchowitz, Michael I. Jordan, and Benjamin Recht. 2017. First-order methods almost always avoid saddle points. *CoRR*, abs/1710.07406.

Jason D Lee, Max Simchowitz, Michael I Jordan, and Benjamin Recht. 2016. Gradient descent only converges to minimizers. In *Conference on Learning Theory*, pages 1246–1257.

Shin-Jye Lee, Tonglin Chen, Lun Yu, and Chin-Hui Lai. 2018. Image classification based on the boost convolutional neural network. *IEEE Access*, 6:12755–12768.

Fei-Fei Li and Pietro Perona. 2005. A Bayesian hierarchical model for learning natural scene categories. In *ICCV*.

Naihan Li, Shujie Liu, Yanqing Liu, Sheng Zhao, and Ming Liu. 2019. Neural speech synthesis with transformer network. In *AAAI*.

Yuanzhi Li and Yang Yuan. 2017. Convergence analysis of two-layer neural networks with ReLU activation. In *Advances in Neural Information Processing Systems*, pages 597–607.

Jason Liang, Elliot Meyerson, Babak Hodjat, Dan Fink, Karl Mutch, and Risto Miikkulainen. 2019. Evolutionary neural AutoML for deep learning. In *GECCO*.

Tsung-Yu Lin, Aruni RoyChowdhury, and Subhransu Maji. 2015. Bilinear CNN models for fine-grained visual recognition. In *ICCV*.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *ICLR*.

Liangchen Luo, Yuanhao Xiong, and Yan Liu. 2019. Adaptive gradient methods with dynamic bound of learning rate. In *International Conference on Learning Representations*.

Chenxin Ma, Virginia Smith, Martin Jaggi, Michael I Jordan, Peter Richtárik, and Martin Takáč. 2015. Adding vs. averaging in distributed primal-dual optimization. *arXiv preprint arXiv:1502.03508*.

Andrew Maas, Raymond E Daly, Peter T Pham, Dan Huang, Andrew Y Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *ACL*.

Jakub Mareček, Peter Richtárik, and Martin Takáč. 2015. Distributed block coordinate descent for minimizing partially separable functions. In *Numerical Analysis and Optimization*, pages 261–288. Springer.

Julian McAuley and Jure Leskovec. 2013. Hidden factors and hidden topics: understanding rating dimensions with review text. In *RecSys*.

Andrew McCallum, Xuerui Wang, and Andrés Corrada-Emmanuel. 2007. Topic and role discovery in social networks with experiments on Enron and academic email. *Journal of artificial intelligence research*, 30:249–272.

David Mimno and Andrew McCallum. 2008. Topic models conditioned on arbitrary features with Dirichlet-multinomial regression. In *UAI*.

Aryan Mokhtari, Alec Koppel, and Alejandro Ribeiro. 2016. A class of parallel doubly stochastic algorithms for large-scale learning. *arXiv preprint arXiv:1606.04991*.

Aryan Mokhtari and Alejandro Ribeiro. 2016. DSA: Decentralized double stochastic averaging gradient algorithm. *The Journal of Machine Learning Research*, 17(1):2165–2199.

Indraneel Mukherjee and Robert E Schapire. 2013. A theory of multiclass boosting. *Journal of Machine Learning Research*, 14:437–497.

Tomohiro Nakatani. 2019. Improving transformer-based end-to-end speech recognition with connectionist temporal classification and language model integration. In *INTERSPEECH*.

Alexey Natekin and Alois Knoll. 2013. Gradient boosting machines, a tutorial. *Frontiers in Neurorobotics*, 7.

Alexandros Nathan and Diego Klabjan. 2017. Optimization for large-scale machine learning with distributed features and observations. In *International Conference on Machine Learning and Data Mining in Pattern Recognition*, pages 132–146.

Deanna Needell, Rachel Ward, and Nati Srebro. 2014. Stochastic gradient descent, weighted sampling, and the randomized Kaczmarz algorithm. *Mathematical Programming*, 155:549–573.

Yurii Nesterov. 2013. *Introductory lectures on convex optimization: A basic course*, volume 87. Springer Science & Business Media.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. 2011. Reading digits in natural images with unsupervised feature learning.

Neal Parikh and Stephen Boyd. 2014. Block splitting for distributed optimization. *Mathematical Programming Computation*, 6(1):77–102.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in pytorch.

Ross J. Quinlan. 2004. Induction of decision trees. *Machine Learning*, 1:81–106.

Alexander Rakhlin and A Tewari. 2009. Lecture notes on online learning. *Draft, April*.

Sashank J. Reddi, Satyen Kale, and Sanjiv Kumar. 2018. On the convergence of ADAM and beyond. In *International Conference on Learning Representations*.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39:1137–1149.

Thomas C Rindflesch and Marcelo Fiszman. 2003. The interaction of domain knowledge and linguistic structure in natural language processing: interpreting hypernymic propositions in biomedical text. *Journal of Biomedical Informatics*, 36(6):462–477.

Alex rizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2017. Imagenet classification with deep convolutional neural networks. In *CACM*.

Herbert Robbins and Sutton Monro. 1951. A stochastic approximation method. *The Annals of Mathematical Statistics*, pages 400–407.

Michal Rosen-Zvi, Thomas L. Griffiths, Mark Steyvers, and Padhraic Smyth. 2004. The author-topic model for authors and documents. *ArXiv*, abs/1207.4169.

Mohammad J Saberian and Nuno Vasconcelos. 2011. Multiclass Boosting: Theory and algorithms. In *NIPS*.

Mark Schmidt, Nicolas Le Roux, and Francis Bach. 2017. Minimizing finite sums with the stochastic average gradient. *Mathematical Programming*, 162(1-2):83–112.

Shai Shalev-Shwartz and Shai Ben-David. 2014. *Understanding machine learning: From theory to algorithms*. Cambridge University Press.

Erik B. Sudderth, Antonio Torralba, William T. Freeman, and Alan S. Willsky. 2005. Learning hierarchical models of scenes, objects, and parts. In *ICCV*.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. *ArXiv*, abs/1409.3215.

John Tsitsiklis, Dimitri Bertsekas, and Michael Athans. 1986. Distributed asynchronous deterministic and stochastic gradient optimization algorithms. *IEEE Transactions on Automatic Control*, 31(9):803–812.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, A. Gomez, L. Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*.

Chong Wang, David M. Blei, and Fei-Fei Li. 2009. Simultaneous image classification and annotation. In *ICCV*.

Xingyu Wang, Lida Zhang, and Diego Klabjan. 2020. Keyword-based topic modeling and keyword selection. *ArXiv*, abs/2001.07866.

Xuerui Wang and Andrew McCallum. 2006. Topics over time: a non-Markov continuous-time model of topical trends. *Proceedings of the 12th ACM SIGKDD*.

Xuerui Wang, Natasha Mohanty, and Andrew McCallum. 2005. Group and topic discovery from relations and their attributes. In *NIPS*.

Markus Weimer, Sriram Rao, and Martin Zinkevich. 2010. A convenient framework for efficient parallel multipass algorithms. In *LCCC: NIPS 2010 Workshop on Learning on Cores, Clusters and Clouds*.

Papis Wongchaisuwat and Diego Klabjan. 2018. Truth Validation with Evidence. *arXiv preprint arXiv:1802.05786*.

Chenwei Wu, Jiajun Luo, and Jason D Lee. 2018. No spurious local minima in a two hidden unit ReLU network.

Ching-Feng Yeh, Jay Mahadeokar, Kaustubh Kalgaonkar, Yongqiang Wang, Duc Le, Mahaveer Jain, Kjell Schubert, Christian Fuegen, and Michael L Seltzer. 2019. Transformer-Transducer: End-to-end speech recognition with self-attention. *ArXiv*, abs/1910.12977.

Chulhee Yun, Suvrit Sra, and Ali Jadbabaie. 2019. Small ReLU networks are powerful memorizers: a tight analysis of memorization capacity. In *NIPS*.

Caoxie Zhang, Honglak Lee, and Kang Shin. 2012. Efficient distributed linear classification algorithms via the alternating direction method of multipliers. In *Artificial Intelligence and Statistics*, pages 1398–1406.

Lijun Zhang, Tie-Yan Liu, and Zhi-Hua Zhou. 2019. Adaptive regret of convex and smooth functions. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 7414–7423, Long Beach, California, USA. PMLR.

Peilin Zhao and Tong Zhang. 2015. Stochastic optimization with importance sampling for regularized loss minimization. In *ICML*.

Shiyu Zhou, Linhao Dong, Shuang Xu, and Bo Xu. 2018. Syllable-based sequence-to-sequence speech recognition with the transformer in mandarin chinese. *ArXiv*, abs/1804.10752.

Changbo Zhu and Huan Xu. 2015. Online gradient descent in function space. *CoRR*, abs/1512.02394.

Jun Zhu, Amr Ahmed, and Eric P. Xing. 2012. MedLDA: maximum margin supervised topic models. *Journal of machine learning research*, 13:2237–2278.

Martin Zinkevich. 2003. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, pages 928–936.

Martin Zinkevich, Markus Weimer, Lihong Li, and Alex J Smola. 2010. Parallelized stochastic gradient descent. In *Advances in neural information processing systems*, pages 2595–2603.

Masrour Zoghi, Tomas Tunys, Mohammad Ghavamzadeh, Branislav Kveton, Csaba Szepesvari, and Zheng Wen. 2017. Online learning to rank in stochastic click models. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, pages 4199–4208. JMLR.

APPENDIX A

Additional Experimental Details for

A.1. Problem Set-up

We study the optimization problem of minimizing

$$\min_{\omega\in\mathbb{R}^d} F(\omega) := \frac{1}{N}\sum_{i=1}^{N} f_i(x_i\omega) = \frac{1}{N}\sum_{k=1}^{P}\sum_{j=1}^{n} f_j^k\left(\sum_{q=1}^{Q}\sum_{p=1}^{P} x_j^{p,q,k}\omega_{q,\pi_q(p)}\right),$$

where the features and the observations of the data $\{(x_i, y_i)\}_{i=1}^{N}$ are split into Q and P partitions respectively, and each feature partition is further separated into P smaller divisions. We have

$$n = N/P, \quad m = d/Q, \quad \tilde{m} = d/QP,$$

$$\omega = (\omega_{11}, \omega_{12}, \cdots, \omega_{1P}, \omega_{21}, \cdots, \omega_{2P}, \cdots, \omega_{QP}).$$

A.2. Notation

Recall that in steps 9- 18, the inner loop of SODDA performs iterations on each parameter subset $\omega_{q,\pi_q(p)}$ (for $i \geq 0$):

$$\bar{\omega}_{q,\pi_q(p)}^{(i+1)} = \bar{\omega}_{q,\pi_q(p)}^{(i)} - \gamma_{t+1}\Big[\nabla_{\omega_{q,\pi_q(p)}} f^p_{j_{q,\pi_q(p)}}(x^{p,q,\pi_q(p)}_{j_{q,\pi_q(p)}}\bar{\omega}^{(i)}_{q,\pi_q(p)}) - \nabla_{\omega_{q,\pi_q(p)}} f^p_{j_{q,\pi_q(p)}}(x^{p,q,\pi_q(p)}_{j_{q,\pi_q(p)}}\tilde{w}_{q,\pi_q(p)}) + \mu^t_{q,\pi_q(p)}\Big],$$

where $j_{q,\pi_q(p)}$ is a randomly selected observation in sub-block $x^{p,q,\pi_q(p)}$. It is convenient to use the notation

$$v^{t,i} = \begin{bmatrix} \nabla_{\omega_{11}} f_{j_{11}}^{\pi_1^{-1}(1)}\left(x_{j_{11}}^{\pi_1^{-1}(1),1,1}\bar{\omega}_{11}^{t,i-1}\right) - \nabla_{\omega_{11}} f_{j_{11}}^{\pi_1^{-1}(1)}\left(x_{j_{11}}^{\pi_1^{-1}(1),1,1}\tilde{\omega}_{11}\right) \\ \nabla_{\omega_{12}} f_{j_{12}}^{\pi_1^{-1}(2)}\left(x_{j_{12}}^{\pi_1^{-1}(2),1,2}\bar{\omega}_{12}^{t,i-1}\right) - \nabla_{\omega_{12}} f_{\pi_1^{-1}(2)}^{\pi_1(2)}\left(x_{j_{12}}^{\pi_1^{-1}(2),1,2}\tilde{\omega}_{12}\right) \\ \vdots \\ \nabla_{\omega_{1P}} f_{j_{1P}}^{\pi_1^{-1}(P)}\left(x_{j_{1P}}^{\pi_1^{-1}(P),1,P}\bar{\omega}_{1P}^{t,i-1}\right) - \nabla_{\omega_{1P}} f_{j_{1P}}^{\pi_1^{-1}(P)}\left(x_{j_{1P}}^{\pi_1^{-1}(P),1,P}\tilde{\omega}_{1P}\right) \\ \nabla_{\omega_{21}} f_{j_{21}}^{\pi_2^{-1}(1)}\left(x_{j_{21}}^{\pi_2^{-1}(1),2,1}\bar{\omega}_{21}^{t,i-1}\right) - \nabla_{\omega_{21}} f_{j_{21}}^{\pi_2^{-1}(1)}\left(x_{j_{21}}^{\pi_2^{-1}(1),2,1}\tilde{\omega}_{21}\right) \\ \vdots \\ \nabla_{\omega_{2P}} f_{j_{2P}}^{\pi_2^{-1}(P)}\left(x_{j_{2P}}^{\pi_2^{-1}(P),2,P}\bar{\omega}_{2P}^{t,i-1}\right) - \nabla_{\omega_{2P}} f_{j_{2P}}^{\pi_2^{-1}(P)}\left(x_{j_{2P}}^{\pi_2^{-1}(P),2,P}\tilde{\omega}_{2P}\right) \\ \vdots \\ \nabla_{\omega_{QP}} f_{j_{QP}}^{\pi_Q^{-1}(P)}\left(x_{j_{QP}}^{\pi_Q^{-1}(P),Q,P}\bar{\omega}_{QP}^{t,i-1}\right) - \nabla_{\omega_{QP}} f_{j_{QP}}^{\pi_Q^{-1}(P)}\left(x_{j_{QP}}^{\pi_Q^{-1}(P),Q,P}\tilde{\omega}_{QP}\right) \end{bmatrix} \in \mathbb{R}^d,$$

where $\pi_q^{-1}(p) \in \{1, 2, \cdots, P\}$ is the inverse function of $\pi_q(p)$, for all q and p. With this notation, we can integrate all subsets $\omega_{q,\pi_q(p)}$ and simplify the inner loop of the SODDA as follows

$$\begin{aligned} &\omega^t \\ &\quad \bar{\omega}^{t,0} = \omega^t \\ &\quad \bar{\omega}^{t,1} = \bar{\omega}^{t,0} - \gamma_{t+1}\left(\mu^t + v^{t,1}\right) \\ &\quad \bar{\omega}^{t,2} = \bar{\omega}^{t,1} - \gamma_{t+1}\left(\mu^t + v^{t,2}\right) \\ &\quad \vdots \\ &\quad \bar{\omega}^{t,B} = \bar{\omega}^{t,B-1} - \gamma_{t+1}\left(\mu^t + v^{t,B}\right) \\ &\omega^{t+1} = \bar{\omega}^{t,B} \end{aligned} \quad .$$

In what follows corresponding assumptions hold. Lastly, we define $\mathcal{F}^t$ as the sigma algebra that measures the history of the algorithm up until iteration t.

We also introduce $f \in \hat{\mathcal{O}}(g)$ if there exists a constant $C > 0$ such that $f(x) \leq C \cdot g(x)$ for every $x \geq 0$.

A.3. Diminishing L.R. Convergence without Feature and Sample Sampling

Lemma 1. *Let $\Phi = \{\phi_1, \cdots, \phi_R\}$ be a set of random vectors measurable with respect to $\sigma-$algebra $\mathcal{H}$, let $g : \Phi \to \mathbb{R}^k$ be a measurable function, and let $\mathcal{b}$ be an integer such that $1 \leq \mathcal{b} \leq R$. Let $\mathcal{B}$ be a set of size $\mathcal{b}$ uniformly and randomly selected vectors from Φ without replacement. Given two constants w_1 and w_2, we have*

$$\mathbb{E}\left[w_1 \sum_{i \in \mathcal{B}} g(\phi_i) + w_2 \sum_{i \notin \mathcal{B}} g(\phi_i) \middle| \mathcal{H} \right] = \left(\frac{\mathcal{b}}{R} w_1 + \frac{R - \mathcal{b}}{R} w_2 \right) \sum_{i=1}^{R} g(\phi_i).$$

PROOF. Using the definition of the expectation we obtain

$$\mathbb{E}\left[w_1 \sum_{i \in \mathcal{B}} g(\phi_i) + w_2 \sum_{i \notin \mathcal{B}} g(\phi_i) \middle| \mathcal{H} \right] = \sum_{\mathcal{B}} \frac{1}{\binom{R}{\mathcal{b}}} \left[w_1 \sum_{i \in \mathcal{B}} g(\phi_i) + w_2 \sum_{i \notin \mathcal{B}} g(\phi_i) \right],$$

where the first summation indicates summation over all subsets of $\mathcal{B}$ of cardinality $\mathcal{b}$. Thus, the expected value of $w_1 \sum_{i \in \mathcal{B}} g(\phi_i) + w_2 \sum_{i \notin \mathcal{B}} g(\phi_i)$ with respect to $\mathcal{B}$ and conditioning on $\mathcal{H}$ is

$$\begin{aligned} \mathbb{E}\left[w_1 \sum_{i \in \mathcal{B}} g(\phi_i) + w_2 \sum_{i \notin \mathcal{B}} g(\phi_i) \middle| \mathcal{H} \right] &= \left(\frac{\mathcal{b}}{R} w_1 + \left(1 - \frac{\mathcal{b}}{R} \right) w_2 \right) \sum_{i=1}^{R} g(\phi_i) \\ &= \left(\frac{\mathcal{b}}{R} w_1 + \frac{R - \mathcal{b}}{R} w_2 \right) \sum_{i=1}^{R} g(\phi_i), \end{aligned}$$

since each i is selected with probability $\frac{\mathcal{b}}{R}$. □

We assume that ω^* is the unique optimal solution to (1). Under these standard assumptions and the previous results, our first proposition argues a supermartingale relationship for the sequence of the loss function errors $F(\omega^t) - F(\omega^*)$.

Proposition 1. *If there is neither feature sampling in step 5 nor sample sampling in step 7 and Assumptions 1-2 hold, and the number of iteration for the inner loop B is $B \geq C_1 d$ where C_1 is a positive constant, and the sequence of learning rates satisfies $\gamma_t \leq 1$ for all t, and are non-summable $\sum_{t=1}^{\infty} \gamma_t = \infty$ and square summable $\sum_{t=1}^{\infty} \gamma_t^2 < \infty$, and the sequence $(c^t)_{t=0}^{\infty}$ is selected so that $c^t \leq d$, then the loss function error sequence $F(\omega^t) - F(\omega^*)$ generated by SODDA satisfies*

$$\text{(A.1)} \qquad \mathbb{E}\left[F(\omega^{t+1}) - F(\omega^*)|\mathcal{F}^t\right] \leq \left[1 - 2\xi\gamma_{t+1}\left(\frac{B}{d} - C_1\gamma_{t+1}\right)\right][F(\omega^t) - F(\omega^*)],$$

where C_1 is a positive constant.

PROOF. We write $\triangledown F(\omega^t) = (\triangledown F(\omega^t)_{11}, \cdots, \triangledown F(\omega^t)_{1P}, \triangledown F(\omega^t)_{21}, \cdots, \triangledown F(\omega^t)_{QP})$ and $e^t = (e^t_{11}, \cdots, e^t_{1P}, e^t_{21}, \cdots, e^t_{QP})$. In order to simplify the notation, we also denote $\pi = (\pi_q)_{q=1}^{Q}$ and $j^{(i)}_{q,\pi_q(p)}$, the index drawn in step 10 of the algorithm for given $\pi_q(p)$, where everything computed is at iteration t.

Claim 1. *For any t we have*

$$\text{(A.2)} \qquad \mathbb{E}\left[\frac{1}{N}\sum_{j=1}^{N} \bar{\triangledown}_{\omega_{c^t}} f_j(x_j\omega^t)\middle| \mathcal{F}^t\right] = \frac{c^t}{d}\triangledown F(\omega^t),$$

$$\text{(A.3)} \qquad \mathbb{E}\left[\left\|\frac{1}{N}\sum_{j=1}^{N} \bar{\triangledown}_{\omega_{c^t}} f_j(x_j\omega^t)\right\|^2\middle| \mathcal{F}^t\right] = \frac{c^t}{d}\left\|\triangledown F(\omega^t)\right\|^2.$$

PROOF. For each j, we have

$$\mathbb{E}\left[\bar{\nabla}_{\omega_{\mathcal{C}^t}} f_j(x_j\omega^t)\middle|\mathcal{F}^t\right] = \frac{1}{\binom{d}{c^t}}\sum_{\mathcal{C}^t}\bar{\nabla}_{\omega_{\mathcal{C}^t}} f_j(x_j\omega^t)$$
$$= \frac{1}{\binom{d}{c^t}}\cdot\binom{d-1}{c^t-1}\nabla f_j(x_j\omega^t) = \frac{c^t}{d}\nabla f_j(x_j\omega^t).$$

This yields

$$\mathbb{E}\left[\frac{1}{N}\sum_{j=1}^{N}\bar{\nabla}_{\omega_{\mathcal{C}^t}} f_j(x_j\omega^t)\middle|\mathcal{F}^t\right] = \frac{1}{N}\sum_{j=1}^{N}\frac{c^t}{d}\nabla f_j(x_j\omega^t) = \frac{c^t}{Nd}\sum_{j=1}^{N}\nabla f_j(x_j\omega^t). \tag{A.4}$$

By substituting the definition of $\nabla F(\omega^t)$ into (A.4) claim (A.2) follows.

Let us proceed to find a relationship between the value of $\left\|\frac{1}{N}\sum_{j=1}^{N}\bar{\nabla}_{\omega_{\mathcal{C}^t}} f_j(x_j\omega^t)\right\|^2$ given $\mathcal{F}^t$ and $\|\nabla F(\omega^t)\|$. Similarly,

$$\mathbb{E}\left[\left\|\frac{1}{N}\sum_{j=1}^{N}\bar{\nabla}_{\omega_{\mathcal{C}^t}} f_j(x_j\omega^t)\right\|^2\middle|\mathcal{F}^t\right] = \frac{1}{\binom{d}{c^t}}\sum_{c^t}\left\|\frac{1}{N}\sum_{j=1}^{N}\nabla f_j(x_j\omega^t)\right\|^2$$
$$= \frac{1}{\binom{d}{c^t}}\cdot\binom{d-1}{c^t-1}\left\|\frac{1}{N}\sum_{j=1}^{N}\nabla f_j(x_j\omega^t)\right\|^2 = \frac{c^t}{d}\left\|\frac{1}{N}\sum_{j=1}^{N}\nabla f_j(x_j\omega^t)\right\|^2. \tag{A.5}$$

By substituting the definition of $\nabla F(\omega^t)$ into (A.5) claim (A.3) follows. □

For $i = 1, 2, \cdots, B$, the expected value of $\|v^{t,i}\|^2$ given all the preceding information is

$$\text{(A.6)}\quad \mathbb{E}\left[\left\|v^{t,1}\right\|^2 |\mathscr{F}^t, \mathscr{B}^t, \mathscr{C}^t, \mathscr{D}^t, \pi\right] = 0$$

$$\mathbb{E}\left[\left\|v^{t,i}\right\|^2 |\mathscr{F}^t, \mathscr{B}^t, \mathscr{C}^t, \mathscr{D}^t, \pi, j_{11}^{(1)}, j_{12}^{(1)}, \cdots, j_{QP}^{(1)}, j_{11}^{(2)}, \cdots, j_{QP}^{(2)}, \cdots, j_{11}^{(i-2)}, \cdots, j_{QP}^{(i-2)}\right]$$

$$= \sum_{j_{QP}^{(i-1)}=1}^{n} \cdots \sum_{j_{11}^{(i-1)}=1}^{n} \frac{1}{n^{QP}}$$

$$\text{(A.7)}\qquad \left\| \begin{pmatrix} \nabla_{\omega_{11}} f_{j_{11}^{(i-1)}}^{\pi_1^{-1}(1)} \left(x_{j_{11}^{(i-1)}}^{\pi_1^{-1}(1),1,1} \bar{\omega}_{11}^{t,i-1} \right) - \nabla_{\omega_{11}} f_{j_{11}^{(i-1)}}^{\pi_1^{-1}(1)} \left(x_{j_{11}^{(i-1)}}^{\pi_1^{-1}(1),1,1} \omega_{11}^{t} \right) \\ \vdots \\ \nabla_{\omega_{1P}} f_{j_{1P}^{(i-1)}}^{\pi_1^{-1}(P)} \left(x_{j_{1P}^{(i-1)}}^{\pi_1^{-1}(P),1,P} \bar{\omega}_{1P}^{t,i-1} \right) - \nabla_{\omega_{1P}} f_{j_{1P}^{(i-1)}}^{\pi_1^{-1}(P)} \left(x_{j_{1P}^{(i-1)}}^{\pi_1^{-1}(P),1,P} \omega_{1P}^{t} \right) \\ \vdots \\ \nabla_{\omega_{QP}} f_{j_{QP}^{(i-1)}}^{\pi_Q^{-1}(P)} \left(x_{j_{QP}^{(i-1)}}^{\pi_Q^{-1}(P),Q,P} \bar{\omega}_{QP}^{t,i-1} \right) - \nabla_{\omega_{QP}} f_{j_{QP}^{(i-1)}}^{\pi_Q^{-1}(P)} \left(x_{j_{QP}^{(i-1)}}^{\pi_Q^{-1}(P),Q,P} \omega_{QP}^{t} \right) \end{pmatrix} \right\|^2,$$

for $i = 2, 3, \cdots, B$.

We prove a bound of the value of $\|v^{t,i}\|^2$ given $\mathscr{F}^t$ by induction for $i \in \{1, 2, \cdots, B\}$.

Claim 2. *For $i = 1, 2, \cdots, B$, we have*

$$\text{(A.8)}\qquad \mathbb{E}\left[\left\|v^{t,i}\right\|^2 |\mathscr{F}^t\right] = \hat{\mathcal{O}}(\gamma_{t+1}^2 \left\|\nabla F(\omega^t)\right\|^2).$$

PROOF. The claim holds for $i = 1$ due to (A.40).

For $i = 1, 2, \cdots, k-1$, we assume that

$$\text{(A.9)}\qquad \mathbb{E}\left[\left\|v^{t,i}\right\|^2 |\mathscr{F}^t\right] = \hat{\mathcal{O}}(\gamma_{t+1}^2 \left\|\nabla F(\omega^t)\right\|^2).$$

Now consider $v^{t,k}$. Let us show that the expected value of $\left\|v^{t,k}\right\|^2$ is bounded. By using (A.7) we have

$$\begin{aligned}
&\mathbb{E}\left[\left\|v^{t,k}\right\|^2 |\mathcal{F}^t, \mathcal{B}^t, \mathcal{C}^t, \mathcal{D}^t, \pi, j_{11}^{(1)}, j_{12}^{(1)}, \cdots, j_{QP}^{(1)}, j_{11}^{(2)}, j_{12}^{(2)}, \cdots, j_{QP}^{(2)}, \cdots, j_{11}^{(k-2)}, \cdots, j_{QP}^{(k-2)}\right] \\
&\leq \sum_{j_{QP}^{(k-1)}=1}^{n} \cdots \sum_{j_{11}^{(k-1)}=1}^{n} \sum_{q=1}^{Q}\sum_{p=1}^{P} \frac{1}{n^{QP}} \left\| \nabla_{\omega_{qp}} f_{j_{qp}^{(k-1)}}^{\pi_q^{-1}(p)} \left(x_{j_{qp}^{(k-1)}}^{\pi_q^{-1}(p),q,p} \bar{\omega}_{qp}^{t,k-1} \right) \right. \\
&\qquad \left. - \nabla_{\omega_{qp}} f_{j_{qp}^{(k-1)}}^{\pi_q^{-1}(p)} \left(x_{j_{qp}^{(k-1)}}^{\pi_q^{-1}(p),q,p} \omega_{qp}^{t} \right) \right\|^2 \\
&= \sum_{q=1}^{Q}\sum_{p=1}^{P} \left(\frac{1}{n} \sum_{j_{q,\pi_q(p)}^{(k-1)}=1}^{n} \left\| \nabla_{\omega_{qp}} f_{j_{qp}^{(k-1)}}^{\pi_q^{-1}(p)} \left(x_{j_{qp}^{(k-1)}}^{\pi_q^{-1}(p),q,p} \bar{\omega}_{qp}^{t,k-1} \right) - \nabla_{\omega_{qp}} f_{j_{qp}^{(k-1)}}^{\pi_q^{-1}(p)} \left(x_{j_{qp}^{(k-1)}}^{\pi_q^{-1}(p),q,p} \omega_{qp}^{t} \right) \right\|^2 \right) \\
&\leq \sum_{q=1}^{Q}\sum_{p=1}^{P} \left(\frac{1}{n} \cdot nL^2 \left\| \bar{\omega}_{qp}^{t,k-1} - \bar{\omega}_{qp}^{t,0} \right\|^2 \right) = \sum_{q=1}^{Q}\sum_{p=1}^{P} \left(L^2 \left\| \bar{\omega}_{qp}^{t,k-1} - \bar{\omega}_{qp}^{t,0} \right\|^2 \right)
\end{aligned} \tag{A.10}$$

$$= \sum_{q=1}^{Q}\sum_{p=1}^{P} L^2 \left\| \gamma_{t+1} \left[(k-1)\mu_{qp}^{t} + v_{qp}^{t,1} + \cdots + v_{qp}^{t,k-1} \right] \right\|^2 .$$

Applying the definition of μ^t yields

$$\mathbb{E}\left[\left\|\mu^t\right\|^2 |\mathcal{F}^t\right] = \mathbb{E}\left[\left\| \frac{1}{N} \sum_{j=1}^{N} \bar{\nabla}_{\omega_{\mathcal{C}^t}} f_j(x_j \omega^t) \right\|^2 \middle| \mathcal{F}^t \right] = \frac{c^t}{d} \left\| \nabla F(\omega^t) \right\|^2 . \tag{A.11}$$

The second inequality holds due to (A.3) in Claim 1. By using the law of iterated expectation, (A.9), (A.10) and (A.11) we get

$$
\begin{aligned}
&\mathbb{E}\left[\left\|v^{t,k}\right\|^2 |\mathcal{F}^t\right] = \mathbb{E}\left[\mathbb{E}\left[\left\|v^{t,k}\right\|^2 |\mathcal{F}^t, , \mathcal{B}^t, \mathcal{C}^t, \mathcal{D}^t, \pi, j_{11}^{(1)} \cdots, j_{QP}^{(k-2)}\right] |\mathcal{F}^t\right] \\
&\leq \mathbb{E}\left[\sum_{q=1}^{Q}\sum_{p=1}^{P} L^2 \left\|\gamma_{t+1}\left[(k-1)\mu_{qp}^t + \sum_{i=1}^{k-1} v_{qp}^{t,i}\right]\right\|^2 \middle| \mathcal{F}^t\right] \\
&\leq kL^2\gamma_{t+1}^2 \sum_{q=1}^{Q}\sum_{p=1}^{P}\left(\mathbb{E}\left[\left\|(k-1)\mu_{qp}^t\right\|^2 \middle| \mathcal{F}^t\right] + \sum_{i=1}^{k-1}\mathbb{E}\left[\left\|v_{qp}^{t,i}\right\|^2 |\mathcal{F}^t\right]\right) \\
&\leq kL^2\gamma_{t+1}^2 QP\left((k-1)^2\,\mathbb{E}\left[\left\|\mu^t\right\|^2 \middle| \mathcal{F}^t\right] + \sum_{i=1}^{k-1}\mathbb{E}\left[\left\|v^{t,i}\right\|^2 |\mathcal{F}^t\right]\right) \\
&= kL^2\gamma_{t+1}^2 QP\left[(k-1)^2\left\|\nabla F(\omega^t)\right\|^2 + (k-1)\hat{\mathcal{O}}(\gamma_{t+1}^2\left\|\nabla F(\omega^t)\right\|^2)\right] \\
&= \hat{\mathcal{O}}(\gamma_{t+1}^2\left\|\nabla F(\omega^t)\right\|^2).
\end{aligned} \tag{A.12}
$$

This completes the proof of the claim. □

By using the conditional Jensen's inequality and (A.9) we get

$$
\mathbb{E}[\left\|v^{t,i}\right\| \mid \mathcal{F}^t] = \mathbb{E}[\sqrt{\|v^{t,i}\|^2} \mid \mathcal{F}^t] \leq \sqrt{\mathbb{E}[\|v^{t,i}\|^2 \mid \mathcal{F}^t]} = \hat{\mathcal{O}}(\gamma_{t+1}\left\|\nabla F(\omega^t)\right\|). \tag{A.13}
$$

By summing up all increments in iteration t, we obtain

$$
\omega^{t+1} = \omega^t - \gamma_{t+1}\left[B\mu^t + v^{t,1} + v^{t,2} + \cdots + v^{t,B}\right].
$$

Then, the expected value of the difference $\omega^{t+1} - \omega^t$ given $\mathcal{F}^t$ is

$$\begin{aligned}\mathbb{E}\left[\omega^{t+1} - \omega^t | \mathcal{F}^t\right] &= -\gamma_{t+1}\,\mathbb{E}\left[B\mu^t + v^{t,1} + \cdots + v^{t,B} | \mathcal{F}^t\right] \\ &= -\gamma_{t+1} B \frac{c^t}{d} \triangledown F(\omega^t) - \gamma_{t+1} \sum_{i=1}^{B} \mathbb{E}[v^{t,i} \mid \mathcal{F}^t],\end{aligned} \tag{A.14}$$

by using (A.2) in Claim 1. Moreover, the expected value of the squared norm $\|\omega^{t+1} - \omega^t\|^2$ given $\mathcal{F}^t$ is

(A.15)

$$\begin{aligned}&\mathbb{E}\left[\left\|\omega^{t+1} - \omega^t\right\|^2 | \mathcal{F}^t\right] = \mathbb{E}\left[\left\|\gamma_{t+1}\left[B\mu^t + v^{t,1} + v^{t,2} + \cdots + v^{t,B}\right]\right\|^2 | \mathcal{F}^t\right] \\ &\leq \gamma_{t+1}^2 (B+1)\left\{B^2\,\mathbb{E}\left[\left\|\mu^t\right\|^2 | \mathcal{F}^t\right] + \sum_{i=1}^{B} \mathbb{E}\left[\left\|v^{t,i}\right\|^2 | \mathcal{F}^t\right]\right\} \\ &= \hat{\mathcal{O}}(\gamma_{t+1}^2)\left\{B^2 \cdot \frac{c^t}{d}\left\|\triangledown F(\omega^t)\right\|^2 + B \cdot \hat{\mathcal{O}}(\gamma_{t+1}^2 \left\|\triangledown F(\omega^t)\right\|^2)\right\} = \hat{\mathcal{O}}(\gamma_{t+1}^2 \left\|\triangledown F(\omega^t)\right\|^2),\end{aligned}$$

due to (A.3), (A.8) and (A.11). From

$$-\triangledown F(\omega^t) \cdot v^{t,i} \leq \left\|\triangledown F(\omega^t)\right\| \left\|v^{t,i}\right\|,$$

for every $\triangledown F(\omega^t)$ and $v^{t,i}$, by using and (A.13) we obtain

$$\begin{aligned}-\gamma_{t+1}\triangledown F(\omega^t) \cdot \mathbb{E}\left[v^{t,i} | \mathcal{F}^t\right] &= \gamma_{t+1}\,\mathbb{E}\left[-\triangledown F(\omega^t) \cdot v^{t,i} | \mathcal{F}^t\right] \leq \gamma_{t+1}\,\mathbb{E}\left[\left\|\triangledown F(\omega^t)\right\| \cdot \left\|v^{t,i}\right\| | \mathcal{F}^t\right] \\ &= \gamma_{t+1}\left\|\triangledown F(\omega^t)\right\| \cdot \mathbb{E}\left[\left\|v^{t,i}\right\| | \mathcal{F}^t\right] = \hat{\mathcal{O}}(\gamma_{t+1}^2 \left\|\triangledown F(\omega^t)\right\|^2),\end{aligned} \tag{A.16}$$

since $\mathbb{E}\left[XY|\mathcal{H}\right] = X\,\mathbb{E}\left[Y|\mathcal{H}\right]$ if X is $\mathcal{H}-$measurable.

For convex F we have

$$F(\omega^{t+1}) \leq F(\omega^t) + \nabla F(\omega^t)^T \left(\omega^{t+1} - \omega^t\right) + \frac{L}{2} \left\|\omega^{t+1} - \omega^t\right\|^2,$$

which in turn yields

$$\begin{aligned}
&\mathbb{E}\left[F(\omega^{t+1})|\mathcal{F}^t\right] \leq F(\omega^t) + \nabla F(\omega^t)^T \mathbb{E}\left[\left(\omega^{t+1} - \omega^t\right)|\mathcal{F}^t\right] + \frac{L}{2}\mathbb{E}\left[\left\|\omega^{t+1} - \omega^t\right\|^2 |\mathcal{F}^t\right] \\
&= F(\omega^t) + \nabla F(\omega^t)^T \left\{-\gamma_{t+1} B \frac{c^t}{d} \nabla F(\omega^t) - \gamma_{t+1} \sum_{i=1}^{B} \mathbb{E}[v^{t,i} \mid \mathcal{F}^t]\right\} \\
&\qquad + \frac{L}{2}\mathbb{E}\left[\left\|\omega^{t+1} - \omega^t\right\|^2 |\mathcal{F}^t\right] \\
&= F(\omega^t) - \gamma_{t+1}\frac{c^t B}{d}\left\|\nabla F(\omega^t)\right\|^2 - \gamma_{t+1}\nabla F(\omega^t)^T \sum_{i=1}^{B} \mathbb{E}\left[v^{t,i}|\mathcal{F}^t\right] + \frac{L}{2}\mathbb{E}\left[\left\|\omega^{t+1} - \omega^t\right\|^2 |\mathcal{F}^t\right] \\
&\leq F(\omega^t) - \gamma_{t+1}\frac{c^t B}{d}\left\|\nabla F(\omega^t)\right\|^2 + \hat{\mathcal{O}}(\gamma_{t+1}^2 \left\|\nabla F(\omega^t)\right\|^2) \\
&\leq F(\omega^t) - \gamma_{t+1}\frac{c^t B}{d}\left\|\nabla F(\omega^t)\right\|^2 + C_1\gamma_{t+1}^2 \left\|\nabla F(\omega^t)\right\|^2,
\end{aligned}$$

where C_1 is a positive constant and we use (A.14), (A.15) and (A.16). Subtracting the optimal objective function $F(\omega^*)$ to the both sides of (A.17) and using the fact that $c^t \geq 1$ imply that

(A.17)
$$\mathbb{E}\left[F(\omega^{t+1}) - F(\omega^*)|\mathcal{F}^t\right] \leq F(\omega^t) - F(\omega^*) - \gamma_{t+1}\frac{B}{d}\left\|\nabla F(\omega^t)\right\|^2 + C_1\gamma_{t+1}^2 \left\|\nabla F(\omega^t)\right\|^2.$$

We proceed to find a lower bound of $\|\nabla F(\omega^t)\|^2$ in terms of $F(\omega^t) - F(\omega^*)$. Assumption 1 implies that, for any $y, z \in \mathbb{R}^m$

(A.18)
$$F(y) \geq F(z) + \nabla F(z)^T(y - z) + \frac{\xi}{2}\|y - z\|^2.$$

For fixed z, the right hand side of (A.18) is a quadratic function of y and it gets its minimum at $\hat{y} = z - \frac{1}{\xi}\nabla F(z)$. Therefore

$$F(y) \geq F(z) + \nabla F(z)^T(\hat{y} - z) + \frac{\xi}{2}\left\|\hat{y} - z\right\|^2 = F(z) - \frac{1}{2\xi}\left\|\nabla F(z)\right\|^2, \tag{A.19}$$

for any $y, z \in \mathbb{R}^d$. Setting $y = \omega^*$ and $z = \omega^t$ in (A.19) gives

$$\left\|\nabla F(\omega^t)\right\|^2 \geq 2\xi\left(F(\omega^t) - F(\omega^*)\right). \tag{A.20}$$

Substituting the lower bound in (A.20) by the norm of gradient square $\left\|\nabla F(\omega^t)\right\|^2$ in (A.17) yields the proposition in (A.1). □

Proposition 1 represents a relationship for the sequence of the loss function errors $F(\omega^t) - F(\omega^*)$. In the following theorem, by employing Taylor expansion, we show that if the sequence of learning rates satisfy the standard stochastic approximation diminishing learning rate rule (non-summable and squared summable), the sequence of loss function errors $F(\omega^t) - F(\omega^*)$ converges to 0.

Proof of Theorem 1.

PROOF. Since $B > \mathscr{d}C_1$ and $\gamma_t \leq 1$, we have $\frac{B}{\mathscr{d}} > C_1\gamma_{t+1}$, therefore, let us define $\phi = \frac{B}{\mathscr{d}} - C_1 > 0$ and $\frac{B}{\mathscr{d}} - C_1\gamma_{j+1} > \phi$. Consequently, we have

$$A_j = 1 - 2\xi\left(\frac{B}{\mathscr{d}} - C_1\gamma_{j+1}\right)\gamma_{j+1} < 1 - 2\xi\phi\gamma_{j+1},$$

which in turn yiels

$$\prod_{j=1}^{t} A_j < \prod_{j=1}^{t} (1 - 2\xi\phi\gamma_{j+1}) \,. \tag{A.21}$$

Applying (A.21) and Taylor expansion $\log(1-x) = -\sum_{n=1}^{\infty} \frac{x^n}{n}$, we consider the logarithm of the production

$$\begin{aligned}
&\log(\prod_{j=1}^{t} A_j) < \log\left(\prod_{j=1}^{t} (1 - 2\xi\phi\gamma_{j+1})\right) = \sum_{j=1}^{t} \log(1 - 2\xi\phi\gamma_{j+1}) \\
&= \sum_{j=1}^{t} - \sum_{n=1}^{\infty} \frac{(2\xi\phi\gamma_{j+1})^n}{n} = -\sum_{j=1}^{t} \left(2\xi\phi\gamma_{j+1} + \sum_{n=2}^{\infty} \frac{(2\xi\phi\gamma_{j+1})^n}{n}\right) \\
&= -\sum_{j=1}^{t} 2\xi\phi\gamma_{j+1} - \sum_{j=1}^{t}\sum_{n=2}^{\infty} \frac{(2\xi\phi\gamma_{j+1})^n}{n} < -\sum_{j=1}^{t} 2\xi\phi\gamma_{j+1} \\
&= -2\xi\phi \sum_{j=1}^{t} \gamma_{j+1} = -\infty,
\end{aligned}$$

which essentially implies $\lim_{t\to\infty} \prod_{j=1}^{t} A_j = 0$.The last equality uses the fact that the learning rate is non-summable. Thus, applying the law of iterated expectation to (A.1) implies (1.2) in Theorem 1. □

A.4. Counter Example without Assumptions 3 and 4

Proof of Theorem 2.

PROOF. We consider the setting where there are two samples $[A|b] = \left[\begin{array}{cc|c} a_{11} & a_{12} & b_1 \\ a_{21} & a_{22} & b_2 \end{array}\right]$, which are split into four partitions, and the parameter vector is specified as $\omega = [\omega_1, \omega_2]$. Then,

applying MSE and linear regression yields

$$F([\omega_1, \omega_2]) = \frac{1}{2} \|A\omega - b\|_2^2 . \tag{A.22}$$

Consequently, the gradient is

$$\bigtriangledown F = A^T A\omega - A^T b,$$

and the Hessian $H = A^T A$. Note that Assumption 1 holds. For Assumption 2, we can select $L = \max_{i,j} |H_{ij}|$. Notice that

$$\left\|A^T A\right\| = \|A\|^2 \geq \left(\frac{1}{\sqrt{2}} \|A\|_F\right)^2 = \frac{a_{11}^2 + a_{12}^2 + a_{21}^2 + a_{22}^2}{2}. \tag{A.23}$$

Let us consider the inner loop in steps 12-18. The approximate individual loss functions using each sample are

$$f_1(\omega_1) = \frac{1}{2}(\omega_1 a_{11} - b_1)^2, \quad f_1(\omega_2) = \frac{1}{2}(\omega_2 a_{12} - b_1)^2,$$
$$f_2(\omega_1) = \frac{1}{2}(\omega_1 a_{21} - b_2)^2, \quad f_2(\omega_2) = \frac{1}{2}(\omega_2 a_{22} - b_2)^2,$$

which in turn yields

$$\bigtriangledown_{\omega_1} f_1(\omega_1 a_{11}) = a_{11}^2 \omega_1 - a_{11} b_1, \quad \bigtriangledown_{\omega_2} f_1(\omega_2 a_{12}) = a_{12}^2 \omega_2 - a_{12} b_1,$$
$$\bigtriangledown_{\omega_1} f_2(\omega_1 a_{21}) = a_{21}^2 \omega_1 - a_{21} b_2, \quad \bigtriangledown_{\omega_2} f_2(\omega_2 a_{22}) = a_{22}^2 \omega_2 - a_{22} b_2.$$

The update formulas in the algorithm for the first sample and ω_1 read

$$\begin{aligned}
\bar{\omega}_1^{(i+1)} &= \bar{\omega}_1^{(i)} - \gamma_{t+1}\left(a_{11}^2\bar{\omega}_1^{(i)} - a_{11}^2\omega_1^t + \mu_1^t\right) \\
\bar{\omega}_1^{(i+1)} &= \left(1 - a_{11}^2\gamma_{t+1}\right)\bar{\omega}_1^{(i)} + \gamma_{t+1}\left(a_{11}^2\omega_1^t - \mu_1^t\right) \\
\bar{\omega}_1^{(i+1)} - \left(\omega_1^t - \frac{\mu_1^t}{a_{11}^2}\right) &= \left(1 - a_{11}^2\gamma_{t+1}\right)\left(\bar{\omega}_1^{(i)} - \left(\omega_1^t - \frac{\mu_1^t}{a_{11}^2}\right)\right).
\end{aligned}$$

Therefore,

$$\bar{\omega}_1^{(i+1)} = \left(1 - a_{11}^2\gamma_{t+1}\right)^{i+1}\left(\bar{\omega}_1^{(0)} - \left(\omega_1^t - \frac{\mu_1^t}{a_{11}^2}\right)\right) + \left(\omega_1^t - \frac{\mu_1^t}{a_{11}^2}\right).$$

Since $\bar{\omega}_1^{(0)} = \omega_1^t$ due to step 13, we obtain

$$\bar{\omega}_1^{(i+1)} = \left(1 - a_{11}^2\gamma_{t+1}\right)^{i+1}\frac{\mu_1^t}{a_{11}^2} + \left(\omega_1^t - \frac{\mu_1^t}{a_{11}^2}\right).$$

If $\gamma_t \leq \frac{1}{\min\left\{a_{11}^2, a_{12}^2, a_{21}^2, a_{22}^2\right\}}$, then $a_{11}^2\gamma_t < 1$, which in turn yields

$$\lim_{i\to\infty}\bar{\omega}_1^{(i+1)} = \omega_1^t - \frac{\mu_1^t}{a_{11}^2}.$$

Thus, if the number of iterations B for the inner loop is big enough, then, approximately, $\omega_1^{t+1} = \omega_1^t - \frac{\mu_1^t}{a_{11}^2}$. Similarly, when using the same data point to update ω_2, we obtain

$$\omega_2^{t+1} = \omega_2^t - \frac{\mu_2^t}{a_{12}^2}.$$

When using $([a_{21}, a_{22}], b_2)$ to update ω_1 and ω_2, we obtain

$$\omega_1^{t+1} = \omega_1^t - \frac{\mu_1^t}{a_{21}^2}, \quad \omega_2^{t+1} = \omega_2^t - \frac{\mu_2^t}{a_{22}^2}.$$

Therefore, the inner loop mimics gradient descent with constant learning rate. Then, the explicit update formula for ω^t is

$$\omega^{t+1} = \omega^t - \eta \bigtriangledown F(\omega^t) = (I - \eta A^T A)\omega^t + \eta A^T b, \tag{A.24}$$

where the second equality holds due to the definition of $\bigtriangledown F$, and $\eta = \begin{bmatrix} \frac{1}{a_{11}^2} & 0 \\ 0 & \frac{1}{a_{12}^2} \end{bmatrix}$ when using the first sample and $\eta = \begin{bmatrix} \frac{1}{a_{21}^2} & 0 \\ 0 & \frac{1}{a_{22}^2} \end{bmatrix}$ when using the second sample. Thus, if $|a_{11}| = |a_{12}| = \min\{|a_{11}|, |a_{12}|, |a_{21}|, |a_{22}|\}$ and $\max\{|a_{21}|, |a_{22}|\} > |a_{11}|$, then, when using $([a_{11}, a_{12}], b_1)$ to update ω_1, the corresponding $\left\|I - \eta A^T A\right\| > 1$ since $\left\|\eta A^T A\right\| \geq \frac{\left\|A^T A\right\|}{\|\eta^{-1}\|} > \left|\frac{a_{11}^2 + a_{12}^2 + a_{21}^2 + a_{22}^2}{2a_{11}^2}\right| = 2$, which in turn yields the divergence of the algorithm. Similarly, the same conclusion applies to $([a_{21}, a_{22}], b_2)$ when $|a_{21}| = |a_{22}| = \min\{|a_{11}|, |a_{12}|, |a_{21}|, |a_{22}|\}$ and $\max\{|a_{11}|, |a_{12}|\} > |a_{21}|$. Some possible values of A and b are in Table A.1.

a_{11}	a_{12}	b_1	a_{21}	a_{22}	b_2	optimal ω^*	ω^{100}
1	1	1	2	3	0	$[3, -1]$	$[3.651 \times 10^{55}, -6.811 \times 10^{56}]$
2	1	1	1	1	0	$[1, -1]$	$[54.606, -29.148]$
1	2	1	1	3	0	$[3, -1]$	$[-4.414 \times 10^{11}, -8.765 \times 10^{11}]$
1	2	1	2	3	1	$[-27, 17]$	$[4.973 \times 10^{29}, -3,455 \times 10^{30}]$
1	4	1	2	3	0	$[-\frac{3}{5}, \frac{2}{5}]$	$[-177.419, 2976.815]$

Table A.1. Counter Examples

□

A.5. Diminishing L.R. Convergence with Feature Sampling

We assume that ω^* is the unique optimal solution to (1), and also that $\|\omega^*\| \leq \frac{M_2}{2}$. By using Assumption 3, we conclude that the distance between any $\omega \in \Omega$ and ω^* is bounded, i.e.

$$\left\|\omega^t - w^*\right\| \leq M_2. \tag{A.25}$$

The second moment of ω^t is also bounded for all t, i.e.

$$\left\|\omega^t\right\|^2 \leq \frac{M_2^2}{4}, \tag{A.26}$$

for any t. Let us define $\mu^t = \frac{1}{\mathscr{d}^t} \sum_{j \in \mathscr{D}^t} \bar{\nabla}_{\omega_{\mathscr{C}^t}} f_j(x_j \omega^t) + e^t$, where e^t is defined as

$$e^t := \frac{1}{\mathscr{d}^t} \sum_{j \in \mathscr{D}^t} \bar{\nabla}_{\omega_{\mathscr{C}^t}} f_j(x_j^{\mathscr{B}^t} \omega_{\mathscr{B}^t}^t) - \frac{1}{\mathscr{d}^t} \sum_{j \in \mathscr{D}^t} \bar{\nabla}_{\omega_{\mathscr{C}^t}} f_j(x_j \omega^t). \tag{A.27}$$

Lemma 2. *If Assumptions 2-4 hold, then* $\|\nabla F(\omega^t)\|$ *and* $\sum_{j=1}^N \|\nabla f_j(x_j \omega^t)\|^2$ *for any* t *satisfy*

$$\left\|\nabla F(\omega^t)\right\| \leq M_2 L, \tag{A.28}$$

$$\sum_{j=1}^{N} \left\|\nabla f_j(x_j \omega^t)\right\|^2 \leq (N-1)G^2 + N M_2^2 L^2. \tag{A.29}$$

PROOF. Using the fact that ω^* is the optimal solution and Assumptions 2 and 3 we obtain

$$\left\|\nabla F(\omega^t)\right\| = \left\|\nabla F(\omega^t) - \nabla F(\omega^*)\right\| \leq L \left\|\omega^t - \omega^*\right\| \leq L M_2.$$

The last inequality holds due to (A.25). Assumption 4 implies that, for any ω^t we have

$$\frac{1}{N-1}\sum_{j=1}^{N}\left(\left\|\nabla f_j(x_j\omega^t)\right\|^2-\left\|\nabla F(\omega^t)\right\|^2\right)\leq G^2,$$

which in turn yields

$$\frac{1}{N-1}\sum_{j=1}^{N}\left\|\nabla f_j(x_j\omega^t)\right\|^2\leq G^2+\frac{N}{N-1}\left\|\nabla F(\omega^t)\right\|^2,$$

and

$$\sum_{j=1}^{N}\left\|\nabla f_j(x_jw^t)\right\|^2\leq (N-1)G^2+N\left\|\nabla F(\omega^t)\right\|^2. \tag{A.30}$$

Inserting (A.28) into (A.30) gives (A.29). □

Claim 3. *If Assumptions 2 and 3 hold and for some constant $\eta\geq 0$ and the learning rate γ_{t+1}, we have*

$$\mathcal{b}^t\in\left[\max\left\{c^t,\frac{d}{1+\frac{4d\eta\gamma_{t+1}^2}{c^tM_2^2L^2}}\right\},d\right] \tag{A.31}$$

for every t, then the expected value of $\left\|e^t\right\|^2$ conditioned on $\mathcal{F}^t$ generated by SODDA is bounded by $\eta\gamma_{t+1}^2$, that is

$$\mathbb{E}\left[\left\|e^t\right\|^2|\mathcal{F}^t\right]\leq\eta\gamma_{t+1}^2,$$

where η is a constant unrelated to B.

PROOF. By using (A.27) we obtain

$$\begin{aligned}
\mathbb{E}\left[\left\|e^t\right\|^2|\mathcal{F}^t\right] &= \mathbb{E}\left[\left\|\frac{1}{\mathscr{d}^t}\sum_{j\in\mathcal{D}^t}\bar{\nabla}_{\omega_{\mathcal{C}^t}}f_j(x_j^{\mathcal{B}^t}\omega_{\mathcal{B}^t}^t)-\frac{1}{\mathscr{d}^t}\sum_{j\in\mathcal{D}^t}\bar{\nabla}_{\omega_{\mathcal{C}^t}}f_j(x_j\omega^t)\right\|^2\middle|\mathcal{F}^t\right]\\
&= \frac{1}{(\mathscr{d}^t)^2}\mathbb{E}\left[\left\|\sum_{j\in\mathcal{D}^t}\left(\bar{\nabla}_{\omega_{\mathcal{C}^t}}f_j(x_j^{\mathcal{B}^t}\omega_{\mathcal{B}^t}^t)-\bar{\nabla}_{\omega_{\mathcal{C}^t}}f_j(x_j\omega^t)\right)\right\|^2\middle|\mathcal{F}^t\right]\\
&\leq \frac{1}{\mathscr{d}^t}\mathbb{E}\left[\sum_{j\in\mathcal{D}^t}\left\|\bar{\nabla}_{\omega_{\mathcal{C}^t}}f_j(x_j^{\mathcal{B}^t}\omega_{\mathcal{B}^t}^t)-\bar{\nabla}_{\omega_{\mathcal{C}^t}}f_j(x_j\omega^t)\right\|^2\middle|\mathcal{F}^t\right]\\
&= \frac{1}{\mathscr{d}^t}\mathbb{E}\left[\sum_{j\in\mathcal{D}^t}\mathbb{E}\left[\left\|\bar{\nabla}_{\omega_{\mathcal{C}^t}}f_j(x_j^{\mathcal{B}^t}\omega_{\mathcal{B}^t}^t)-\bar{\nabla}_{\omega_{\mathcal{C}^t}}f_j(x_j\omega^t)\right\|^2\middle|\mathcal{F}^t,\mathcal{B}^t,\mathcal{D}^t\right]\middle|\mathcal{F}^t\right]. \qquad \text{(A.32)}
\end{aligned}$$

Applying Lemma 1 with $w_1 = 1$, $w_2 = 0$,$\Phi = \left\{\left(\bar{\nabla}_{\omega_{\mathcal{B}^t}}f_j(x_j^{\mathcal{B}^t}\omega_{\mathcal{B}^t}^t)-\bar{\nabla}_{\omega_{\mathcal{B}^t}}f_j(x_j\omega^t)\right)_i\right\}_{i=1}^{\mathscr{b}^t}$, $g(z)=z^2$, $\mathcal{H}=\sigma(\mathcal{F}^t,\mathcal{B}^t,\mathcal{D}^t)$ and $\mathcal{B}=\mathcal{C}^t$ to (A.32) yields

$$\begin{aligned}
\mathbb{E}\left[\left\|e^t\right\|^2|\mathcal{F}^t\right] &\leq \frac{c^t}{\mathscr{b}^t\mathscr{d}^t}\mathbb{E}\left[\sum_{j\in\mathcal{D}^t}\left\|\bar{\nabla}_{\omega_{\mathcal{B}^t}}f_j(x_j^{\mathcal{B}^t}\omega_{\mathcal{B}^t}^t)-\bar{\nabla}_{\omega_{\mathcal{B}^t}}f_j(x_j\omega^t)\right\|^2\middle|\mathcal{F}^t\right]\\
&= \frac{c^t}{\mathscr{b}^t\mathscr{d}^t}\mathbb{E}\left[\sum_{j\in\mathcal{D}^t}\left\|\bar{\nabla}_{\omega_{\mathcal{B}^t}}f_j(x_j\omega_{(\mathcal{B}^t,0)}^t)-\bar{\nabla}_{\omega_{\mathcal{B}^t}}f_j(x_j\omega^t)\right\|^2\middle|\mathcal{F}^t\right]\\
&\leq \frac{L^2c^t}{\mathscr{b}^t\mathscr{d}^t}\mathbb{E}\left[\sum_{j\in\mathcal{D}^t}\left\|\omega_{(\mathcal{B}^t,0)}^t-\omega^t\right\|^2\middle|\mathcal{F}^t\right]\\
&= \frac{L^2c^t}{\mathscr{b}^t\mathscr{d}^t}\mathbb{E}\left[\mathbb{E}\left[\sum_{j\in\mathcal{D}^t}\left\|\omega_{[d]\setminus\mathcal{B}^t}^t\right\|^2\middle|\mathcal{F}^t,\mathcal{B}^t\right]\middle|\mathcal{F}^t\right]\\
&= \frac{L^2c^t}{\mathscr{b}^t\mathscr{d}^t}\mathbb{E}\left[\mathbb{E}\left[\mathscr{d}^t\left\|\omega_{[d]\setminus\mathcal{B}^t}^t\right\|^2\middle|\mathcal{F}^t,\mathcal{B}^t\right]\middle|\mathcal{F}^t\right]\\
&= \frac{L^2c^t}{\mathscr{b}^t}\mathbb{E}\left[\left\|\omega_{[d]\setminus\mathcal{B}^t}^t\right\|^2\middle|\mathcal{F}^t\right].
\end{aligned}$$

The second inequality uses Assumption 2 and we use $[d] = \{1, \cdots, d\}$. Now, let us use Lemma 1 again with $w_1 = 0$, $w_2 = 1$, $\Phi = \{(\omega^t)_i\}_{i=1}^d$, $g(z) = z^2$, $\mathcal{H} = \mathcal{F}^t$ and $\mathcal{B} = \mathcal{B}^t$ to get

$$\begin{aligned}\mathbb{E}\left[\left\|e^t\right\|^2 |\mathcal{F}^t\right] &\leq \frac{L^2 c^t}{\mathcal{b}^t} \frac{d - \mathcal{b}^t}{d} \left\|\omega^t\right\|^2 \\ &\leq \frac{L^2 c^t (d - \mathcal{b}^t)}{d\mathcal{b}^t} \frac{M_2^2}{4}.\end{aligned}$$

The last inequality uses (A.26). In order to bound the expected value of $\|e^t\|^2$ by $\eta\gamma_{t+1}^2$, we require

$$\frac{L^2 c^t (d - \mathcal{b}^t)}{d\mathcal{b}^t} \frac{M_2^2}{4} \leq \eta\gamma_{t+1}^2,$$

which is equivalent to

$$\mathcal{b}^t \geq \frac{d}{1 + \frac{4d\eta\gamma_{t+1}^2}{c^t M_2^2 L^2}}.$$

Meanwhile, in order to make $\nabla_{\omega_{c^t}} f_j(x_j^{\mathcal{B}^t} \omega_{\mathcal{B}^t}^t)$ well defined, we require $\mathcal{b}^t \geq c^t$. □

Next, similar to Proposition 1, we present the following proposition.

Proposition 2. *If Assumptions 1-4 hold, and the sequence of learning rates satisfies $\gamma_t \leq 1$ for all t, and the sequences $(\mathcal{b}^t, c^t, \mathcal{d}^t)_{t=0}^{\infty}$ are selected so that (A.31) for some constant $\eta \geq 0$, $c^t \leq d$ and $\mathcal{d}^t \leq N$, then the loss function error sequence $F(\omega^t) - F(\omega^*)$ generated by SODDA satisfies*

$$\mathbb{E}\left[F(\omega^{t+1}) - F(\omega^*)|\mathcal{F}^t\right] \leq (1 - \frac{2\xi B}{d}\gamma_{t+1})[F(\omega^t) - F(\omega^*)] + C_3\gamma_{t+1}^2, \tag{A.33}$$

where C_3 is a positive constant.

PROOF. We write $\nabla F(\omega^t) = (\nabla F(\omega^t)_{11}, \cdots, \nabla F(\omega^t)_{1P}, \nabla F(\omega^t)_{21}, \cdots, \nabla F(\omega^t)_{QP})$ and $e^t = (e^t_{11}, \cdots, e^t_{1P}, e^t_{21}, \cdots, e^t_{QP})$. In order to simplify the notation, we also denote $\pi = (\pi_q)_{q=1}^Q$ and $j^{(i)}_{q,\pi_q(p)}$, the index drawn in step 10 of the algorithm for given $\pi_q(p)$, where everything computed is at iteration t.

Claim 4. *For any t we have*

$$\mathbb{E}\left[\frac{1}{\mathscr{d}^t}\sum_{j\in\mathscr{D}^t}\bar{\nabla}_{\omega_{\mathcal{C}^t}} f_j(x_j\omega^t)\middle|\mathscr{F}^t\right] = \frac{c^t}{d}\nabla F(\omega^t), \tag{A.34}$$

$$\mathbb{E}\left[\left\|\frac{1}{\mathscr{d}^t}\sum_{j\in\mathscr{D}^t}\bar{\nabla}_{\omega_{\mathcal{C}^t}} f_j(x_j\omega^t)\right\|^2\middle|\mathscr{F}^t\right] \leq \frac{c^t}{Nd}\left[(N-1)G^2 + NM_2^2L^2\right]. \tag{A.35}$$

PROOF. Applying Lemma 1 with $w_1 = 1$, $w_2 = 0$, $\Phi = \left\{\bar{\nabla}_{\omega_{\mathcal{C}^t}} f_j(x_j\omega^t)\right\}_{j=1}^N$, $g(z) = z$, $\mathcal{H} = \sigma(\mathscr{F}^t, \mathcal{C}^t)$, $\mathcal{B} = \mathscr{D}^t$ and the law of iterated expectation imply

$$\mathbb{E}\left[\mathbb{E}\left[\frac{1}{\mathscr{d}^t}\sum_{j\in\mathscr{D}^t}\bar{\nabla}_{\omega_{\mathcal{C}^t}} f_j(x_j\omega^t)\middle|\mathscr{F}^t,\mathcal{C}^t\right]\middle|\mathscr{F}^t\right] = \frac{1}{\mathscr{d}^t}\cdot\frac{\mathscr{d}^t}{N}\sum_{j=1}^N\mathbb{E}\left[\bar{\nabla}_{\omega_{\mathcal{C}^t}} f_j(x_j\omega^t)\middle|\mathscr{F}^t\right].$$

For each j, we in turn have

$$\begin{aligned}\mathbb{E}\left[\bar{\nabla}_{\omega_{\mathcal{C}^t}} f_j(x_j\omega^t)\middle|\mathscr{F}^t\right] &= \frac{1}{\binom{d}{c^t}}\sum_{\mathcal{C}^t}\bar{\nabla}_{\omega_{\mathcal{C}^t}} f_j(x_j\omega^t)\\ &= \frac{1}{\binom{d}{c^t}}\cdot\binom{d-1}{c^t-1}\nabla f_j(x_j\omega^t) = \frac{c^t}{d}\nabla f_j(x_j\omega^t).\end{aligned}$$

This yields

$$\mathbb{E}\left[\frac{1}{\mathscr{d}^t}\sum_{j\in\mathscr{D}^t}\bar{\nabla}_{\omega_{\mathcal{C}^t}} f_j(x_j\omega^t)\middle|\mathscr{F}^t\right] = \frac{1}{N}\sum_{j=1}^N\frac{c^t}{d}\nabla f_j(x_j\omega^t) = \frac{c^t}{Nd}\sum_{j=1}^N\nabla f_j(x_j\omega^t). \tag{A.36}$$

By substituting the definition of $\nabla F(\omega^t)$ into (A.36) claim (A.34) follows.

Let us proceed to find an upper bound for the expected value of $\left\|\frac{1}{\ell^t}\sum_{j\in\mathcal{D}^t}\bar{\nabla}_{\omega_{C^t}}f_j(x_j\omega^t)\right\|^2$ given $\mathcal{F}^t$. Applying the law of iterated expectation and Lemma 1 with $w_1=1$, $w_2=0$, $\Phi=\left\{\bar{\nabla}_{\omega_{C^t}}f_j(x_j\omega^t)\right\}_{j=1}^N$, $g(z)=\|z\|^2$, $\mathcal{H}=\sigma(\mathcal{F}^t,c^t)$ and $\mathcal{B}=\mathcal{D}^t$ give

$$\begin{aligned}\mathbb{E}\left[\left\|\frac{1}{\ell^t}\sum_{j\in\mathcal{D}^t}\bar{\nabla}_{\omega_{C^t}}f_j(x_j\omega^t)\right\|^2\middle|\mathcal{F}^t\right]&\leq\frac{1}{\ell^t}\,\mathbb{E}\left[\sum_{j\in\mathcal{D}^t}\left\|\bar{\nabla}_{\omega_{C^t}}f_j(x_j\omega^t)\right\|^2\middle|\mathcal{F}^t\right]\\&=\frac{1}{\ell^t}\,\mathbb{E}\left[\mathbb{E}\left[\sum_{j\in\mathcal{D}^t}\left\|\bar{\nabla}_{\omega_{C^t}}f_j(x_j\omega^t)\right\|^2\middle|\mathcal{F}^t,\mathcal{C}^t\right]\middle|\mathcal{F}^t\right]=\frac{1}{\ell^t}\cdot\frac{\ell^t}{N}\,\mathbb{E}\left[\sum_{j=1}^N\left\|\bar{\nabla}_{\omega_{C^t}}f_j(x_j\omega^t)\right\|^2\middle|\mathcal{F}^t\right]\\&=\frac{1}{N}\sum_{j=1}^N\mathbb{E}\left[\left\|\bar{\nabla}_{\omega_{C^t}}f_j(x_j\omega^t)\right\|^2\middle|\mathcal{F}^t\right],\end{aligned}$$

which in turn yields

$$\begin{aligned}\mathbb{E}\left[\left\|\frac{1}{\ell^t}\sum_{j\in\mathcal{D}^t}\bar{\nabla}_{\omega_{C^t}}f_j(x_j\omega^t)\right\|^2\middle|\mathcal{F}^t\right]&\leq\frac{1}{N}\sum_{j=1}^N\frac{c^t}{d}\,\mathbb{E}\left[\left\|\nabla f_j(x_j\omega^t)\right\|^2\middle|\mathcal{F}^t\right]\\&=\frac{c^t}{Nd}\sum_{j=1}^N\left\|\nabla f_j(x_j\omega^t)\right\|^2,\end{aligned}\tag{A.37}$$

where we apply Lemma 1 for each j again with $w_1=1$, $w_2=0$, $\Phi=\left\{(\nabla f_j(x_j\omega^t))_i\right\}_{i=1}^d$, $g(z)=z^2$, $\mathcal{H}=\mathcal{F}^t$ and $\mathcal{B}=C^t$. By inserting (A.29) from Lemma 2 into (A.37) the claim in (A.35) follows. □

From Claim 3 we conclude

$$\mathbb{E}\left[\left\|e^t\right\|^2|\mathcal{F}^t\right]=\mathcal{O}(\gamma_{t+1}^2).\tag{A.38}$$

By using the conditional Jensen's inequality and (A.38) we get

$$\mathbb{E}\left[\left\|e^t\right\|\left|\mathcal{F}^t\right.\right] = \mathbb{E}\left[\sqrt{\left\|e^t\right\|^2}\left|\mathcal{F}^t\right.\right] \leq \sqrt{\mathbb{E}\left[\left\|e^t\right\|^2\left|\mathcal{F}^t\right.\right]} \leq \sqrt{\eta}\gamma_{t+1} = \mathcal{O}(\gamma_{t+1}). \tag{A.39}$$

For $i = 1, 2, \cdots, B$, the expected value of $\left\|v^{t,i}\right\|^2$ given all the preceding information is

(A.40)

$$\begin{aligned}
&\mathbb{E}\left[\left\|v^{t,1}\right\|^2\left|\mathcal{F}^t, \mathcal{B}^t, \mathcal{C}^t, \mathcal{D}^t, \pi\right.\right] = 0 \\
&\mathbb{E}\left[\left\|v^{t,i}\right\|^2\left|\mathcal{F}^t, \mathcal{B}^t, \mathcal{C}^t, \mathcal{D}^t, \pi, j_{11}^{(1)}, j_{12}^{(1)}, \cdots, j_{QP}^{(1)}, j_{11}^{(2)}, j_{12}^{(2)}, \cdots, j_{QP}^{(2)}, \cdots, j_{11}^{(i-2)}, \cdots, j_{QP}^{(i-2)}\right.\right] \\
&\quad = \sum_{j_{QP}^{(i-1)}=1}^{n} \cdots \sum_{j_{11}^{(i-1)}=1}^{n} \frac{1}{n^{QP}}
\end{aligned}$$

(A.41)

$$\left\| \begin{pmatrix} \nabla_{\omega_{11}} f_{j_{11}^{(i-1)}}^{\pi_1^{-1}(1)}\left(x_{j_{11}^{(i-1)}}^{\pi_1^{-1}(1),1,1}\bar{\omega}_{11}^{t,i-1}\right) - \nabla_{\omega_{11}} f_{j_{11}^{(i-1)}}^{\pi_1^{-1}(1)}\left(x_{j_{11}^{(i-1)}}^{\pi_1^{-1}(1),1,1}\omega_{11}^{t}\right) \\ \vdots \\ \nabla_{\omega_{1P}} f_{j_{1P}^{(i-1)}}^{\pi_1^{-1}(P)}\left(x_{j_{1P}^{(i-1)}}^{\pi_1^{-1}(P),1,P}\bar{\omega}_{1P}^{t,i-1}\right) - \nabla_{\omega_{1P}} f_{j_{1P}^{(i-1)}}^{\pi_1^{-1}(P)}\left(x_{j_{1P}^{(i-1)}}^{\pi_1^{-1}(P),1,P}\omega_{1P}^{t}\right) \\ \vdots \\ \nabla_{\omega_{QP}} f_{j_{QP}^{(i-1)}}^{\pi_Q^{-1}(P)}\left(x_{j_{QP}^{(i-1)}}^{\pi_Q^{-1}(P),Q,P}\bar{\omega}_{QP}^{t,i-1}\right) - \nabla_{\omega_{QP}} f_{j_{QP}^{(i-1)}}^{\pi_Q^{-1}(P)}\left(x_{j_{QP}^{(i-1)}}^{\pi_Q^{-1}(P),Q,P}\omega_{QP}^{t}\right) \end{pmatrix} \right\|^2,$$

for $i = 2, 3, \cdots, B$.

We prove a bound of the value of $\left\|v^{t,i}\right\|^2$ given $\mathcal{F}^t$ by induction for $i \in \{1, 2, \cdots, B\}$.

Claim 5. *For $i = 1, 2, \cdots, B$, we have*

$$\mathbb{E}\left[\left\|v^{t,i}\right\|^2 |\mathcal{F}^t\right] = \mathcal{O}(\gamma_{t+1}^2). \tag{A.42}$$

PROOF. The claim holds for $i = 1$ due to (A.40).

For $i = 1, 2, \cdots, k-1$, we assume that

$$\mathbb{E}\left[\left\|v^{t,i}\right\|^2 |\mathcal{F}^t\right] = \mathcal{O}(\gamma_{t+1}^2). \tag{A.43}$$

Now consider $v^{t,k}$. Let us show that the expected value of $\left\|v^{t,k}\right\|^2$ is bounded. By using (A.41) we have

$$\begin{aligned}
&\mathbb{E}\left[\left\|v^{t,k}\right\|^2 |\mathcal{F}^t, \mathcal{B}^t, \mathcal{C}^t, \mathcal{D}^t, \pi, j_{11}^{(1)}, j_{12}^{(1)}, \cdots, j_{QP}^{(1)}, j_{11}^{(2)}, j_{12}^{(2)}, \cdots, j_{QP}^{(2)}, \cdots, j_{11}^{(k-2)}, \cdots, j_{QP}^{(k-2)}\right] \\
&\leq \sum_{j_{QP}^{(k-1)}=1}^{n} \cdots \sum_{j_{11}^{(k-1)}=1}^{n} \sum_{q=1}^{Q}\sum_{p=1}^{P} \frac{1}{n^{QP}} \left\| \nabla_{\omega_{qp}} f_{j_{qp}^{(k-1)}}^{\pi_q^{-1}(p)} \left(x_{j_{qp}^{(k-1)}}^{\pi_q^{-1}(p),q,p} \bar{\omega}_{qp}^{t,k-1} \right) \right. \\
&\qquad \left. - \nabla_{\omega_{qp}} f_{j_{qp}^{(k-1)}}^{\pi_q^{-1}(p)} \left(x_{j_{qp}^{(k-1)}}^{\pi_q^{-1}(p),q,p} \omega_{qp}^{t} \right) \right\|^2 \\
&= \sum_{q=1}^{Q}\sum_{p=1}^{P} \left(\frac{1}{n} \sum_{j_{q,\pi_q(p)}^{(k-1)}=1}^{n} \left\| \nabla_{\omega_{qp}} f_{j_{qp}^{(k-1)}}^{\pi_q^{-1}(p)} \left(x_{j_{qp}^{(k-1)}}^{\pi_q^{-1}(p),q,p} \bar{\omega}_{qp}^{t,k-1} \right) - \nabla_{\omega_{qp}} f_{j_{qp}^{(k-1)}}^{\pi_q^{-1}(p)} \left(x_{j_{qp}^{(k-1)}}^{\pi_q^{-1}(p),q,p} \omega_{qp}^{t} \right) \right\|^2 \right) \\
&\leq \sum_{q=1}^{Q}\sum_{p=1}^{P} \left(\frac{1}{n} \cdot nL^2 \left\| \bar{\omega}_{qp}^{t,k-1} - \bar{\omega}_{qp}^{t,0} \right\|^2 \right) = \sum_{q=1}^{Q}\sum_{p=1}^{P} \left(L^2 \left\| \bar{\omega}_{qp}^{t,k-1} - \bar{\omega}_{qp}^{t,0} \right\|^2 \right) \\
&= \sum_{q=1}^{Q}\sum_{p=1}^{P} L^2 \left\| \gamma_{t+1} \left[(k-1)\mu_{qp}^{t} + v_{qp}^{t,1} + \cdots + v_{qp}^{t,k-1} \right] \right\|^2 .
\end{aligned} \tag{A.44}$$

Applying the definition of μ^t yields

$$\mathbb{E}\left[\left\|\mu^t\right\|^2 |\mathcal{F}^t\right] \leq 2\left\{\mathbb{E}\left[\left\|e^t\right\|^2 \middle| \mathcal{F}^t\right] + \mathbb{E}\left[\left\|\frac{1}{\mathscr{d}^t}\sum_{j\in\mathcal{D}^t}\bar{\nabla}_{\omega_{\mathcal{C}^t}} f_j(x_j\omega^t)\right\|^2 \middle| \mathcal{F}^t\right]\right\}$$

$$= \mathcal{O}(\gamma_{t+1}^2) + \mathcal{O}(1). \tag{A.45}$$

The second equality holds true due to (A.35) in Claim 4 and (A.38). By using the law of iterated expectation, (A.43), (A.44) and (A.45) we get

$$\begin{aligned}
&\mathbb{E}\left[\left\|v^{t,k}\right\|^2 |\mathcal{F}^t\right] = \mathbb{E}\left[\mathbb{E}\left[\left\|v^{t,k}\right\|^2 |\mathcal{F}^t, , \mathcal{B}^t, \mathcal{C}^t, \mathcal{D}^t, \pi, j_{11}^{(1)}\cdots, j_{QP}^{(k-2)}\right] |\mathcal{F}^t\right] \\
&\leq \mathbb{E}\left[\sum_{q=1}^{Q}\sum_{p=1}^{P} L^2 \left\|\gamma_{t+1}\left[(k-1)\mu_{qp}^t + \sum_{i=1}^{k-1} v_{qp}^{t,i}\right]\right\|^2 \middle| \mathcal{F}^t\right] \\
&\leq kL^2\gamma_{t+1}^2 \sum_{q=1}^{Q}\sum_{p=1}^{P}\left(\mathbb{E}\left[\left\|(k-1)\mu_{qp}^t\right\|^2 \middle| \mathcal{F}^t\right] + \sum_{i=1}^{k-1}\mathbb{E}\left[\left\|v_{qp}^{t,i}\right\|^2 |\mathcal{F}^t\right]\right) \\
&\leq kL^2\gamma_{t+1}^2 QP\left((k-1)^2\,\mathbb{E}\left[\left\|\mu^t\right\|^2 \middle| \mathcal{F}^t\right] + \sum_{i=1}^{k-1}\mathbb{E}\left[\left\|v^{t,i}\right\|^2 |\mathcal{F}^t\right]\right) \\
&= kL^2\gamma_{t+1}^2 QP\left[2(k-1)^2\left(\mathcal{O}(\gamma_{t+1}^2) + \mathcal{O}(1)\right) + \mathcal{O}(\gamma_{t+1}^2)\right] = \mathcal{O}(\gamma_{t+1}^2).
\end{aligned} \tag{A.46}$$

This completes the proof of the claim. □

By using the conditional Jensen's inequality and (A.42) we get

$$\mathbb{E}[\left\|v^{t,i}\right\| \mid \mathcal{F}^t] = \mathbb{E}[\sqrt{\|v^{t,i}\|^2} \mid \mathcal{F}^t] \leq \sqrt{\mathbb{E}[\|v^{t,i}\|^2 \mid \mathcal{F}^t]} = \mathcal{O}(\gamma_{t+1}). \tag{A.47}$$

By summing up all increments in iteration t, we obtain

$$\omega^{t+1} = \omega^t - \gamma_{t+1}\left[B\mu^t + v^{t,1} + v^{t,2} + \cdots + v^{t,B}\right].$$

Then, the expected value of the difference $\omega^{t+1} - \omega^t$ given $\mathcal{F}^t$ is

$$\begin{aligned}\mathbb{E}\left[\omega^{t+1} - \omega^t | \mathcal{F}^t\right] &= -\gamma_{t+1}\,\mathbb{E}\left[B\mu^t + v^{t,1} + \cdots + v^{t,B} | \mathcal{F}^t\right] \\ &= -\gamma_{t+1} B\left(\mathbb{E}[e^t \mid \mathcal{F}^t] + \frac{c^t}{d}\nabla F(\omega^t)\right) - \gamma_{t+1}\sum_{i=1}^{B}\mathbb{E}[v^{t,i} \mid \mathcal{F}^t],\end{aligned} \tag{A.48}$$

by using (A.34) in Claim 4. Moreover, the expected value of the squared norm $\|\omega^{t+1} - \omega^t\|^2$ given $\mathcal{F}^t$ is

$$\begin{aligned}\mathbb{E}\left[\left\|\omega^{t+1} - \omega^t\right\|^2 | \mathcal{F}^t\right] &= \mathbb{E}\left[\left\|\gamma_{t+1}\left[B\mu^t + v^{t,1} + v^{t,2} + \cdots + v^{t,B}\right]\right\|^2 | \mathcal{F}^t\right] \\ &\leq \gamma_{t+1}^2(B+1)\left\{B^2\,\mathbb{E}\left[\left\|\mu^t\right\|^2 | \mathcal{F}^t\right] + \sum_{i=1}^{B}\mathbb{E}\left[\left\|v^{t,i}\right\|^2 | \mathcal{F}^t\right]\right\} \\ &= \mathcal{O}(\gamma_{t+1}^2)\left\{\mathcal{O}(\gamma_{t+1}^2) + \mathcal{O}(1) + \mathcal{O}(\gamma_{t+1}^2)\right\} = \mathcal{O}(\gamma_{t+1}^2),\end{aligned} \tag{A.49}$$

due to Claim 3, (A.35), (A.42) and (A.45). From

$$-\nabla F(\omega^t)\cdot v^{t,i} \leq \left\|\nabla F(\omega^t)\right\|\left\|v^{t,i}\right\|,$$

for every $\nabla F(\omega^t)$ and $v^{t,i}$, by using (A.28) and (A.47) we obtain

$$\begin{aligned}-\gamma_{t+1}\nabla F(\omega^t)\cdot\mathbb{E}\left[v^{t,i}|\mathcal{F}^t\right] &= \gamma_{t+1}\,\mathbb{E}\left[-\nabla F(\omega^t)\cdot v^{t,i}|\mathcal{F}^t\right] \leq \gamma_{t+1}\,\mathbb{E}\left[\left\|\nabla F(\omega^t)\right\|\cdot\left\|v^{t,i}\right\| | \mathcal{F}^t\right] \\ &= \gamma_{t+1}\left\|\nabla F(\omega^t)\right\|\cdot\mathbb{E}\left[\left\|v^{t,i}\right\| | \mathcal{F}^t\right] = \mathcal{O}(\gamma_{t+1}^2),\end{aligned} \tag{A.50}$$

since $\mathbb{E}\left[XY|\mathcal{H}\right] = X\,\mathbb{E}\left[Y|\mathcal{H}\right]$ if X is $\mathcal{H}-$measurable. Similarly, applying (A.28) and (A.39) yields

$$\begin{aligned} -\gamma_{t+1}B\triangledown F(\omega^t)^T\,\mathbb{E}\left[e^t|\mathcal{F}^t\right] &\leq \gamma_{t+1}B\left\|\triangledown F(\omega^t)\right\|\mathbb{E}\left[\left\|e^t\right\||\mathcal{F}^t\right] \\ &= \mathcal{O}(\gamma_{t+1})\cdot\mathcal{O}(\gamma_{t+1}) = \mathcal{O}(\gamma_{t+1}^2). \end{aligned} \tag{A.51}$$

For convex F we have

$$F(\omega^{t+1}) \leq F(\omega^t) + \triangledown F(\omega^t)^T\left(\omega^{t+1}-\omega^t\right) + \frac{L}{2}\left\|\omega^{t+1}-\omega^t\right\|^2,$$

which in turn yields

(A.52)

$$\begin{aligned} &\mathbb{E}\left[F(\omega^{t+1})|\mathcal{F}^t\right] \leq F(\omega^t) + \triangledown F(\omega^t)^T\,\mathbb{E}\left[\left(\omega^{t+1}-\omega^t\right)|\mathcal{F}^t\right] + \frac{L}{2}\,\mathbb{E}\left[\left\|\omega^{t+1}-\omega^t\right\|^2|\mathcal{F}^t\right] \\ &= F(\omega^t) + \triangledown F(\omega^t)^T\left\{-\gamma_{t+1}B\left(\mathbb{E}[e^t\mid\mathcal{F}^t] + \frac{c^t}{d}\triangledown F(\omega^t)\right) - \gamma_{t+1}\sum_{i=1}^{B}\mathbb{E}[v^{t,i}\mid\mathcal{F}^t]\right\} \\ &\qquad + \frac{L}{2}\,\mathbb{E}\left[\left\|\omega^{t+1}-\omega^t\right\|^2|\mathcal{F}^t\right] \\ &= F(\omega^t) - \gamma_{t+1}\frac{c^tB}{d}\left\|\triangledown F(\omega^t)\right\|^2 - \gamma_{t+1}B\triangledown F(\omega^t)^T\,\mathbb{E}\left[e^t|\mathcal{F}^t\right] - \gamma_{t+1}\triangledown F(\omega^t)^T\sum_{i=1}^{B}\mathbb{E}\left[v^{t,i}|\mathcal{F}^t\right] \\ &\qquad + \frac{L}{2}\,\mathbb{E}\left[\left\|\omega^{t+1}-\omega^t\right\|^2|\mathcal{F}^t\right] \\ &\leq F(\omega^t) - \gamma_{t+1}\frac{c^tB}{d}\left\|\triangledown F(\omega^t)\right\|^2 + \mathcal{O}(\gamma_{t+1}^2) \leq F(\omega^t) - \gamma_{t+1}\frac{c^tB}{d}\left\|\triangledown F(\omega^t)\right\|^2 + C_3\gamma_{t+1}^2, \end{aligned}$$

where C_3 is a positive constant and we use (A.48), (A.49), (A.50) and (A.51). Subtracting the optimal objective function $F(\omega^*)$ to the both sides of (A.52) and using the fact that $c^t \geq 1$ imply

that

$$\mathbb{E}\left[F(\omega^{t+1}) - F(\omega^*)|\mathcal{F}^t\right] \leq F(\omega^t) - F(\omega^*) - \gamma_{t+1}\frac{B}{d}\left\|\nabla F(\omega^t)\right\|^2 + C_3\gamma_{t+1}^2. \tag{A.53}$$

We proceed to find a lower bound of $\|\nabla F(\omega^t)\|^2$ in terms of $F(\omega^t) - F(\omega^*)$. Assumption 1 implies that, for any $y, z \in \mathbb{R}^m$

$$F(y) \geq F(z) + \nabla F(z)^T(y - z) + \frac{\xi}{2}\|y - z\|^2. \tag{A.54}$$

For fixed z, the right hand side of (A.54) is a quadratic function of y and it gets its minimum at $\hat{y} = z - \frac{1}{\xi}\nabla F(z)$. Therefore

$$F(y) \geq F(z) + \nabla F(z)^T(\hat{y} - z) + \frac{\xi}{2}\|\hat{y} - z\|^2 = F(z) - \frac{1}{2\xi}\|\nabla F(z)\|^2, \tag{A.55}$$

for any $y, z \in \mathbb{R}^d$. Setting $y = \omega^*$ and $z = \omega^t$ in (A.55) gives

$$\left\|\nabla F(\omega^t)\right\|^2 \geq 2\xi\left(F(\omega^t) - F(\omega^*)\right). \tag{A.56}$$

Substituting the lower bound in (A.56) by the norm of gradient square $\|\nabla F(\omega^t)\|^2$ in (A.53) yields the proposition in (A.33). □

Proposition 2 represents a supermartingale relationship for the sequence of the loss function errors $F(\omega^t) - F(\omega^*)$. In the following theorem, by employing the supermartingale convergence argument, we show that if the sequence of learning rates satisfy the standard stochastic approximation diminishing learning rate rule (non-summable and squared summable), the sequence of loss function errors $F(\omega^t) - F(\omega^*)$ converges to 0 almost surely. Combining with

strong convexity of $F(\omega)$ in Assumption 1, this result implies that $\|\omega^t - \omega^*\|$ converges to 0 almost surely.

Proof of Theorem 3.

PROOF. We use the relationship in (A.33) to build a supermartingale sequence. First, let us define

$$\alpha^t := F(\omega^t) - F(\omega^*) + \sum_{u=t}^{\infty} C_1 \gamma_{u+1}^2, \tag{A.57}$$

$$\beta^t := \frac{2\xi B}{d} \gamma_{t+1} \left(F(\omega^t) - F(\omega^*) \right). \tag{A.58}$$

Note that α^t is well-defined since $\sum_{u=t}^{\infty} \gamma_{u+1}^2 < \sum_{u=1}^{\infty} \gamma_u^2 < \infty$. The definition of α^t and β^t in (A.57) and (A.58), and the inequality in (A.33) imply the expected value of α^{t+1} given $\mathcal{F}^t$ is

$$\mathbb{E}\left[\alpha^{t+1} | \mathcal{F}^t\right] \leq \alpha^t - \beta^t. \tag{A.59}$$

Since α^t and β^t are nonnegative and due to (A.59), they satisfy the conditions of the supermartingale convergence theorem. Thus, we conclude that

$$\text{(i)} \quad \alpha^t \text{ converges to a limit a.s., and} \tag{A.60}$$

$$\text{(ii)} \quad \sum_{t=1}^{\infty} \beta^t < \infty. \quad \text{a.s.} \tag{A.61}$$

Property (A.61) yields

$$\sum_{t=0}^{\infty} \frac{2c^t \xi B}{d} \gamma_{t+1} \left(F(\omega^t) - F(\omega^*) \right) < \infty. \quad \text{a.s.}$$

Since $\sum_{t=0}^{\infty} \gamma_{t+1} = \infty$, there exists a subsequence of $F(\omega^t) - F(\omega^*)$ which converges to 0, i.e.

$$\liminf_{t\to\infty} F(\omega^t) - F(\omega^*) = 0. \quad \text{a.s.} \tag{A.62}$$

Since $\sum_{u=t}^{\infty} C_1 \gamma_{u+1}^2$ is deterministic and due to (A.60), $F(\omega^t) - F(\omega^*)$ converges to a limit almost surely. In association with (A.62) we conclude

$$\lim_{t\to\infty} F(\omega^t) - F(\omega^*) = 0. \quad \text{a.s.} \tag{A.63}$$

We proceed to show the almost convergence of $\|\omega^t - \omega^*\|^2$. Using (A.54) again and setting $y = \omega^t$ and $z = \omega^*$ implies

$$F(\omega^t) \geq F(\omega^*) + \nabla F(\omega^*)^T(\omega^t - \omega^*) + \frac{\xi}{2}\left\|\omega^t - \omega^*\right\|^2. \tag{A.64}$$

Since the gradient of the optimal solution is 0, i.e.$\nabla F(\omega^*) = 0$, (A.64) can be rearranged as

$$F(\omega^t) - F(\omega^*) \geq \frac{\xi}{2}\left\|\omega^t - \omega^*\right\|^2.$$

Observing that the upper bound of $\|\omega^t - \omega^*\|^2$ converges to 0 almost surely by (A.63), we conclude that the sequence $\|\omega^t - \omega^*\|^2$ converges to zero almost surely. Hence, the claim in (1.3) is valid. □

Proof of Theorem 4.

PROOF. Replacing γ_{t+1} by $\frac{1}{t+1}$ and computing the expected value of (A.33) given $\mathcal{F}^0$ by using the law of iterated expectation we obtain

$$\mathbb{E}[F(\omega^{t+1}) - F(\omega^*)] \leq \left(1 - \frac{2\xi B}{(t+1)d}\right) \mathbb{E}[F(\omega^t) - F(\omega^*)] + \frac{C_1}{(t+1)^2}. \tag{A.65}$$

Let us define

$$\begin{aligned} a_t &:= \mathbb{E}[F(\omega^{t+1}) - F(\omega^*)] \\ \lambda &:= \frac{2\xi B}{d} \\ \beta &:= C_1. \end{aligned}$$

Note that β is positive. Based on the relationship in (A.65), we obtain

$$a_{t+1} \leq \left(1 - \frac{\lambda}{t+1}\right) a_t + \frac{\beta}{(t+1)^2}. \tag{A.66}$$

for all times $t \geq 0$. Now, we proceed to show

$$a_t \leq \frac{Q}{t+1}, \tag{A.67}$$

where $Q = \max\left\{a_0, 2a_1, \cdots, ([\lambda]+1)a_{[\lambda]}, ([\lambda]+2)\, a_{[\lambda]+1}, \frac{\beta}{\lambda - 1}\right\}$. The definition of Q implies that the relationship in (A.67) holds for $t = 1, 2, \cdots, [\lambda]$. The remaining cases are shown by induction.

When $t = [\lambda] + 1$, the definition of Q implies

$$a_{[\lambda]+1} \leq \frac{Q}{[\lambda]+2}.$$

When $t = k - 1$, we assume that the relationship in (A.67) holds. Considering the case when $t = k$ and using (A.66) implies

$$a_{k+1} \leq \left(1 - \frac{\lambda}{k+1}\right) a_k + \frac{\beta}{(k+1)^2} \leq \left(1 - \frac{\lambda}{k+1}\right) \frac{Q}{k+1} + \frac{\beta}{(k+1)^2}.$$

In order to satisfy (A.67), we require

$$\left(1 - \frac{\lambda}{k+1}\right) \frac{Q}{k+1} + \frac{\beta}{(k+1)^2} \leq \frac{Q}{k+2}.$$

Elementary algebraic manipulation shows that this is equivalent to

$$\beta(k+2) \leq Q\left[\lambda(k+2) - (k+1)\right]$$

and in turn

$$\frac{\beta(k+2)}{\lambda(k+2) - (k+1)} = \frac{\beta}{\lambda - \frac{k+1}{k+2}} \leq Q,$$

where we require $\lambda \geq 1$. The definition of Q, i.e. $Q \geq \frac{\beta}{\lambda - 1}$ and the relationship that $\lambda - \frac{k+1}{k+2} > \lambda - 1$ imply that

$$\frac{\beta}{\lambda - \frac{k+1}{k+2}} < \frac{\beta}{\lambda - 1} \leq Q,$$

and thus (A.67) holds for $t = k$. Thus, if $B \geq \frac{d}{2\xi}$, for any time $t \geq 0$, the result in (1.4) holds where the constant Q is defined based on (1.5). $\square$

Corollary 1 *If Assumptions 1-2 hold and the sequence of learning rates are non-summable* $\sum_{t=1}^{\infty}\gamma_t = \infty$ *and square summable* $\sum_{t=1}^{\infty}\gamma_t^2 < \infty$*, then the sequence of parameters* ω^t *generated by* RADiSA *converges almost surely to the optimal solution* ω^**, that is*

$$\lim_{t\to\infty}\left\|\omega^t - \omega^*\right\|^2 = 0 \quad \text{a.s.} \tag{A.68}$$

Moreover, if learning rate is defined as $\gamma_t := \frac{1}{t}$ *for* $t = 1, 2, \cdots$ *and the batch size is chosen such that* $B \geq \frac{1}{2\xi}$*, then the expected loss function errors* $\mathbb{E}[F(\omega^t) - F(\omega^*)]$ *of* RADiSA *converges to 0 at least with a sublinear convergence rate of order* $\mathcal{O}(1/t)$*, i.e.*

$$\mathbb{E}[F(\omega^t) - F(\omega^*)] \leq \frac{Q}{1+t}, \tag{A.69}$$

where constant Q *is defined in (1.5) with some positive constant* C_1' *taking the place of* C_1 *and* $c^t = d$*.*

PROOF. RADiSA is a special case of SODDA with $c^t = d$, $\mathscr{d}^t = N$. □

A.6. Constant Learning Rate with Feature Sampling

Proposition 3. *If Assumptions 1-4 hold, and the learning rate is constant* $\gamma_t = \gamma$ *such that* $BL\gamma QP \leq 1$ *and* $\gamma \leq 1$*, and the sequences* $(\mathscr{b}^t, c^t, \mathscr{d}^t)_{t=0}^{\infty}$ *satisfy the same conditions as in Theorem 3, then the loss function error sequence* $F(\omega^t) - F(\omega^*)$ *generated by SODDA satisfies*

$$\mathbb{E}\left[F(\omega^{t+1}) - F(\omega^*)\middle|\mathscr{F}^t\right] \leq \left(1 - \frac{2\xi B}{d}\gamma\right)\left[F(\omega^t) - F(\omega^*)\right] + C_4 B^4\gamma^2, \tag{A.70}$$

where C_4 *is a positive constant.*

PROOF. For $i = 1, 2, \cdots, B$, the expected value of $\|v^{t,i}\|$ given all the preceding information is

(A.71)

$$\mathbb{E}\left[\left\|v^{t,1}\right\| | \mathcal{F}^t, \mathcal{B}^t, \mathcal{C}^t, \mathcal{D}^t, \pi\right] = 0$$

$$\mathbb{E}\left[\left\|v^{t,i}\right\| | \mathcal{F}^t, \mathcal{B}^t, \mathcal{C}^t, \mathcal{D}^t, \pi, j_{11}^{(1)}, j_{12}^{(1)}, \cdots, j_{QP}^{(1)}, j_{11}^{(2)}, j_{12}^{(2)}, \cdots, j_{QP}^{(2)}, \cdots, j_{11}^{(i-2)}, j_{12}^{(i-2)}, \cdots, j_{QP}^{(i-2)}\right]$$
$$= \sum_{j_{QP}^{(i-1)}=1}^{n} \cdots \sum_{j_{11}^{(i-1)}=1}^{n} \frac{1}{n^{QP}}$$

(A.72)

$$\left\| \begin{pmatrix} \nabla_{\omega_{11}} f_{j_{11}^{(i-1)}}^{\pi_1^{-1}(1)} \left(x_{j_{11}^{(i-1)}}^{\pi_1^{-1}(1),1,1} \bar{\omega}_{11}^{t,i-1} \right) - \nabla_{\omega_{11}} f_{j_{11}^{(i-1)}}^{\pi_1^{-1}(1)} \left(x_{j_{11}^{(i-1)}}^{\pi_1^{-1}(1),1,1} \omega_{11}^{t} \right) \\ \vdots \\ \nabla_{\omega_{1P}} f_{j_{1P}^{(i-1)}}^{\pi_1^{-1}(P)} \left(x_{j_{1P}^{(i-1)}}^{\pi_1^{-1}(P),1,P} \bar{\omega}_{1P}^{t,i-1} \right) - \nabla_{\omega_{1P}} f_{j_{1P}^{(i-1)}}^{\pi_1^{-1}(P)} \left(x_{j_{1P}^{(i-1)}}^{\pi_1^{-1}(P),1,P} \omega_{1P}^{t} \right) \\ \vdots \\ \nabla_{\omega_{QP}} f_{j_{QP}^{(i-1)}}^{\pi_Q^{-1}(P)} \left(x_{j_{QP}^{(i-1)}}^{\pi_Q^{-1}(P),Q,P} \bar{\omega}_{QP}^{t,i-1} \right) - \nabla_{\omega_{QP}} f_{j_{QP}^{(i-1)}}^{\pi_Q^{-1}(P)} \left(x_{j_{QP}^{(i-1)}}^{\pi_Q^{-1}(P),Q,P} \omega_{QP}^{t} \right) \end{pmatrix} \right\|,$$

for $i = 2, 3, \cdots, B$.

We prove a bound of the expected value of $\sum_{i=1}^{B} \|v^{t,i}\|$ and $\sum_{i=1}^{B} \|v^{t,i}\|^2$ given $\mathcal{F}^t$ by induction.

Claim 6. *For any t, if $BL\gamma QP \leq 1$ and $\gamma \leq 1$, we have*

$$\sum_{i=1}^{B} \mathbb{E}\left[\left\|v^{t,i}\right\| \middle| \mathcal{F}^t\right] = \hat{\mathcal{O}}(B^3\gamma) \tag{A.73}$$

$$\sum_{i=1}^{B} \mathbb{E}\left[\left\|v^{t,i}\right\|^2 \middle| \mathcal{F}^t\right] = \hat{\mathcal{O}}(B^4\gamma^2) + \hat{\mathcal{O}}(B^7\gamma^4). \tag{A.74}$$

PROOF. By using (A.71) we have

$$\mathbb{E}\left[\left\|v^{t,1}\right\| \middle| \mathcal{F}^t\right] = \mathbb{E}\left[\mathbb{E}\left[\left\|v^{t,1}\right\| | \mathcal{F}^t, \mathcal{B}^t, \mathcal{C}^t, \mathcal{D}^t, \pi\right] \middle| \mathcal{F}^t\right] = 0. \tag{A.75}$$

For $i = 2, 3, \cdots, B$, using (A.72) gives

$$\begin{aligned}
&\mathbb{E}\left[\left\|v^{t,i}\right\| | \mathcal{F}^t, \mathcal{B}^t, \mathcal{C}^t, \mathcal{D}^t, \pi, j_{11}^{(1)}, j_{12}^{(1)}, \cdots, j_{QP}^{(1)}, j_{11}^{(2)}, j_{12}^{(2)}, \cdots, j_{QP}^{(2)}, \cdots, j_{11}^{(i-2)}, j_{12}^{(i-2)}, \cdots, j_{QP}^{(i-2)}\right] \\
&\leq \sum_{j_{QP}^{(i-1)}=1}^{n} \cdots \sum_{j_{11}^{(i-1)}=1}^{n} \sum_{q=1}^{Q}\sum_{p=1}^{P} \frac{1}{n^{QP}} \left\| \nabla_{\omega_{qp}} f_{j_{qp}^{(k-1)}}^{\pi_q^{-1}(p)} \left(x_{j_{qp}^{(k-1)}}^{\pi_q^{-1}(p),q,p} \bar{\omega}_{qp}^{t,k-1} \right) \right. \\
&\qquad \left. - \nabla_{\omega_{qp}} f_{j_{qp}^{(k-1)}}^{\pi_q^{-1}(p)} \left(x_{j_{qp}^{(k-1)}}^{\pi_q^{-1}(p),q,p} \omega_{qp}^{t} \right) \right\| \\
&= \sum_{q=1}^{Q}\sum_{p=1}^{P} \left(\frac{1}{n} \sum_{j_{q,\pi_q(p)}^{(i-1)}=1}^{n} \left\| \nabla_{\omega_{qp}} f_{j_{qp}^{(k-1)}}^{\pi_q^{-1}(p)} \left(x_{j_{qp}^{(k-1)}}^{\pi_q^{-1}(p),q,p} \bar{\omega}_{qp}^{t,k-1} \right) - \nabla_{\omega_{qp}} f_{j_{qp}^{(k-1)}}^{\pi_q^{-1}(p)} \left(x_{j_{qp}^{(k-1)}}^{\pi_q^{-1}(p),q,p} \omega_{qp}^{t} \right) \right\| \right) \\
&\leq \sum_{q=1}^{Q}\sum_{p=1}^{P} \left(\frac{1}{n} \cdot nL \left\| \bar{\omega}_{qp}^{t,i-1} - \bar{\omega}_{qp}^{t,0} \right\| \right) = \sum_{q=1}^{Q}\sum_{p=1}^{P} \left(L \left\| \bar{\omega}_{qp}^{t,i-1} - \bar{\omega}_{qp}^{t,0} \right\| \right)
\end{aligned} \tag{A.76}$$

$$= \sum_{q=1}^{Q}\sum_{p=1}^{P} L \left\| \gamma \left[(i-1)\mu_{qp}^{t} + v_{qp}^{t,1} + \cdots + v_{qp}^{t,i-1} \right] \right\|.$$

By using the law of iterated expectation and (A.76) we get

$$
\begin{aligned}
&\mathbb{E}\left[\left\|v^{t,i}\right\| |\mathcal{F}^t\right] = \mathbb{E}\left[\mathbb{E}\left[\left\|v^{t,i}\right\| |\mathcal{F}^t, , \mathcal{B}^t, \mathcal{C}^t, \mathcal{D}^t, \pi, j_{11}^{(1)} \cdots, j_{QP}^{(i-2)}\right] |\mathcal{F}^t\right] \\
&\leq \mathbb{E}\left[\sum_{q=1}^{Q}\sum_{p=1}^{P} L \left\| \gamma \left[(i-1)\mu_{qp}^t + \sum_{j=1}^{i-1} v_{qp}^{t,j}\right]\right\| \middle| \mathcal{F}^t\right] \\
&\leq L\gamma \sum_{q=1}^{Q}\sum_{p=1}^{P}\left(\mathbb{E}\left[\left\|(i-1)\mu_{qp}^t\right\| \middle| \mathcal{F}^t\right] + \sum_{j=1}^{i-1}\mathbb{E}\left[\left\|v_{qp}^{t,j}\right\| |\mathcal{F}^t\right]\right) \\
&\leq L\gamma QP\left((i-1)\,\mathbb{E}\left[\left\|\mu^t\right\| \middle| \mathcal{F}^t\right] + \sum_{j=1}^{i-1}\mathbb{E}\left[\left\|v^{t,j}\right\| |\mathcal{F}^t\right]\right).
\end{aligned} \tag{A.77}
$$

Let us define

$$
\begin{aligned}
a_i &:= \mathbb{E}\left[\left\|v^{t,i}\right\| \middle| \mathcal{F}^t\right] \\
\nu &= L\gamma QP \\
D_1 &:= \mathbb{E}\left[\left\|\mu^t\right\| \middle| \mathcal{F}^t\right].
\end{aligned}
$$

Then the recursive formula becomes

$$
\begin{aligned}
a_1 &= 0 \\
a_i &\leq \nu\left((i-1)D_1 + \sum_{j=1}^{i-1} a_j\right),
\end{aligned} \tag{A.78}
$$

for $i = 2, 3, \cdots, B$. Let us define $\bar{a}_i$ as

$$
\bar{a}_i = \begin{cases} 0, & i = 1 \\ \nu\left((i-1)D_1 + \sum_{j=1}^{i-1}\bar{a}_j\right), & i \neq 1. \end{cases} \tag{A.79}
$$

Now, let us show that $a_i \leq \bar{a}_i$ for $i = 1, 2, \cdots, B$ by induction. When $i = 1$, applying the definitions of a_i and $\bar{a}_i$ yields $a_1 = \bar{a}_1$. Assume that when $i = 1, 2, \cdots, k-1$, $a_i \leq \bar{a}_i$ holds true. Now, consider $\bar{a}_k$. Since $\nu, D_1, a_i \geq 0$ for any i, by using (A.78) and (A.79) we have

$$\bar{a}_k = \nu \left((k-1)D_1 + \sum_{j=1}^{k-1} \bar{a}_j \right) \geq \nu \left((k-1)D_1 + \sum_{j=1}^{k-1} a_j \right) \geq a_k.$$

Therefore, $S_l \leq \bar{S}_l$, where we define $S_l = \sum_{i=1}^{l} a_i$ and $\bar{S}_l = \sum_{i=1}^{l} \bar{a}_i$. Summing up all the recursive equations for $\bar{a}_i$ in (A.79) up to l implies

$$\bar{S}_l = \frac{l(l-1)}{2} \nu D_1 + \nu \left(\bar{S}_1 + \cdots + \bar{S}_{l-1} \right) \tag{A.80}$$

and

$$\bar{S}_{l+1} - \bar{S}_l = l\nu D_1 + \nu \bar{S}_l,$$

which in turn yields

$$\frac{\bar{S}_{l+1}}{(1+\nu)^{l+1}} - \frac{\bar{S}_l}{(1+\nu)^l} = \frac{l\nu D_1}{(1+\nu)^{l+1}}.$$

By summing up all increments for $l = 1, \cdots, B-1$, we obtain

$$\frac{\bar{S}_B}{(1+\nu)^B} = \frac{\bar{S}_B}{(1+\nu)^B} - \frac{\bar{S}_1}{(1+\nu)} = \frac{\nu D_1}{(1+\nu)} \sum_{l=1}^{B} \frac{l}{(1+\nu)^l} = \frac{D_1 \left[(1+\nu)^B - 1 - \nu B \right]}{\nu (1+\nu)^B},$$

which in turn yields

$$\sum_{i=1}^{B} \mathbb{E} \left[\left\| v^{t,i} \right\| \middle| \mathcal{F}^t \right] = \sum_{i=1}^{B} a_i = S_B \leq \bar{S}_B = \frac{D_1 \left[(1+\nu)^B - 1 - \nu B \right]}{\nu}.$$

Since $\binom{B}{l} = \frac{B!}{l!(B-l)!} \leq B^l$, we obtain

$$(1+\nu)^B = \sum_{l=0}^{B} \binom{B}{l} \nu^l \leq \sum_{l=0}^{B} (B\nu)^l,$$

which in turn yields

$$\sum_{i=1}^{B} \mathbb{E}\left[\left\| v^{t,i} \right\| \middle| \mathcal{F}^t \right] \leq \frac{D_1 \sum_{l=2}^{B} (B\nu)^l}{\nu}. \tag{A.81}$$

Substituting the definitions of ν and C back into (A.81) gives

$$\sum_{i=1}^{B} \mathbb{E}\left[\left\| v^{t,i} \right\| \middle| \mathcal{F}^t \right] \leq \frac{\mathbb{E}\left[\left\| \mu^t \right\| \middle| \mathcal{F}^t \right] \sum_{l=2}^{B} (BL\gamma QP)^l}{L\gamma QP}. \tag{A.82}$$

By using the conditional Jensen's inequality and (A.45) we get

$$\mathbb{E}\left[\left\| \mu^t \right\| \middle| \mathcal{F}^t \right] = \mathbb{E}\left[\sqrt{\left\| \mu^t \right\|^2} \middle| \mathcal{F}^t \right] \leq \sqrt{\mathbb{E}\left[\left\| \mu^t \right\|^2 \middle| \mathcal{F}^t \right]} = \sqrt{\hat{\mathcal{O}}(1) + \hat{\mathcal{O}}(\gamma^2)} = \hat{\mathcal{O}}(1). \tag{A.83}$$

The last equality holds due to the property that $\gamma \leq 1$. Moreover, since $BL\gamma QP \leq 1$, we have

$$\sum_{l=2}^{B} (BL\gamma QP)^l = B \cdot \hat{\mathcal{O}}(B^2\gamma^2) = \hat{\mathcal{O}}(B^3\gamma^2). \tag{A.84}$$

Substituting (A.83) and (A.84) in (A.82) implies the claim in (A.73).

Now, let us proceed to find an upper bound for $\sum_{i=1}^{B} \mathbb{E}\left[\left\| v^{t,i} \right\|^2 \middle| \mathcal{F}^t \right]$. From (A.46), we have

$$\mathbb{E}\left[\left\| v^{t,1} \right\|^2 \middle| \mathcal{F}^t \right] = 0$$

$$\mathbb{E}\left[\left\| v^{t,i} \right\|^2 \middle| \mathcal{F}^t \right] \leq iL^2\gamma^2 QP \left((i-1)^2 \, \mathbb{E}\left[\left\| \mu^t \right\|^2 \middle| \mathcal{F}^t \right] + \sum_{j=1}^{i-1} \mathbb{E}\left[\left\| v^{t,j} \right\|^2 \middle| \mathcal{F}^t \right] \right).$$

Let us define

$$b_i := \mathbb{E}\left[\left\|v^{t,i}\right\|^2 \middle| \mathcal{F}^t\right]$$

$$\theta := L^2\gamma^2 QP$$

$$D_2 := \mathbb{E}\left[\left\|\mu^t\right\|^2 \middle| \mathcal{F}^t\right].$$

Then the recursive formula becomes

$$\begin{aligned} b_1 &= 0 \\ b_i &\leq i\theta\left((i-1)^2 D_2 + \sum_{j=1}^{i-1} b_j\right), \end{aligned} \tag{A.85}$$

for $i = 2, 3, \cdots, B$. Let us define $\bar{b}_i$ as

$$\bar{b}_i = \begin{cases} 0, & i = 1 \\ i\theta\left((i-1)^2 D_2 + \sum_{j=1}^{i-1} \bar{b}_j\right), & i \neq 1. \end{cases} \tag{A.86}$$

As before we derive $b_i \leq \bar{b}_i$ for $i = 1, 2, \cdots, B$. Therefore, $\mathcal{S}_l \leq \bar{\mathcal{S}}_l$, where we define $\mathcal{S}_l = \sum_{i=1}^{l} b_i$ and $\bar{\mathcal{S}}_l = \sum_{i=1}^{l} \bar{b}_i$. Summing up all the recursive equations for $\bar{b}_i$ in (A.86) up to l implies

$$\bar{\mathcal{S}}_l = \theta D_2 \sum_{i=1}^{l-1}(i+1)i^2 + \theta \sum_{i=1}^{l-1}(i+1)\bar{\mathcal{S}}_i, \tag{A.87}$$

and

$$\bar{\mathcal{S}}_{l+1} - \bar{\mathcal{S}}_l = \theta D_2 (l+1)l^2 + \theta(l+1)\bar{\mathcal{S}}_l,$$

which in turn yields

$$\frac{\bar{\mathcal{S}}_{l+1}}{\Pi_{i=1}^{l+1}(1+i\theta)} - \frac{\bar{\mathcal{S}}_{l}}{\Pi_{i=1}^{l}(1+i\theta)} = \frac{\theta D_2 (l+1) l^2}{\Pi_{i=1}^{l+1}(1+i\theta)}.$$

By summing up all increments for $l = 1, 2, \cdots, B-1$, we obtain

$$\frac{\bar{\mathcal{S}}_{B}}{\Pi_{i=1}^{B}(1+i\theta)} = \frac{\bar{\mathcal{S}}_{B}}{\Pi_{i=1}^{B}(1+i\theta)} - \frac{\bar{\mathcal{S}}_{1}}{(1+\theta)} = \theta D_2 \left[\sum_{l=1}^{B-1} \frac{(l+1)l^2}{\Pi_{i=1}^{l+1}(1+i\theta)} \right],$$

which in turn yields

$$\begin{aligned} \sum_{i=1}^{B} \mathbb{E}\left[\left\| v^{t,i} \right\|^2 \middle| \mathcal{F}^t \right] = \sum_{i=1}^{B} b_i = \mathcal{S}_l \leq \bar{\mathcal{S}}_B &= \theta D_2 \left[\sum_{l=1}^{B-1} \left((l+1) l^2 \Pi_{i=l+2}^{B} (1+i\theta) \right) \right] \\ &\leq \theta D_2 B (B-1)^2 \left[\sum_{l=1}^{B-1} \left(\Pi_{i=l+2}^{B} (1+i\theta) \right) \right], \end{aligned} \tag{A.88}$$

where we denote $\Pi_{i=B}^{B+1}(1+i\theta) = 1$. Since

$$\sum_{l=1}^{B-1} \left(\Pi_{i=l+2}^{B} (1+i\theta) \right) \leq \sum_{l=1}^{B-1} \left(\Pi_{i=1}^{B} (1+i\theta) \right) \leq \sum_{l=1}^{B-1} (1+B\theta)^B = (B-1)(1+B\theta)^B,$$

it in turn yields

$$\sum_{i=1}^{B} \mathbb{E}\left[\left\| v^{t,i} \right\|^2 \middle| \mathcal{F}^t \right] \leq \theta D_2 B (B-1)^3 (1+B\theta)^B. \tag{A.89}$$

Substituting the definitions of θ and D_2 back into (A.89) gives

$$\sum_{i=1}^{B} \mathbb{E}\left[\left\| v^{t,i} \right\|^2 \middle| \mathcal{F}^t \right] \leq L^2 \gamma^2 Q P \, \mathbb{E}\left[\left\| \mu^t \right\|^2 \middle| \mathcal{F}^t \right] B^4 (B L^2 \gamma^2 Q P + 1)^B. \tag{A.90}$$

Since $BL\gamma QP \leq 1$ and $QP \geq 1$, we conclude

$$B^2L^2\gamma^2QP \leq (BL\gamma QP)^2 \leq 1,$$

which in turn yields

$$\begin{aligned}(1 + BL^2\gamma^2QP)^B &= 1 + \sum_{i=1}^{B} \binom{B}{i} (BL^2\gamma^2QP)^i \leq 1 + \sum_{i=1}^{B} B^i (BL^2\gamma^2QP)^i \\ &= 1 + \sum_{i=1}^{B} (B^2L^2\gamma^2QP)^i = 1 + \sum_{i=1}^{B} \hat{\mathcal{O}}(B^2L^2\gamma^2QP) \\ &= 1 + \hat{\mathcal{O}}(B^3L^2\gamma^2QP). \end{aligned} \tag{A.91}$$

Combining (A.45) and (A.91) gives

$$\sum_{i=1}^{B} \mathbb{E}\left[\left\| v^{t,i} \right\|^2 \middle| \mathcal{F}^t \right] = \hat{\mathcal{O}}(B^4\gamma^2)(1 + \hat{\mathcal{O}}(B^3L^2\gamma^2QP)) = \hat{\mathcal{O}}(B^4\gamma^2) + \hat{\mathcal{O}}(B^7\gamma^4).$$

This completes the proof of the claim. □

By using (A.28), (A.45), (A.73), (A.74) and Lipschitz continuity of $\nabla F(\omega)$ we have

$$\begin{aligned}
&\mathbb{E}\left[F(\omega^{t+1})|\mathcal{F}^t\right] \leq F(\omega^t) + \nabla F(\omega^t)^T \mathbb{E}\left[\left(\omega^{t+1}-\omega^t\right)|\mathcal{F}^t\right] + \frac{L}{2}\mathbb{E}\left[\left\|\omega^{t+1}-\omega^t\right\|^2|\mathcal{F}^t\right] \\
&= F(\omega^t) + \nabla F(\omega^t)^T\left\{-\gamma B\frac{c^t}{d}\nabla F(\omega^t) - \gamma\sum_{i=1}^{B}\mathbb{E}[v^{t,i}\mid\mathcal{F}^t]\right\} \\
&\qquad + \frac{L}{2}\gamma^2(B+1)\left\{B^2\,\mathbb{E}\left[\left\|\mu^t\right\|^2|\mathcal{F}^t\right] + \sum_{i=1}^{B}\mathbb{E}\left[\left\|v^{t,i}\right\|^2|\mathcal{F}^t\right]\right\} \\
&\leq F(\omega^t) - \gamma\frac{B}{d}\left\|\nabla F(\omega^t)\right\|^2 + \gamma\left\|\nabla F(\omega^t)\right\|\sum_{i=1}^{B}\mathbb{E}\left[\left\|v^{t,i}\right\||\mathcal{F}^t\right] \\
&\qquad + \frac{L}{2}\gamma^2(B+1)\left\{B^2\,\mathbb{E}\left[\left\|\mu^t\right\|^2|\mathcal{F}^t\right] + \sum_{i=1}^{B}\mathbb{E}\left[\left\|v^{t,i}\right\|^2|\mathcal{F}^t\right]\right\} \\
&\leq F(\omega^t) - \gamma\frac{B}{d}\left\|\nabla F(\omega^t)\right\|^2 + \hat{\mathcal{O}}(B^3\gamma^2) + \hat{\mathcal{O}}(B\gamma^2)\left\{\hat{\mathcal{O}}(B^2) + \hat{\mathcal{O}}(B^4\gamma^2) + \hat{\mathcal{O}}(B^7\gamma^4)\right\} \\
&= F(\omega^t) - \gamma\frac{B}{d}\left\|\nabla F(\omega^t)\right\|^2 + \hat{\mathcal{O}}(B^3\gamma^2) + \hat{\mathcal{O}}(B^5\gamma^4) + \hat{\mathcal{O}}(B^8\gamma^6).
\end{aligned}$$

Since $LQP \geq 1$ and $BL\gamma QP \leq 1$, the above equation becomes

$$\mathbb{E}\left[F(\omega^{t+1})|\mathcal{F}^t\right] \leq F(\omega^t) - \gamma\frac{B}{d}\left\|\nabla F(\omega^t)\right\|^2 + \hat{\mathcal{O}}(B^4\gamma^2). \tag{A.92}$$

Subtracting $F(\omega^*)$ from both sides of (A.92) and applying (A.56) yields the claim in (A.70), where C_2 is a positive constant. □

Proof of Theorem 5.

PROOF. We use the relationship in (A.70) to construct a supermartingale sequence. Define the stochastic processes α^t and β^t as

$$\alpha^t := \left[F(\omega^t) - F(\omega^*)\right] \times \mathbb{1}_{\left\{\min_{u\leq t} F(\omega^u) - F(\omega^*) > \frac{C_2 dB^3\gamma}{2\xi}\right\}} \tag{A.93}$$

$$\beta^t := \frac{2\xi B}{d}\gamma \left[F(\omega^t) - F(\omega^*) - \frac{C_2 dB^3\gamma}{2\xi}\right] \times \mathbb{1}_{\left\{\min_{u\leq t} F(\omega^u) - F(\omega^*) > \frac{C_2 dB^3\gamma}{2\xi}\right\}}. \tag{A.94}$$

The process α^t tracks the optimality gap $F(\omega^t) - F(\omega^*)$ until the gap becomes smaller than $\frac{C_2 dB^3\gamma}{2\xi}$ for the first time. Notice that the stochastic process α^t is never negative. Likewise, the same properties hold for β^t. Based on the relationship in (A.70) and the definitions of stochastic processes α^t and β^t in (A.93) and (A.94), we obtain that for all $t \geq 0$

$$\mathbb{E}\left[\alpha^{t+1} \middle| \mathcal{F}^t\right] \leq \alpha^t - \beta^t. \tag{A.95}$$

Given the relationship in (A.95) and non-negativity of stochastic processes α^t and β^t we obtain that α^t is supermartingale. The supermartingale convergence theorem yields

$$\begin{aligned} &\text{(i)} \quad \alpha^t \text{ converges to a limit a.s., and} \\ &\text{(ii)} \quad \sum_{t=1}^{\infty} \beta^t < \infty. \quad \text{a.s.} \end{aligned} \tag{A.96}$$

Property (A.96) implies that the sequence β^t is converging to 0 almost surely, i.e.,

$$\lim_{t\to\infty} \beta^t = 0 \quad \text{a.s.} \tag{A.97}$$

Based on the definition of β^t in (A.94), the limit in (A.97) is true if one of the following events holds:

$$\text{(i)} \quad \text{the indicator function is 0 after large } t,$$

$$\text{(ii)} \quad \lim_{t\to\infty} F(\omega^t) - F(\omega^*) - \frac{C_2 d B^3 \gamma}{2\xi} = 0.$$

From either one of these two events we conclude that

$$\liminf_{t\to\infty} F(\omega^t) - F(\omega^*) \leq \frac{C_2 d B^3 \gamma}{2\xi} \quad \text{a.s.} \tag{A.98}$$

Therefore, the claim in (1.6) is valid. The result in (A.98) shows that the loss function value sequence $F(\omega^t)$ almost sure converges to a neighborhood of the optimal loss function value $F(\omega^*)$.

We proceed to prove the result in (1.7). We compute the expected value of (A.70) given $\mathcal{F}^0$ to obtain

$$\mathbb{E}\left[F(\omega^{t+1}) - F(\omega^*)\right] \leq \left(1 - \frac{2\xi B}{d}\gamma\right) \mathbb{E}\left[F(\omega^t) - F(\omega^*)\right] + C_2 B^4 \gamma^2. \tag{A.99}$$

Rewriting the relationship in (A.99) for step $t-1$ gives

$$\mathbb{E}\left[F(\omega^t) - F(\omega^*)\right] \leq \left(1 - \frac{2\xi B}{d}\gamma\right) \mathbb{E}\left[F(\omega^{t-1}) - F(\omega^*)\right] + C_2 B^4 \gamma^2. \tag{A.100}$$

Substituting the upper bound in (A.100) for the expectation of $F(\omega^t) - F(\omega^*)$ in (A.99) implies

$$\mathbb{E}\left[F(\omega^{t+1}) - F(\omega^*)\right] \leq \left(1 - \frac{2\xi B}{d}\gamma\right)^2 \mathbb{E}\left[F(\omega^{t-1}) - F(\omega^*)\right] + C_2 B^4 \gamma^2 \left(1 + \left(1 - \frac{2\xi B}{d}\gamma\right)\right). \tag{A.101}$$

By recursively applying steps (A.100) and (A.101) we can bound the expectation of $F(\omega^{t+1}) - F(\omega^*)$ in terms of the initial loss function error $F(\omega^0) - F(\omega^*)$ as

(A.102)

$$\mathbb{E}\left[F(\omega^{t+1}) - F(\omega^*)\right] \leq \left(1 - \frac{2\xi B}{d}\gamma\right)^{t+1} \left[F(\omega^0) - F(\omega^*)\right] + C_2 B^4 \gamma^2 \sum_{u=0}^{t} \left(1 - \frac{2\xi B}{d}\gamma\right)^u.$$

Substituting t by $t-1$ and simplifying the sum in the right-hand side of (A.102) yields

(A.103)

$$\mathbb{E}\left[F(\omega^t) - F(\omega^*)\right] \leq \left(1 - \frac{2\xi B}{d}\gamma\right)^t \left[F(\omega^0) - F(\omega^*)\right] + \frac{C_2 d B^3 \gamma}{2\xi}\left[1 - \left(1 - \frac{2\xi B}{d}\gamma\right)^t\right].$$

Since $\gamma < \frac{d}{2\xi B}$, the term $1 - \left(1 - \frac{2\xi B}{d}\gamma\right)^t$ in the right-hand side of (A.103) is strictly smaller than 1 and the claim in (1.7) follows. □

A.7. Convergence of Constant L.R. with Feature Sampling

Proof of Theorem 6.

PROOF. Given $\mathscr{b}^t = d$ and $\mathscr{d}^t = N$, applying Claim 4 implies

$$\mathbb{E}\left[\frac{1}{N}\sum_{j=1}^{N}\bar{\nabla}_{\mathcal{C}^t} f_j(x_j\omega^t)\middle|\mathscr{F}^t\right] = \frac{c^t}{N}\nabla F(\omega^t), \tag{A.104}$$

$$\begin{aligned}\mathbb{E}\left[\left\|\frac{1}{N}\sum_{j=1}^{N}\bar{\nabla}_{\mathcal{C}^t} f_j(x_j\omega^t)\right\|^2\middle|\mathscr{F}^t\right] &= \frac{1}{\binom{d}{c^t}}\sum_{\mathcal{C}^t}\left\|\frac{1}{N}\sum_{j=1}^{N}\bar{\nabla}_{\mathcal{C}^t} f_j(x_j\omega^t)\right\|^2 \\ &= \frac{1}{\binom{d}{c^t}}\binom{d-1}{c^t-1}\left\|\frac{1}{N}\sum_{j=1}^{N}\bar{\nabla}_{\mathcal{C}^t} f_j(x_j\omega^t)\right\|^2 = \frac{c^t}{d}\left\|\nabla F(\omega^t)\right\|^2, \end{aligned} \tag{A.105}$$

which in turn gives an upper bound to $\mathbb{E}\left[\left\|\frac{1}{N}\sum_{j=1}^{N}\bar{\nabla}_{\mathcal{C}^t} f_j(x_j\omega^t)\right\|\middle|\mathscr{F}^t\right]$; that is

(A.106)

$$\mathbb{E}\left[\left\|\frac{1}{N}\sum_{j=1}^{N}\bar{\nabla}_{\mathcal{C}^t} f_j(x_j\omega^t)\right\|\middle|\mathscr{F}^t\right] \leq \sqrt{\mathbb{E}\left[\left\|\frac{1}{N}\sum_{j=1}^{N}\bar{\nabla}_{\mathcal{C}^t} f_j(x_j\omega^t)\right\|^2\middle|\mathscr{F}^t\right]} = \sqrt{\frac{c^t}{d}}\left\|\nabla F(\omega^t)\right\|.$$

On the one hand, given $BL\gamma QP \leq 1$, we find that

$$\sum_{l=2}^{B}(BL\gamma QP) \leq (B-1)(BL\gamma QP)^2 = (B-1)B^2L^2\gamma^2Q^2P^2. \tag{A.107}$$

By combining this expression with (A.82), we conclude that

$$\mathbb{E}\left[\left\|\sum_{i=1}^{B}v^{t,i}\right\|\middle|\mathscr{F}^t\right] \leq \sum_{i=1}^{B}\mathbb{E}\left[\left\|v^{t,i}\right\|\middle|\mathscr{F}^t\right] \leq \mathbb{E}\left[\left\|\mu^t\right\|\middle|\mathscr{F}^t\right](B-1)B^2L\gamma QP. \tag{A.108}$$

On the other hand, given $\gamma \leq 1$, from expression (A.91), we find that

$$(1+BL^2\gamma^2QP)^B \leq 1+\sum_{i=1}^{B}(B^2L^2\gamma^2QP)^i = 1+B^3L^2QP. \tag{A.109}$$

By applying (A.109) to (A.90), we deduce that

(A.110)
$$\mathbb{E}\left[\left\|\sum_{i=1}^{B} v^{t,i}\right\|^2 \middle| \mathcal{F}^t\right] \leq B\sum_{i=1}^{B}\mathbb{E}\left[\left\|v^{t,i}\right\|^2 \middle| \mathcal{F}^t\right] \leq B^5(1+B^3L^2QP)L^2\gamma^2QP\,\mathbb{E}\left[\left\|\mu^t\right\|^2 \middle| \mathcal{F}^t\right].$$

Second, let us evaluate the error after each iteration. By summing up all increments in iteration t, we obtain

(A.111)
$$\begin{aligned}
\mathbb{E}\left[\left\|\omega^{t+1}-\omega^*\right\|^2 \middle| \mathcal{F}^t\right] &= \mathbb{E}\left[\left\|\omega^t-\gamma\left(B\mu^t+\sum_{i=1}^{B}v^{t,i}\right)-\omega^*\right\|^2 \middle| \mathcal{F}^t\right] \\
&= \left\|\omega^t-\omega^*\right\|^2 - 2\left\langle \mathbb{E}\left[\gamma\left(B\mu^t+\sum_{i=1}^{B}v^{t,i}\right)\middle| \mathcal{F}^t\right], \omega^t-\omega^*\right\rangle \\
&\quad + \mathbb{E}\left[\left\|\gamma\left(B\mu^t+\sum_{i=1}^{B}v^{t,i}\right)\right\|^2 \middle| \mathcal{F}^t\right] \\
&\leq \left\|\omega^t-\omega^*\right\|^2 - 2\gamma B\left\langle \mathbb{E}\left[\mu^t\middle|\mathcal{F}^t\right], \omega^t-\omega^*\right\rangle + 2\gamma\,\mathbb{E}\left[\left\|\sum_{i=1}^{B}v^{t,i}\right\| \middle| \mathcal{F}^t\right]\left\|\omega^t-\omega^*\right\| \\
&\quad + 2\gamma^2B^2\,\mathbb{E}\left[\left\|\mu^t\right\|^2\middle|\mathcal{F}^t\right] + 2\gamma^2\,\mathbb{E}\left[\left\|\sum_{i=1}^{B}v^{t,i}\right\|^2\middle|\mathcal{F}^t\right].
\end{aligned}$$

Based on (A.104), (A.105), (A.106), (A.108) and (A.110), (A.111) can be further simplified as

(A.112)

$$\begin{aligned}
&\mathbb{E}\left[\left\|\omega^{t+1}-\omega^*\right\|^2\middle|\mathcal{F}^t\right] \leq \left\|\omega^t-\omega^*\right\|^2 - \frac{2\gamma Bc^t}{d}\left\langle\nabla F(\omega^t),\omega^t-\omega^*\right\rangle + \frac{2\gamma^2B^2c^t}{d}\left\|\nabla F(\omega^t)\right\|^2 \\
&\quad + 2(B-1)B^2L\gamma^2QP\,\mathbb{E}\left[\left\|\mu^t\right\|\middle|\mathcal{F}^t\right]\left\|\omega^t-\omega^*\right\| \\
&\quad + 2B^5(1+B^3L^2QP)L^2\gamma^4QP\,\mathbb{E}\left[\left\|\mu^t\right\|^2\middle|\mathcal{F}^t\right] \\
&\quad \leq \left\|\omega^t-\omega^*\right\|^2 - \frac{2\gamma Bc^t}{d}\left\langle\nabla F(\omega^t),\omega^t-\omega^*\right\rangle + \frac{2\gamma^2B^2c^t}{d}\left\|\nabla F(\omega^t)\right\|^2 \\
&\quad + 2(B-1)B^2L\gamma^2QP\sqrt{\frac{c^t}{d}}\left\|\nabla F(\omega^t)\right\|\left\|\omega^t-\omega^*\right\| \\
&\quad + 2B^5(1+B^3L^2QP)L^2\gamma^4QP\frac{c^t}{d}\left\|\nabla F(\omega^t)\right\|^2.
\end{aligned}$$

Recalling the Lipschitz continuity of the gradient of the objective function stated in Assumption 2, we have that [(Nesterov, 2013),Theorem 2.1.5]

$$\frac{1}{L}\left\|\nabla F(\omega^t)\right\|^2 = \frac{1}{L}\left\|\nabla F(\omega^t)-\nabla F(\omega^*)\right\|^2 \leq \left\langle\nabla F(\omega^t)-\nabla F(\omega^*),\omega^t-\omega^*\right\rangle. \tag{A.113}$$

Because $F(\omega)$ is strongly convex by Assumption 1, we have

$$\frac{1}{\xi}\left\|\nabla F(\omega^t)\right\| = \frac{1}{\xi}\left\|\nabla F(\omega^t)-\nabla F(\omega^*)\right\| \geq \left\|\omega^t-\omega^*\right\|. \tag{A.114}$$

By using (A.113) and (A.114), (A.112) can be reformulated as

(A.115)

$$\begin{aligned}
\mathbb{E}\left[\left\|\omega^{t+1}-\omega^*\right\|^2\middle|\mathcal{F}^t\right] &\leq \left\|\omega^t-\omega^*\right\|^2-\frac{2\gamma Bc^t}{Ld}\left\|\nabla F(\omega^t)\right\|^2+\frac{2\gamma^2B^2c^t}{d}\left\|\nabla F(\omega^t)\right\|^2 \\
&\quad+\frac{2(B-1)B^2L\gamma^2QP}{\xi}\sqrt{\frac{c^t}{d}}\left\|\nabla F(\omega^t)\right\|^2+2B^5(1+B^3L^2QP)L^2\gamma^4QP\frac{c^t}{d}\left\|\nabla F(\omega^t)\right\|^2 \\
&=\left\|\omega^t-\omega^*\right\|^2+\left(-\frac{2\gamma Bc^t}{Ld}+\frac{2\gamma^2B^2c^t}{d}+\frac{2(B-1)B^2L\gamma^2QP}{\xi}\sqrt{\frac{c^t}{d}}\right. \\
&\quad\left.+2B^5(1+B^3L^2QP)L^2\gamma^4QP\frac{c^t}{d}\right)\left\|\nabla F(\omega^t)\right\|^2 \\
&=\left\|\omega^t-\omega^*\right\|^2+A(t)\left\|\nabla F(\omega^t)\right\|^2,
\end{aligned}$$

with

$$A(t)=-\frac{2\gamma Bc^t}{Ld}+\frac{2\gamma^2B^2c^t}{d}+\frac{2(B-1)B^2L\gamma^2QP}{\xi}\sqrt{\frac{c^t}{d}}+2B^5(1+B^3L^2QP)L^2\gamma^4QP\frac{c^t}{d}.$$

Therefore, $\mathbb{E}\left[\left\|\omega^{t+1}-\omega^*\right\|^2\middle|\mathcal{F}^t\right]\leq\left\|\omega^t-\omega^*\right\|^2$ as long as $A(t)\leq 0$ for all t, $BL\gamma QP\leq 1$ and $\gamma\leq 1$.

In view of Assumption 2 we have

(A.116)

$$\begin{aligned}
\mathbb{E}\left[F(\omega^{t+1})\middle|\mathcal{F}^t\right] &\leq \mathbb{E}\left[F(\omega^t)+\left\langle\nabla F(\omega^t),\omega^{t+1}-\omega^t\right\rangle+\frac{L}{2}\left\|\omega^{t+1}-\omega^t\right\|^2\middle|\mathcal{F}^t\right] \\
&= F(\omega^t)+\mathbb{E}\left[\left\langle\nabla F(\omega^t),-\gamma B\mu^t-\gamma\sum_{i=1}^{t}v^{t,i}\right\rangle\middle|\mathcal{F}^t\right]+\frac{L}{2}\mathbb{E}\left[\left\|\gamma B\mu^t+\gamma\sum_{i=1}^{B}v^{t,i}\right\|^2\middle|\mathcal{F}^t\right] \\
&\leq F(\omega^t)-\gamma B\left\langle\nabla F(\omega^t),\mathbb{E}\left[\mu^t\middle|\mathcal{F}^t\right]\right\rangle+\gamma\left\|\nabla F(\omega^t)\right\|\mathbb{E}\left[\left\|\sum_{i=1}^{B}v^{t,i}\right\|\middle|\mathcal{F}^t\right] \\
&\quad+L\gamma^2B^2\,\mathbb{E}\left[\left\|\mu^t\right\|^2\middle|\mathcal{F}^t\right]+L\gamma^2\,\mathbb{E}\left[\left\|\sum_{i=1}^{B}v^{t,i}\right\|^2\middle|\mathcal{F}^t\right].
\end{aligned}$$

Substituting (A.104), (A.105), (A.108) and (A.110) implies

(A.117)

$$\begin{aligned}
\mathbb{E}\left[F(\omega^{t+1})\middle|\mathcal{F}^t\right] &\leq F(\omega^t)-\frac{\gamma Bc^t}{d}\left\|\nabla F(\omega^t)\right\|^2+(B-1)B^2L\gamma^2QP\sqrt{\frac{c^t}{d}}\left\|\nabla F(\omega^t)\right\|^2 \\
&\quad+\frac{L\gamma^2B^2c^t}{d}\left\|\nabla F(\omega^t)\right\|^2+B^5(1+B^3L^2QP)L^3\gamma^4QP\frac{c^t}{d}\left\|\nabla F(\omega^t)\right\|^2 \\
&= F(\omega^t)+\left(-\frac{\gamma Bc^t}{d}+(B-1)B^2L\gamma^2QP\sqrt{\frac{c^t}{d}}+\frac{L\gamma^2B^2c^t}{d}\right. \\
&\quad\left.+B^5(1+B^3L^2QP)L^3\gamma^4QP\frac{c^t}{d}\right)\left\|\nabla F(\omega^t)\right\|^2 \\
&= F(\omega^t)+B(t)\left\|\nabla F(\omega^t)\right\|^2.
\end{aligned}$$

A similar requirement is needed in (A.117) as that in (A.115), i.e. $B(t)<0$ for all t.

Let us denote $\triangle_t=F(\omega^t)-F(\omega^*)$. Then if $A(t)\leq 0$ for all t, we obtain

$$\triangle_t=F(\omega^t)-F(\omega^*)\leq\left\langle\nabla F(\omega^t),\omega^t-\omega^*\right\rangle\leq\left\|\omega^0-\omega^*\right\|\left|\nabla F(\omega^t)\right\|.$$

Thus, if $B(t) < 0$ for all t and by using the law of iterated expectation, (A.117) becomes

$$\mathbb{E}\left[\triangle_{t+1}\right] \leq \mathbb{E}\left[\triangle_k\right] + \min_t B(t)\, \mathbb{E}\left[\left\|\triangledown F(\omega^t)\right\|^2\right] \leq \mathbb{E}\left[\triangle_k\right] + \frac{\min_t B(t)}{\left\|\omega^0 - \omega^*\right\|^2}\, \mathbb{E}\left[\triangle_k^2\right],$$

which in turn yields

$$\frac{1}{\mathbb{E}\left[\triangle_{t+1}\right]} \geq \frac{1}{\mathbb{E}\left[\triangle_t\right]} - \frac{\min_t B(t)}{\left\|\omega^0 - \omega^*\right\|^2} \frac{\mathbb{E}\left[\triangle_t\right]}{\mathbb{E}\left[\triangle_{t+1}\right]} \geq \frac{1}{\mathbb{E}\left[\triangle_t\right]} - \frac{\min_t B(t)}{\left\|\omega^0 - \omega^*\right\|^2}.$$

Summing up these inequalities, we get

$$\frac{1}{\mathbb{E}\left[\triangle_{t+1}\right]} \geq \frac{1}{\triangle_0} - \frac{\min_t B(t)}{\left\|\omega^0 - \omega^*\right\|^2}(t+1),$$

which in turn yields

$$\lim_{t\to\infty} \mathbb{E}\left[\triangle_{t+1}\right] = 0. \tag{A.118}$$

Hence, (1.8) follows from (A.118) and similar reasoning as Theorem 3, provided $A(t) \leq 0$, $B(t) < 0$ for all t and $BL\gamma QP \leq 1$, i.e.

$$\frac{2\gamma B c^t}{Ld} \geq \frac{2\gamma^2 B^2 c^t}{d} + \frac{2(B-1)B^2 L\gamma^2 QP}{\xi}\sqrt{\frac{c^t}{d}} + 2B^5(1 + B^3L^2QP)L^2\gamma^4 QP\frac{c^t}{d} \tag{A.119}$$

$$\frac{\gamma B c^t}{d} > (B-1)B^2 L\gamma^2 QP\sqrt{\frac{c^t}{d}} + \frac{L\gamma^2 B^2 c^t}{d} + B^5(1 + B^3L^2QP)L^3\gamma^4 QP\frac{c^t}{d} \tag{A.120}$$

$$BL\gamma QP \leq 1 \tag{A.121}$$

$$\gamma \leq 1. \tag{A.122}$$

By multiplying (A.119) and (A.120) by $\frac{1}{2\gamma B}$ and $\frac{1}{\gamma B}$, respectively, we have that

$$\frac{c^t}{Ld} \geq \frac{\gamma B c^t}{d} + \frac{(B-1)BL\gamma QP}{\xi}\sqrt{\frac{c^t}{d}} + B^4(1+B^3L^2QP)L^2\gamma^3 QP\frac{c^t}{d}, \tag{A.123}$$

$$\frac{c^t}{d} > (B-1)BL\gamma QP\sqrt{\frac{c^t}{d}} + \frac{L\gamma B c^t}{d} + B^4(1+B^3L^2QP)L^3\gamma^3 QP\frac{c^t}{d}. \tag{A.124}$$

Note that if we are able to find a constant learning rate γ satisfying

(A.125)
$$\bar{A}_1 = \frac{\min_t c^t}{Ld} \geq \gamma\left[\left(B + \frac{(B-1)BLQP}{\xi}\right) + B^4(1+B^3L^2QP)L^2\gamma^2 QP\right] = \bar{B}_1\gamma + \bar{C}_1\gamma^3$$

(A.126)
$$\bar{A}_2 = \frac{\min_t c^t}{d} > \gamma\left[((B-1)BLQP + LB) + B^4(1+B^3L^2QP)L^3\gamma^2 QP\right] = \bar{B}_2\gamma + \bar{C}_2\gamma^3,$$

with

$$\begin{aligned}
\bar{A}_1 &= \frac{\min_t c^t}{Ld} \\
\bar{B}_1 &= B + \frac{(B-1)BLQP}{\xi} \\
\bar{C}_1 &= B^4(1+B^3L^2QP)L^2QP \\
\bar{A}_2 &= \frac{\min_t c^t}{d} \\
\bar{B}_2 &= (B-1)BLQP + LB \\
\bar{C}_2 &= B^4(1+B^3L^2QP)L^3QP,
\end{aligned}$$

then the same constant learning rate γ is also valid for (A.123) and (A.124). Observing that the right-hand sides of both (A.125) and (A.126) have the same form, i.e. they are both cubic

equations with 0 being the only real root. Solving (A.125) and (A.126) show that

$$\gamma \in (0, \min\{\gamma_1, \gamma_2\}),$$

where

$$\gamma_1 = -2\sqrt{\frac{\bar{B}_1}{3\bar{C}_1}}\sinh\left(\frac{1}{3}\operatorname{arcsinh}\left(-\frac{3\bar{A}_1}{2\bar{B}_1}\sqrt{\frac{3\bar{C}_1}{\bar{B}_1}}\right)\right)$$
$$\gamma_2 = -2\sqrt{\frac{\bar{B}_2}{3\bar{C}_2}}\sinh\left(\frac{1}{3}\operatorname{arcsinh}\left(-\frac{3\bar{A}_2}{2\bar{B}_2}\sqrt{\frac{3\bar{C}_2}{\bar{B}_2}}\right)\right).$$

Combining (A.121), (A.122) with the above equation, finally, the constant learning rate is required to be

$$\gamma \in \left(0, \min\left\{1, \frac{1}{BLQP}, \gamma_1, \gamma_2\right\}\right).$$

□

APPENDIX B

Appendix

B.1. Extensions

We first introduce techniques to guarantee boundedness of the weight ω, i.e. how to remove condition 1 in Assumption 1. We then point out problems in the proofs of AMSGRAD (Reddi et al., 2018) and ADABOUND (Luo et al., 2019) and provide a different proof for AMSGRAD.

B.1.1. Unbounded Case

Projection is a popular technique to guarantee that a weight does not exceed a certain bound ((Blum, 1998), (Hazan et al., 2007), (Duchi et al., 2011), (Luo et al., 2019)). For unbounded weight $\hat{\omega}$, we introduce the following notation. Given convex sets $\mathscr{P}_1$, $\mathscr{P}_2$, vectors ω_1, ω_1', g_1 and matrix $\hat{v}$, we define projections

$$\Pi_{\mathscr{P}_1}(\hat{\omega}) = \operatorname*{argmin}_{\omega \in \mathscr{P}_1} \|\omega - \hat{\omega}\|$$

$$\Pi^1_{\mathscr{P}_1,\mathscr{P}_2,\omega_1,g_1,\omega_1'}(\hat{\omega}_2)$$

$$= \operatorname*{argmin}_{\omega_2' : \omega_2' \cdot \left[\left\|\omega_1' - \eta g_1\right\| / \sqrt{\frac{1}{2}+\xi_1}\right] \in \mathscr{P}_2} \left\| \omega_2' - \operatorname*{argmin}_{\omega_2 : \omega_1^T \omega_2 \in \mathscr{P}_1} \left\| \omega_1^T \omega_2 - \omega_1^T \hat{\omega}_2 \right\| \right\|$$

$$\Pi^2_{\mathscr{P}_1,\mathscr{P}_2,\omega_1,g_1,\omega_1',\hat{v}}(\hat{\omega}_2)$$

$$= \operatorname*{argmin}_{\omega_2' : \omega_2' \cdot \left[\left\|\omega_1' - \eta g_1\right\| / \sqrt{\frac{1}{2}+\xi_2}\right] \in \mathscr{P}_2} \left\| \omega_2' - \operatorname*{argmin}_{\omega_2 : \omega_1^T \omega_2 \in \mathscr{P}_1} \left\| \left(\sqrt[4]{\hat{v}} \odot \omega_2 \right)^T \omega_1 - \left(\sqrt[4]{\hat{v}} \odot \hat{\omega}_2 \right)^T \omega_1 \right\| \right\|.$$

Projection Π is the standard projection which maps vector $\hat{\omega}$ into set $\mathscr{P}_1$. If an optimal weight ω_* is such that $\omega_* \in \mathscr{P}_1$, then we have

$$\left\|\Pi_{\mathscr{P}_1}(\hat{\omega}_{t+1}) - \omega_*\right\| \leq \left\|\hat{\omega}_{t+1} - \omega_*\right\|,$$

which could be directly applied in the proofs of Theorem 7 and 8.

For Π^1 and Π^2, we could regard them as a combination of two standard projections. Note that, for the outer projection, we require that it does not affect the product of $\omega_1^T\omega_2$, which could be done by projection methods for linear equality constraints. In this way, we have

$$\begin{aligned}
&\left\|\omega_{1,t+1}^T\Pi^1_{\mathscr{P}_1,\mathscr{P}_2,\omega_{1,t+1},g_{1,t},\omega_{1,t}}(\hat{\omega}_{2,t+1}) - \omega_{1,*}^T\omega_{2,*}\right\| \leq \left\|\omega_{1,t+1}^T\hat{\omega}_{2,t+1} - \omega_{1,*}^T\omega_{2,*}\right\| \\
&\left\|\left(\sqrt[4]{\hat{v}_{2,t}} \odot \Pi^2_{\mathscr{P}_1,\mathscr{P}_2,\omega_{1,t+1},g_{1,t},\omega_{1,t},\hat{v}_{2,t}}(\hat{\omega}_{2,t+1})\right)^T\omega_{1,t+1} - \left(\sqrt[4]{\hat{v}_{2,t}} \odot \omega_{2,*}\right)^T\omega_{1,*}\right\| \\
&\quad\leq \left\|\left(\sqrt[4]{\hat{v}_{2,t}} \odot (\hat{\omega}_{2,t+1})\right)^T\omega_{1,t+1} - \left(\sqrt[4]{\hat{v}_{2,t}} \odot \omega_{2,*}\right)^T\omega_{1,*}\right\|,
\end{aligned}$$

which could also be directly applied in the proofs of Theorem 9 and 10.

B.1.2. Standard setting of ADAM

First, let us point out the problem in AMSGRAD (Reddi et al., 2018). At the bottom of Page 18 in (Reddi et al., 2018), the authors obtain an upper bound for the regret which has a term containing $\sum_{t=1}^T \frac{\beta_{1t}\hat{v}_{t,i}^{1/2}}{\alpha_t}$. Without assuming that β_{1t} is exponentially decaying, it is questionable to establish $\mathcal{O}(\sqrt{T})$ given $\alpha_t = \frac{1}{\sqrt{t}}$ since $\sum_{t=1}^T \sqrt{t} > \mathcal{O}(\sqrt{T})$. Although this questionable term can be bounded by assumptions on β_{1t}, the last term in Theorem 4 is $\mathcal{O}(\log(T)\sum_{i=1}^d \left\|g_{1:T,i}\right\|_2) = \mathcal{O}(\log(T)\sqrt{T})$ since $g_{1:T,i}$ is the concatenation of the gradients from 0 to current time T in the

i^{th} coordinate. Moreover, the authors argue that decaying β_{1t} is crucial to guarantee the convergence, however, our proof shows $\mathcal{O}(\sqrt{T})$ regret for AMSGRAD with constant β and both constant and diminishing stepsizes, which is more practically relevant. For a diminishing stepsize, the slight change we need to make in the proof is that η_t needs to be considered together with $\sqrt{\hat{v}_{t,j}}$ in (B.11) and the rest of proof of Theorem 8. Applying the fact that $\frac{\sqrt{\hat{v}_{t,j}}}{\eta_t} \geq \frac{\sqrt{\hat{v}_{t-1,j}}}{\eta_t}$ and $\sum_{t=1}^{T} \frac{1}{\sqrt{t}} = 2\sqrt{T} - 1$ yields $\mathcal{O}(\sqrt{T})$ regret in standard online setting.

Table B.1 summarizes the various regret bounds in different convex settings.

	gradient descent		Adam	
	constant	diminishing	constant	diminishing
standard online	$\mathcal{O}(\sqrt{T})$(us) $\mathcal{O}(T)$(Zinkevich, 2003)	$\mathcal{O}(\sqrt{T})$(us) $\mathcal{O}(\sqrt{T})$(Zinkevich, 2003)	$\mathcal{O}(\sqrt{T})$(us)	$\mathcal{O}(\sqrt{T})$(us) $\mathcal{O}(\sqrt{T})$(Reddi et al., 2018) (flawed) $\mathcal{O}(\log(T)\sqrt{T})$(Reddi et al., 2018) (true)
streaming	$\mathcal{O}(\sqrt{T})$(us)	$\mathcal{O}(T)$(us)	$\mathcal{O}(\sqrt{T})$(us)	$\mathcal{O}(T)$(us)

Table B.1. Summary of known regret bounds for online learning and streaming in convex setting

B.2. Regret with Rolling Window Analysis of OGD

Proof of Theorem 7.

PROOF. For any $p \in \mathbb{N}$ and fixed T, from step 4 in Algorithm 2, for any ω^*, we obtain

$$\|\omega_{t+1} - \omega^*\|^2 = \|\omega_t - \eta \bigtriangledown f_t(\omega_t) - \omega^*\|^2$$

$$= \|\omega_t - \omega^*\|^2 - 2\eta \langle \omega_t - \omega^*, \bigtriangledown f_t(\omega_t) \rangle + \eta^2 \|\bigtriangledown f_t(\omega_t)\|^2,$$

which in turn yields

$$\langle \omega_t - \omega^*, \bigtriangledown f_t(\omega_t) \rangle = \frac{\|\omega_t - \omega^*\|^2 - \|\omega_{t+1} - \omega^*\|^2}{2\eta} + \frac{\eta}{2} \|\bigtriangledown f_t(\omega_t)\|^2 . \tag{B.1}$$

Applying convexity of f_t yields

$$f_t(\omega_t) - f_t(\omega^*) \leq \langle \omega_t - \omega^*, \bigtriangledown f_t(\omega_t) \rangle . \tag{B.2}$$

Inserting (B.1) into (B.2) gives

$$f_t(\omega_t) - f_t(\omega^*) \leq \frac{\|\omega_t - \omega^*\|^2 - \|\omega_{t+1} - \omega^*\|^2}{2\eta} + \frac{\eta}{2} \|\bigtriangledown f_t(\omega_t)\|^2 .$$

By summing up all differences, we obtain

$$\begin{aligned}
\sum_{t=p}^{T+p} [f_t(\omega_t) - f_t(\omega^*)] &\leq \frac{1}{2} \sum_{t=p}^{T+p} \left[\frac{\|\omega_t - \omega^*\|^2 - \|\omega_{t+1} - \omega^*\|^2}{\eta} + \eta \|\bigtriangledown f_t(\omega_t)\|^2 \right] \\
&\leq \frac{1}{2} \left(\frac{\|\omega_p - \omega^*\|^2}{\eta} \right) + dG_\infty \sum_{t=p}^{T+p} \eta \\
&\leq \frac{D_\infty^2 \sqrt{T}}{2\eta_1} + dG_\infty \eta_1 \sqrt{T} = \mathcal{O}(\sqrt{T}). \qquad \text{(B.3)}
\end{aligned}$$

The second inequality holds due to 2 in Assumption 1 and the last inequality uses 4 in Assumption 1 and the definition of η. Since (B.3) holds for any p and ω^*, setting $\omega^* = \omega_p^*$ for each p yields the statement in Theorem 7. □

B.3. Regret with Rolling Window Analyses of CONVGADAM

Lemma 3. *Under the conditions assumed in Theorem 8, we have*

$$\sum_{t=p}^{T+p} \left\| \frac{1}{\sqrt[4]{\hat{v}_t}} \odot m_t \right\|^2 \leq \mathcal{O}(T).$$

PROOF. By the definition of $\hat{v}_t$, for any $t = p, p+1, \cdots, T+p$, we obtain

$$\begin{aligned}
\left\| \frac{1}{\sqrt[4]{\hat{v}_t}} \odot m_t \right\|^2 &= \sum_{j=1}^{d} \frac{m_{t,j}^2}{\sqrt{\hat{v}_{t,j}}} \leq \sum_{j=1}^{d} \frac{m_{t,j}^2}{\sqrt{v_{t,j}}} = \sum_{j=1}^{d} \frac{\left((1-\beta_1)\sum_{i=1}^{t} \beta_1^{t-i} g_{i,j}\right)^2}{\sqrt{(1-\beta_2)\sum_{i=1}^{t} \beta_2^{t-i} g_{i,j}^2}} \\
&\leq \frac{(1-\beta_1)^2}{\sqrt{1-\beta_2}} \sum_{j=1}^{d} \frac{\left(\sum_{i=1}^{t} \beta_1^{t-i}\right)\left(\sum_{i=1}^{t} \beta_1^{t-i} g_{i,j}^2\right)}{\sqrt{\sum_{i=1}^{t} \beta_2^{t-i} g_{i,j}^2}} \\
&\leq \frac{1-\beta_1}{\sqrt{1-\beta_2}} \sum_{j=1}^{d} \frac{\sum_{i=1}^{t} \beta_1^{t-i} g_{i,j}^2}{\sqrt{\sum_{i=1}^{t} \beta_2^{t-i} g_{i,j}^2}} \leq \frac{1-\beta_1}{\sqrt{1-\beta_2}} \sum_{j=1}^{d} \sum_{i=1}^{t} \left(\frac{\beta_1}{\sqrt{\beta_2}}\right)^{t-i} \|g_{i,j}\|_2 \\
&= \frac{1-\beta_1}{\sqrt{1-\beta_2}} \sum_{j=1}^{d} \sum_{i=1}^{t} \lambda^{t-i} \|g_{i,j}\|_2 . \qquad \text{(B.4)}
\end{aligned}$$

The second equality follows from the updating rule of Algorithm 3. The second inequality follows from the Cauchy-Schwarz inequality, while the third inequality follows from the inequality

$\sum_{i=1}^{t} \beta_1^{t-i} \leq \frac{1}{1-\beta_1}$. Using (B.4) for all time steps yields

$$
\begin{aligned}
&\sum_{t=p}^{T+p} \frac{1}{\sqrt{\hat{v}_t}} \odot (m_t \odot m_t) \\
\leq &\frac{1-\beta_1}{\sqrt{1-\beta_2}} \sum_{t=p}^{T+p} \sum_{j=1}^{d} \sum_{i=1}^{t} \lambda^{t-i} \left\| g_{i,j} \right\|_2 \\
= &\frac{1-\beta_1}{\sqrt{1-\beta_2}} \sum_{j=1}^{d} \sum_{t=p}^{T+p} \left(\sum_{i=p+1}^{t} \lambda^{t-i} \left\| g_{i,j} \right\|_2 + \sum_{i=1}^{p} \lambda^{t-i} \left\| g_{i,j} \right\|_2 \right) \\
= &\frac{1-\beta_1}{\sqrt{1-\beta_2}} \sum_{j=1}^{d} \left(\sum_{t=p+1}^{T+p} \sum_{i=p+1}^{t} \lambda^{t-i} \left\| g_{i,j} \right\|_2 + \sum_{t=p}^{T+p} \sum_{i=1}^{p} \lambda^{t-i} \left\| g_{i,j} \right\|_2 \right) \\
= &\frac{1-\beta_1}{\sqrt{1-\beta_2}} \sum_{j=1}^{d} \left(\sum_{t=p+1}^{T+p} \sum_{i=p+1}^{t} \lambda^{t-i} \left\| g_{i,j} \right\|_2 + \left(\sum_{i=1}^{p} \lambda^{p-i} \left\| g_{i,j} \right\|_2 \right) \left(\sum_{i=0}^{T} \lambda^{i} \right) \right). \qquad \text{(B.5)}
\end{aligned}
$$

We first bound the first term in (B.5) for each j as follows,

$$
\begin{aligned}
&\sum_{t=p+1}^{T+p} \sum_{i=p+1}^{t} \lambda^{t-i} \left\| g_{i,j} \right\|_2 = \sum_{i=p+1}^{T+p} \left\| g_{i,j} \right\|_2 \sum_{t=i}^{T+p} \lambda^{T+p-t} \\
\leq &\frac{1}{1-\lambda} \sum_{t=p+1}^{T+p} \left\| g_{i,j} \right\|_2 \leq \frac{T G_\infty}{1-\lambda}. \qquad \text{(B.6)}
\end{aligned}
$$

The first inequality follows from the fact that $\sum_{t=i}^{T+p} \lambda^{T+p-t} < \frac{1}{1-\lambda}$ and the last inequality is due to 2 in Assumption 1. Using a similar argument, we further bound the second term in (B.5) as follows,

$$
\begin{aligned}
&\left(\sum_{i=1}^{p} \lambda^{p-i} \left\| g_{i,j} \right\|_2 \right) \left(\sum_{i=0}^{T} \lambda^{i} \right) \leq \frac{1}{1-\lambda} \left(\sum_{i=1}^{p} \lambda^{p-i} \left\| g_{i,j} \right\|_2 \right) \\
\leq &\frac{G_\infty}{1-\lambda} \left(\sum_{i=1}^{p} \lambda^{p-i} \right) \leq \frac{G_\infty}{(1-\lambda)^2}. \qquad \text{(B.7)}
\end{aligned}
$$

Inserting (B.6) and (B.7) into (B.5) implies

$$\sum_{t=p}^{T+p}\left\|\frac{1}{\sqrt[4]{\hat{v}_t}}\odot m_t\right\|^2 \leq \frac{d\left(1-\beta_1\right)}{\sqrt{1-\beta_2}}\left(\frac{TG_\infty}{1-\lambda}+\frac{G_\infty}{\left(1-\lambda\right)^2}\right).$$

This completes the proof of the lemma. □

In order to establish the regret analysis of Algorithm 3, we further need the following intermediate result.

Lemma 4. *Under the conditions in Theorem 8, we have*

$$\sum_{t=p}^{T+p}\left\|m_{t-1}\right\|^2 \leq \mathcal{O}(T).$$

PROOF. By the definition of m_t, we obtain

$$\begin{aligned}
&\sum_{t=p}^{T+p}\left\|m_{t-1}\right\|^2 = \sum_{t=p}^{T+p}\sum_{j=1}^{d} m_{t-1,j}^2 \\
=&\sum_{t=p}^{T+p}\sum_{j=1}^{d}\left((1-\beta_1)\sum_{i=1}^{t}\beta_1^{t-i}g_{i,j}\right)^2 \\
\leq&(1-\beta_1)^2\sum_{t=p}^{T+p}\sum_{j=1}^{d}\left(\sum_{i=1}^{t}\beta_1^{t-i}\right)\left(\sum_{i=1}^{t}\beta_1^{t-i}g_{i,j}^2\right) \\
\leq&(1-\beta_1)\sum_{t=p}^{T+p}\sum_{j=1}^{d}\left(\sum_{i=1}^{t}\beta_1^{t-i}g_{i,j}^2\right) \leq (1-\beta_1)\sum_{t=p}^{T+p}\sum_{j=1}^{d}\left(G_\infty\sum_{i=1}^{t}\beta_1^{t-i}\right) \\
\leq&\sum_{t=p}^{T+p}\sum_{j=1}^{d}G_\infty = dTG_\infty.
\end{aligned}$$

The first inequality follows from the Cauchy-Schwarz inequality. The second and the last inequalities use the fact that $\sum_{i=1}^{t} \beta_1^{t-i} \leq \frac{1}{1-\beta_1}$. The third inequality is due to 2 in Assumption 1. This completes the proof of the lemma. □

Proof of Theorem 8.

PROOF. Based on the update step 8 in Algorithm 3 and given any $\omega^* \in \mathbb{R}^d$, we obtain

$$\begin{aligned}
\|\omega_{t+1} - \omega^*\|^2 &= \left\|\omega_t - \frac{\eta}{\sqrt{\hat{v}_t}} \odot m_t - \omega^*\right\|^2 \\
&= \|\omega_t - \omega^*\|^2 - 2\left\langle \omega_t - \omega^*, \frac{\eta}{\sqrt{\hat{v}_t}} \odot m_t \right\rangle + \left\| \frac{\eta}{\sqrt{\hat{v}_t}} \odot m_t \right\|^2 \\
&= \|\omega_t - \omega^*\|^2 - 2\left\langle \omega_t - \omega^*, \frac{\eta(1-\beta_1)}{\sqrt{\hat{v}_t}} \odot g_t \right\rangle - 2\left\langle \omega_t - \omega^*, \frac{\eta\beta_1}{\sqrt{\hat{v}_t}} \odot m_{t-1} \right\rangle \\
&\qquad + \left\| \frac{\eta}{\sqrt{\hat{v}_t} \odot m_t} \right\|^2 . \qquad \text{(B.8)}
\end{aligned}$$

The first inequality uses the same argument as those used in Theorem 7. Rearranging (B.8) gives

$$\begin{aligned}
\langle \omega_t - \omega^*, g_t \rangle &= \frac{\left[\left\| \sqrt[4]{\hat{v}_t} \odot (\omega_t - \omega^*) \right\|^2 - \left\| \sqrt[4]{\hat{v}_t} \odot (\omega_{t+1} - \omega^*) \right\|^2\right]}{2\eta(1-\beta_1)} \\
&\quad - \frac{\beta_1}{1-\beta_1} \left\langle \frac{\omega_t - \omega^*}{\sqrt{\eta}}, m_{t-1}\sqrt{\eta} \right\rangle + \frac{1}{2\eta(1-\beta_1)} \left\| \frac{\eta}{\sqrt[4]{\hat{v}_t}} \odot m_t \right\|^2 \\
&\leq \frac{\left[\left\| \sqrt[4]{\hat{v}_t} \odot (\omega_t - \omega^*) \right\|^2 - \left\| \sqrt[4]{\hat{v}_t} \odot (\omega_{t+1} - \omega^*) \right\|^2\right]}{2\eta(1-\beta_1)} \\
&\quad + \frac{\beta_1}{1-\beta_1} \left[\frac{\|\omega_t - \omega^*\|^2}{2\eta} + \frac{m_{t-1} \odot m_{t-1}\eta}{2} \right] + \frac{\eta}{2(1-\beta_1)} \left\| \frac{1}{\sqrt[4]{\hat{v}_t}} \odot m_t \right\|^2 . \qquad \text{(B.9)}
\end{aligned}$$

From the strong convexity property of f_t in 4 in Assumption 1, we obtain

$$f_t(\omega_t) - f_t(\omega^*) \leq \langle \omega_t - \omega^*, \bigtriangledown f_t(\omega_t) \rangle - \frac{H}{2} \left\| \omega_t - \omega^* \right\|^2.$$

Using (B.9) in the above inequality and summing up over all time steps yields

$$\begin{aligned}
&\sum_{t=p}^{T+p} \left[f_t(\omega_t) - f_t(\omega^*) \right] \\
&\leq \sum_{t=p}^{T+p} \left\{ \frac{\left[\left\| \sqrt[4]{\hat{v}_t} \odot (\omega_t - \omega^*) \right\|^2 - \left\| \sqrt[4]{\hat{v}_t} \odot (\omega_{t+1} - \omega^*) \right\|^2 \right]}{2\eta(1-\beta_1)} + \left\| \omega_t - \omega^* \right\|^2 \left[\frac{\beta_1}{2\eta(1-\beta_1)} - \frac{H}{2} \right] \right. \\
&\qquad \left. + \frac{\eta}{2(1-\beta_1)} \left[\beta_1 m_{t-1} \odot m_{t-1} + \left\| \frac{1}{\sqrt[4]{\hat{v}_t}} \odot m_t \right\|^2 \right] \right\}. \qquad \text{(B.10)}
\end{aligned}$$

We proceed by separating (B.10) into 3 parts and find upper bounds for each one of them. Considering the first part in (B.10), we have

$$\begin{aligned}
&\sum_{t=p}^{T+p} \frac{\left[\left\|\sqrt[4]{\hat{v}_t}\odot(\omega_t-\omega^*)\right\|^2-\left\|\sqrt[4]{\hat{v}_t}\odot(\omega_{t+1}-\omega^*)\right\|^2\right]}{2\eta(1-\beta_1)}\\
\leq&\frac{\left\|\sqrt[4]{\hat{v}_p}\odot(\omega_p-\omega^*)\right\|^2}{2\eta(1-\beta_1)}+\frac{1}{2\eta(1-\beta_1)}\sum_{t=p+1}^{T+p}\left(\left\|\sqrt[4]{\hat{v}_t}\odot(\omega_t-\omega^*)\right\|^2\right.\\
&\qquad\left.-\left\|\sqrt[4]{\hat{v}_{t-1}}\odot(\omega_t-\omega^*)\right\|^2\right)\\
=&\frac{1}{2\eta(1-\beta_1)}\left[\left\|\sqrt[4]{\hat{v}_p}\odot(\omega_p-\omega^*)\right\|^2+\sum_{t=p+1}^{T+p}\left(\sum_{j=1}^{d}\sqrt{\hat{v}_{t,j}}(\omega_{t,j}-\omega^{*,j})^2\right.\right.\\
&\qquad\left.\left.-\sum_{j=1}^{d}\sqrt{\hat{v}_{t-1,j}}(\omega_{t,j}-\omega^{*,j})^2\right)\right]
\end{aligned}$$

(B.11)

$$=\frac{1}{2\eta(1-\beta_1)}\left[\left\|\sqrt[4]{\hat{v}_p}\odot(\omega_p-\omega^*)\right\|^2+\sum_{t=p+1}^{T+p}\left(\sum_{j=1}^{d}(\omega_{t,j}-\omega^{*,j})^2\left(\sqrt{\hat{v}_{t,j}}-\sqrt{\hat{v}_{t-1,j}}\right)\right)\right].$$

Since $\hat{v}_{t,j}$ is maximum of all $v_{t,j}$ for each j until the current time step, i.e. $\sqrt{\hat{v}_{t,j}} - \sqrt{\hat{v}_{t-1,j}} \geq 0$, by using 1 in Assumption 1, (B.11) can be further bounded as follows,

$$\begin{aligned}
&\sum_{t=p}^{T+p} \frac{\left[\left\| \sqrt[4]{\hat{v}_t} \odot (\omega_t - \omega^*) \right\|^2 - \left\| \sqrt[4]{\hat{v}_t} \odot (\omega_{t+1} - \omega^*) \right\|^2\right]}{2\eta(1-\beta_1)} \\
\leq &\frac{1}{2\eta(1-\beta_1)} \left[\left\| \sqrt[4]{\hat{v}_p} \odot (\omega_p - \omega^*) \right\|^2 + D_\infty^2 \sum_{j=1}^{d} \sum_{t=p+1}^{T+p} \left(\sqrt{\hat{v}_{t,j}} - \sqrt{\hat{v}_{t-1,j}} \right) \right] \\
\leq &\frac{1}{2\eta(1-\beta_1)} \left[D_\infty^2 \sum_{j=1}^{d} \sqrt{\hat{v}_{p,j}} + D_\infty^2 \sum_{j=1}^{d} \sum_{t=p+1}^{T+p} \left(\sqrt{\hat{v}_{t,j}} - \sqrt{\hat{v}_{t-1,j}} \right) \right] \\
= &\frac{1}{2\eta(1-\beta_1)} D_\infty^2 \sum_{j=1}^{d} \sqrt{\hat{v}_{p+T,j}}.
\end{aligned}$$

By the definition of $\hat{v}_t$ in step 6 in Algorithm 3, for any t and j, we have

$$v_{t,j} = (1-\beta_2) \sum_{i=1}^{t} \beta_2^{t-i} g_{i,j}^2 \leq (1-\beta_2) G_\infty^2 \sum_{i=1}^{t} \beta_2^{t-i} \leq G_\infty^2,$$

which in turn yields

$$\sum_{t=p}^{T+p} \frac{\left[\left\| \sqrt[4]{\hat{v}_t} \odot (\omega_t - \omega^*) \right\|^2 - \left\| \sqrt[4]{\hat{v}_t} \odot (\omega_{t+1} - \omega^*) \right\|^2\right]}{2\eta(1-\beta_1)} \leq \frac{d D_\infty^2 G_\infty}{2\eta(1-\beta_1)} = \mathcal{O}(\sqrt{T}). \tag{B.12}$$

The last equality is due to the setting of the stepsize, i.e. $\eta = \frac{\eta_1}{\sqrt{T}}$. For the second term in (B.10), from the relationship between β_1 and H, we obtain

$$\frac{\beta_1}{1-\beta_1} \leq H\eta,$$

which in turn yields

$$\frac{\beta_1}{2\eta(1-\beta_1)} - \frac{H}{2} \leq 0. \tag{B.13}$$

Thus, (B.13) guarantees negativity of the second term in (B.10). For the third term in (B.10), by using Lemmas 3 and 4, we assert

$$\frac{\eta}{2(1-\beta_1)} \left[\beta_1 m_{t-1} \odot m_{t-1} + \left\| \frac{1}{\sqrt[4]{\hat{v}_t}} \odot m_t \right\|^2 \right] \leq \mathcal{O}(\frac{1}{\sqrt{T}}) \cdot \mathcal{O}(T) = \mathcal{O}(\sqrt{T}). \tag{B.14}$$

The desired result follows directly from (B.10), (B.12), (B.13) and (B.14). □

B.4. Regret with Rolling Window Analysis of dnnOGD for Two-Layer ReLU Neural Network

For a two-layer ReLU neural network, we first introduce $\mathcal{F}^t$ that records all previous iterates up until t.

Lemma 5. *If conditions 1 and 2 hold from Assumption 2, we have*

$$\mathbb{E}\left[l_t\left(\omega_{1,t}, \omega_{2,t}\right) \mid \mathcal{F}^t \right] = \frac{\rho^2}{2} (\omega_{1,t}^T \omega_{2,t} z^t - \omega_{1,*}^T \omega_{2,*} z^t)^2. \tag{B.15}$$

PROOF. Based on condition 2 in Assumption 2, we obtain

$$\begin{aligned}
&\mathbb{E}_{\sigma_1,\sigma_2}\left[f_t(\omega_{1,t},\omega_{2,t}) \mid \mathcal{F}^t\right] \\
=&\frac{1}{2}\,\mathbb{E}_{\sigma_1}\left[\omega_{1,t}^T\sigma_1\left(\omega_{2,t}z^t\right)-y^t \mid \mathcal{F}^t\right]\cdot\mathbb{E}_{\sigma_2}\left[\omega_{1,t}^T\sigma_2\left(\omega_{2,t}z^t\right)-y^t \mid \mathcal{F}^t\right] \\
=&\frac{1}{2}\left(\rho\omega_{1,t}^T\omega_{2,t}z^t-y^t\right)\cdot\left(\rho\omega_{1,t}^T\omega_{2,t}z^t-y^t\right)=\frac{1}{2}\left(\rho\omega_{1,t}^T\omega_{2,t}z^t-y^t\right)^2 \\
=&\frac{\rho^2}{2}(\omega_{1,t}^T\omega_{2,t}z^t-\omega_{1,*}^T\omega_{2,*}z^t)^2.
\end{aligned}$$

On the other hand, we get

$$\begin{aligned}
&\mathbb{E}_{\sigma_1,\sigma_2}\left[f_t(\omega_{1,*},\omega_{2,*}) \mid \mathcal{F}^t\right] \\
=&\frac{1}{2}\,\mathbb{E}_{\sigma_1}\left[\omega_{1,*}^T\sigma_1\left(\omega_{2,*}z^t\right)-y^t \mid \mathcal{F}^t\right]\cdot\mathbb{E}_{\sigma_2}\left[\omega_{1,*}^T\sigma_2\left(\omega_{2,*}z^t\right)-y^t \mid \mathcal{F}^t\right] \\
=&\frac{1}{2}\left(\mathbb{E}_{\sigma_1}\left[\omega_{1,*}^T\sigma_1\left(\omega_{2,*}z^t\right) \mid \mathcal{F}^t\right]-y^t\right)\cdot\left(\mathbb{E}_{\sigma_2}\left[\omega_{1,*}^T\sigma_2\left(\omega_{2,*}z^t\right) \mid \mathcal{F}^t\right]-y^t\right) \\
=&0,
\end{aligned}$$

which in turn yields

$$\begin{aligned}
&\mathbb{E}\left[l_t\left(\omega_{1,t},\omega_{2,t}\right) \mid \mathcal{F}^t\right] \\
=&\,\mathbb{E}_{\sigma_1,\sigma_2}\left[f_t(\omega_{1,t},\omega_{2,t}) \mid \mathcal{F}^t\right]-\mathbb{E}_{\sigma_1,\sigma_2}\left[f_t(\omega_{1,*},\omega_{2,*}) \mid \mathcal{F}^t\right] \\
=&\frac{\rho^2}{2}(\omega_{1,t}^T\omega_{2,t}z^t-\omega_{1,*}^T\omega_{2,*}z^t)^2.
\end{aligned}$$

This completes the proof of the lemma. □

Lemma 6. *Under the conditions assumed in Theorem 9, we have*

$$\mathbb{E}_{\sigma_1,\sigma_2}\left[g_{1,t} \mid \mathcal{F}^t\right] = \rho^2\left(\omega_{1,t}^T\omega_{2,t}z^t - \omega_{1,*}^T\omega_{2,*}z^t\right)\omega_{2,t}z^t \tag{B.16}$$

$$\mathbb{E}_{\sigma_1,\sigma_2}\left[g_{2,t} \mid \mathcal{F}^t\right] = \rho^2\left(\omega_{1,t}^T\omega_{2,t}z^t - \omega_{1,*}^T\omega_{2,*}z^t\right)\omega_{1,t}\left(z^t\right)^T. \tag{B.17}$$

PROOF. From steps 4 and 5, we have

$$\begin{aligned}
&\mathbb{E}_{\sigma_1,\sigma_2}\left[g_{1,t} \mid \mathcal{F}^t\right] \\
=&\mathbb{E}_{\sigma_1,\sigma_2}\left[\bigtriangledown_{\omega_1}\left(\frac{1}{2}\left(\omega_{1,t}^T\sigma_1\left(\omega_{2,t}z^t\right) - y^t\right)\left(\omega_{1,t}^T\sigma_2\left(\omega_{2,t}z^t\right) - y^t\right)\right) \mid \mathcal{F}^t\right] \\
=&\mathbb{E}_{\sigma_1,\sigma_2}\left[\left(\omega_{1,t}^T\sigma_1\left(\omega_{2,t}z^t\right) - y^t\right)\sigma_2\left(\omega_{2,t}z^t\right) \mid \mathcal{F}^t\right] \\
=&\mathbb{E}_{\sigma_1}\left[\omega_{1,t}^T\sigma_1\left(\omega_{2,t}z^t\right) - \omega_{1,*}^T\sigma_1\left(\omega_{2,*}z^t\right) \mid \mathcal{F}^t\right]\mathbb{E}_{\sigma_2}\left[\sigma_2\left(\omega_{2,t}z^t\right) \mid \mathcal{F}^t\right] \\
=&\rho\left(\omega_{1,t}^T\omega_{2,t}z^t - \omega_{1,*}^T\omega_{2,*}z^t\right)\rho\omega_{2,t}z^t = \rho^2\left(\omega_{1,t}^T\omega_{2,t}z^t - \omega_{1,*}^T\omega_{2,*}z^t\right)\omega_{2,t}z^t.
\end{aligned}$$

Similarly,

$$\begin{aligned}
&\mathbb{E}_{\sigma_1,\sigma_2}\left[g_{2t} \mid \mathcal{F}^t\right] \\
=&\mathbb{E}_{\sigma_1,\sigma_2}\left[\bigtriangledown_{\omega_2}\left(\frac{1}{2}\left(\omega_{1,t}^T\sigma_1\left(\omega_{2,t}z^t\right) - y^t\right)\left(\omega_{1,t}^T\sigma_2\left(\omega_{2,t}z^t\right) - y^t\right)\right) \mid \mathcal{F}^t\right] \\
=&\mathbb{E}_{\sigma_1,\sigma_2}\left[\left(\omega_{1,t}^T\sigma_1\left(\omega_{2,t}z^t\right) - y^t\right)\omega_{1,t}\left(\sigma_2(z^t)\right)^T \mid \mathcal{F}^t\right] \\
=&\mathbb{E}_{\sigma_1}\left[\omega_{1,t}^T\sigma_1\left(\omega_{2,t}z^t\right) - \omega_{1,*}^T\sigma_1\left(\omega_{2,*}z^t\right) \mid \mathcal{F}^t\right]\mathbb{E}_{\sigma_2}\left[\omega_{1,t}\left(\sigma_2(z^t)\right)^T \mid \mathcal{F}^t\right] \\
=&\rho\left(\omega_{1,t}^T\omega_{2,t}z^t - \omega_{1,*}^T\omega_{2,*}z^t\right)\rho\omega_{1,t}\left(z^t\right)^T \\
=&\rho^2\left(\omega_{1,t}^T\omega_{2,t}z^t - \omega_{1,*}^T\omega_{2,*}z^t\right)\omega_{1,t}\left(z^t\right)^T.
\end{aligned}$$

This completes the proof of the lemma. □

B.5. Proof of Theorem 9

PROOF. First, based on the update step 6 and 7 in Algorithm 4, we obtain

$$
\begin{aligned}
&\mathbb{E}_{\sigma_1,\sigma_2}\left[\left\|\omega_{1,t+1}^T\omega_{2,t+1}-\omega_{1,*}^T\omega_{2,*}\right\|^2 \mid \mathcal{F}^t\right]=\mathbb{E}_{\sigma_1,\sigma_2}\left[\left\|\omega_{2,t+1}^T\omega_{1,t+1}-\omega_{2,*}^T\omega_{1,*}\right\|^2 \mid \mathcal{F}^t\right]\\
=&\,\mathbb{E}_{\sigma_1,\sigma_2}\left[\left\|\left(\omega_{2,t}-\eta g_{2,t}\right)^T\left(\omega_{1,t}-\eta g_{1,t}\right)-\omega_{2,*}^T\omega_{1,*}\right\|^2 \mid \mathcal{F}^t\right]\\
=&\,\mathbb{E}_{\sigma_1,\sigma_2}\left[\left\|\omega_{2,t}^T\omega_{1,t}-\omega_{2,*}^T\omega_{1,*}-\eta\left(g_{2,t}^T\omega_{1,t}+\omega_{2,t}^Tg_{1,t}\right)+\eta^2g_{1,t}^Tg_{2,t}\right\|^2 \mid \mathcal{F}^t\right]\\
=&\,\mathbb{E}_{\sigma_1,\sigma_2}\left[\left\|\omega_{2,t}^T\omega_{1,t}-\omega_{2,*}^T\omega_{1,*}\right\|^2-2\eta\left\langle\omega_{2,t}^T\omega_{1,t}-\omega_{2,*}^T\omega_{1,*},g_{2,t}^T\omega_{1,t}+\omega_{2,t}^Tg_{1,t}\right\rangle\right.\\
&\qquad\left.+\eta^2\left(2\left\langle\omega_{2,t}^T\omega_{1,t}-\omega_{2,*}^T\omega_{1,*},g_{2,t}^Tg_{1,t}\right\rangle+\left\|\eta g_{1,t}^Tg_{2,t}-\left(g_{2,t}^T\omega_{1,t}+\omega_{2,t}^Tg_{1,t}\right)\right\|^2\right) \mid \mathcal{F}^t\right]\\
=&\left\|\omega_{2,t}^T\omega_{1,t}-\omega_{2,*}^T\omega_{1,*}\right\|^2-2\eta\left\langle\omega_{2,t}^T\omega_{1,t}-\omega_{2,*}^T\omega_{1,*},\mathbb{E}_{\sigma_1,\sigma_2}\left[g_{2,t}^T \mid \mathcal{F}^t\right]\omega_{1,t}\right.\\
&\qquad\left.+\omega_{2,t}^T\,\mathbb{E}_{\sigma_1,\sigma_2}\left[g_{1,t} \mid \mathcal{F}^t\right]\right\rangle+\eta^2\,\mathbb{E}_{\sigma_1,\sigma_2}\left[2\left\langle\omega_{2,t}^T\omega_{1,t}-\omega_{2,*}^T\omega_{1,*},g_{2,t}^Tg_{1,t}\right\rangle\right.\\
&\qquad\left.+\left\|\eta g_{1,t}^Tg_{2,t}-\left(g_{2,t}^T\omega_{1,t}+\omega_{2,t}^Tg_{1,t}\right)\right\|^2 \mid \mathcal{F}^t\right].
\end{aligned}
\tag{B.18}
$$

By Lemma 6 we conclude that $\mathbb{E}\left[\|g_{1,t}\| \mid \mathcal{F}^t\right]$ and $\mathbb{E}\left[\|g_{2,t}\| \mid \mathcal{F}^t\right]$ are bounded due to 3 in Assumption 2, which in turn yields

$$\begin{aligned}
&\mathbb{E}_{\sigma_1,\sigma_2}\left[2\left\langle \omega_{2,t}^T\omega_{1,t} - \omega_{2,*}^T\omega_{1,*}, g_{2,t}^T g_{1,t}\right\rangle + \left\|\eta g_{1,t}^T g_{2,t} - \left(g_{2,t}^T\omega_{1,t} + \omega_{2,t}^T g_{1,t}\right)\right\|^2 \mid \mathcal{F}^t\right] \\
&\quad \leq \left\|\omega_{2,t}^T\omega_{1,t} - \omega_{2,*}^T\omega_{1,*}\right\|^2 \cdot \mathbb{E}_{\sigma_1,\sigma_2}\left[\left\|g_{2,t}^T g_{1,t}\right\|^2 \mid \mathcal{F}^t\right] \\
&\qquad\qquad + \mathbb{E}_{\sigma_1,\sigma_2}\left[\left\|\eta g_{1,t}^T g_{2,t} - \left(g_{2,t}^T\omega_{1,t} + \omega_{2,t}^T g_{1,t}\right)\right\|^2 \mid \mathcal{F}^t\right] \\
&\quad \leq M_1, \qquad \text{(B.19)}
\end{aligned}$$

where M_1 is a fixed positive number. The first inequality comes from the Cauchy-Schwarz inequality and the second inequality is due to the boundedness of $\omega_{1,t}$, $\omega_{2,t}$, $\omega_{1,*}$, $\omega_{2,*}$, $g_{1,t}$, $g_{2,t}$ and η. Inserting (B.19) into (B.18) gives

$$\begin{aligned}
&\left\langle \omega_{2,t}^T\omega_{1,t} - \omega_{2,*}^T\omega_{1,*}, \mathbb{E}_{\sigma_1,\sigma_2}\left[g_{2,t}^T \mid \mathcal{F}^t\right]\omega_{1,t}\right\rangle + \left\langle \omega_{2,t}^T\omega_{1,t} - \omega_{2,*}^T\omega_{1,*}, \omega_{2,t}^T\,\mathbb{E}_{\sigma_1,\sigma_2}\left[g_{1,t} \mid \mathcal{F}^t\right]\right\rangle \\
&= \left\langle \omega_{2,t}^T\omega_{1,t} - \omega_{2,*}^T\omega_{1,*}, \mathbb{E}_{\sigma_1,\sigma_2}\left[g_{2,t}^T \mid \mathcal{F}^t\right]\omega_{1,t} + \omega_{2,t}^T\,\mathbb{E}_{\sigma_1,\sigma_2}\left[g_{1,t} \mid \mathcal{F}^t\right]\right\rangle \\
&\leq \frac{\left\|\omega_{2,t}^T\omega_{1,t} - \omega_{2,*}^T\omega_{1,*}\right\|^2 - \mathbb{E}_{\sigma_1,\sigma_2}\left[\left\|\omega_{1,t+1}^T\omega_{2,t+1} - \omega_{1,*}^T\omega_{2,*}\right\|^2 \mid \mathcal{F}^t\right]}{2\eta} + \frac{\eta M_1}{2}. \qquad \text{(B.20)}
\end{aligned}$$

Using (B.16) yields

$$
\begin{aligned}
&\left\langle \omega_{2,t}^T\omega_{1,t} - \omega_{2,*}^T\omega_{1,*}, \mathbb{E}_{\sigma_1,\sigma_2}\left[g_{2,t}^T \mid \mathcal{F}^t\right]\omega_{1,t}\right\rangle \\
=&\left\langle \omega_{2,t}^T\omega_{1,t} - \omega_{2,*}^T\omega_{1,*}, \rho^2\left(\omega_{1,t}^T\omega_{2,t}z^t - \omega_{1,*}^T\omega_{2,*}z^t\right)z^t\omega_{1,t}^T\omega_{1,t}\right\rangle \\
=&\rho^2\left(\omega_{1,t}^T\omega_{2,t}z^t - \omega_{1,*}^T\omega_{2,*}z^t\right)\left(\omega_{1,t}^T\omega_{2,t} - \omega_{1,*}^T\omega_{2,*}\right)z^t\left\|\omega_{1,t}\right\|^2 \\
=&\rho^2\left(\omega_{1,t}^T\omega_{2,t}z^t - \omega_{1,*}^T\omega_{2,*}z^t\right)^2\left\|\omega_{1,t}\right\|^2 = \mathbb{E}\left[l_t\left(\omega_{1,t},\omega_{2,t}\right) \mid \mathcal{F}^t\right]\cdot 2\left\|\omega_{1,t}\right\|^2. \qquad \text{(B.21)}
\end{aligned}
$$

The last equality follows from (B.15) in Lemma 5. Then, we have

$$
\begin{aligned}
&\left|\left\langle \omega_{2,t}^T\omega_{1,t} - \omega_{2,*}^T\omega_{1,*}, \omega_{2,t}^T\,\mathbb{E}_{\sigma_1,\sigma_2}\left[g_{1,t} \mid \mathcal{F}^t\right]\right\rangle\right| \\
=&\left|\left\langle \omega_{2,t}^T\omega_{1,t} - \omega_{2,*}^T\omega_{1,*}, \omega_{2,t}^T\rho^2\left(\omega_{1,t}^T\omega_{2,t}z^t - \omega_{1,*}^T\omega_{2,*}z^t\right)\omega_{2,t}z^t\right\rangle\right| \\
=&\left|\rho^2\left(\omega_{1,t}^T\omega_{2,t}z^t - \omega_{1,*}^T\omega_{2,*}z^t\right)\left(\omega_{1,t}^T\omega_{2,t} - \omega_{1,*}^T\omega_{2,*}\right)\omega_{2,t}^T\omega_{2,t}z^t\right| \\
\leq&\rho^2\left(\omega_{1,t}^T\omega_{2,t}z^t - \omega_{1,*}^T\omega_{2,*}z^t\right)^2\frac{\left\|\omega_{1,t}^T\omega_{2,t} - \omega_{1,*}^T\omega_{2,*}\right\|\left\|\omega_{2,t}^T\omega_{2,t}\right\|\left\|z^t\right\|}{\left|\left(\omega_{1,t}^T\omega_{2,t} - \omega_{1,*}^T\omega_{2,*}\right)z^t\right|} \\
\leq&\rho^2\left(\omega_{1,t}^T\omega_{2,t}z^t - \omega_{1,*}^T\omega_{2,*}z^t\right)^2\left\|\omega_{2,t}^T\omega_{2,t}\right\|\frac{\left\|\omega_{1,t}^T\omega_{2,t} - \omega_{1,*}^T\omega_{2,*}\right\|\left\|z^t\right\|}{\left|\left(\omega_{1,t}^T\omega_{2,t} - \omega_{1,*}^T\omega_{2,*}\right)z^t\right|} \\
\leq&\rho^2\left(\omega_{1,t}^T\omega_{2,t}z^t - \omega_{1,*}^T\omega_{2,*}z^t\right)^2\frac{\alpha}{\cos\left(\epsilon\right)} = \mathbb{E}\left[l_t\left(\omega_{1,t},\omega_{2,t}\right) \mid \mathcal{F}^t\right]\cdot\frac{2\alpha}{\cos\left(\epsilon\right)}. \qquad \text{(B.22)}
\end{aligned}
$$

Note that $\left\|\omega_{2,t}^T\omega_{2,t}\right\| = \sigma_{max}(\omega_{2,t}^T) \leq \left\|\omega_{2,t}^T\right\|_F \leq \alpha$ by 3 in Assumption 2. If $\left(\omega_{1,t}^T\omega_{2,t} - \omega_{1,*}^T\omega_{2,*}\right)z^t = 0$, then the inequality holds trivially. Using (B.20), (B.21) and (B.22) we obtain

$$
\begin{aligned}
&\mathbb{E}_{\sigma_1,\sigma_2}\left[l_t\left(\omega_{1,t},\omega_{2,t}\right) \mid \mathcal{F}^t\right]\cdot 2\left(\left\|\omega_{1,t}\right\|^2 - \frac{\alpha}{\cos(\epsilon)}\right) \\
\leq&\frac{\left\|\omega_{2,t}^T\omega_{1,t} - \omega_{2,*}^T\omega_{1,*}\right\|^2 - \mathbb{E}_{\sigma_1,\sigma_2}\left[\left\|\omega_{1,t+1}^T\omega_{2,t+1} - \omega_{1,*}^T\omega_{2,*}\right\|^2 \mid \mathcal{F}^t\right]}{2\eta} + \frac{\eta M_1}{2}. \qquad \text{(B.23)}
\end{aligned}
$$

From update step 6 we notice that $\|\omega_{1,t}\|^2 = \frac{1}{2} + \xi_1 = \frac{1}{2} + \frac{\alpha}{\cos(\epsilon)}$, thus, (B.23) could be further simplified as

$$\mathbb{E}_{\sigma_1,\sigma_2}\left[l_t\left(\omega_{1,t},\omega_{2,t}\right) \mid \mathcal{F}^t\right] \leq \frac{\left\|\omega_{2,t}^T\omega_{1,t} - \omega_{2,*}^T\omega_{1,*}\right\|^2 - \mathbb{E}_{\sigma_1,\sigma_2}\left[\left\|\omega_{1,t+1}^T\omega_{2,t+1} - \omega_{1,*}^T\omega_{2,*}\right\|^2 \mid \mathcal{F}^t\right]}{2\eta} + \frac{\eta M_1}{2}.$$

Applying the law of iterated expectation implies

$$\mathbb{E}\left[l_t\left(\omega_{1,t},\omega_{2,t}\right)\right] \leq \frac{\mathbb{E}\left[\left\|\omega_{2,t}^T\omega_{1,t} - \omega_{2,*}^T\omega_{1,*}\right\|^2\right] - \mathbb{E}\left[\left\|\omega_{1,t+1}^T\omega_{2,t+1} - \omega_{1,*}^T\omega_{2,*}\right\|^2\right]}{2\eta} + \frac{\eta M_1}{2}$$

By summing up all differences, we obtain

$$\begin{aligned}\sum_{t=p}^{T+p}\mathbb{E}\left[l_t\left(\omega_{1,t},\omega_{2,t}\right)\right] &\leq \frac{1}{2}\sum_{t=p}^{T+p}\frac{\mathbb{E}\left[\left\|\omega_{2,t}^T\omega_{1,t} - \omega_{2,*}^T\omega_{1,*}\right\|^2\right] - \mathbb{E}\left[\left\|\omega_{1,t+1}^T\omega_{2,t+1} - \omega_{1,*}^T\omega_{2,*}\right\|^2\right]}{\eta} \\ &\quad + \frac{M_1}{2}\eta T \\ &= \frac{1}{2}\frac{\mathbb{E}\left[\left\|\omega_{2,p}^T\omega_{1,p} - \omega_{2,*}^T\omega_{1,*}\right\|^2\right] - \mathbb{E}\left[\left\|\omega_{1,p+T+1}^T\omega_{2,p+T+1} - \omega_{1,*}^T\omega_{2,*}\right\|^2\right]}{\eta} + \frac{M_1}{2}\eta T \\ &= \mathcal{O}(\sqrt{T}).\end{aligned} \tag{B.24}$$

The last equality uses 3 from Assumption 2 and the definition of $\eta = \frac{\eta_1}{\sqrt{T}}$. The desired result in Theorem 9 follows directly from (B.24) since it holds for any p.

□

B.6. Regret with Rolling Window Analyses of dnnAdam for Two-Layer NN

Lemma 7. *In Algorithm 5, given $\omega_{2,t}, \omega_{1,t}, \omega_{2,*}, \omega_{1,*}$ and $\hat{v}_{2,t}$, there exists a bounded matrix $\tilde{v}_{2,t}$ such that*

$$\left(\sqrt{\hat{v}_{2,t}} \odot \omega_{2,t}\right)^T \omega_{1,t} - \left(\sqrt{\hat{v}_{2,t}} \odot \omega_{2,*}\right)^T \omega_{1,*} = \left(\sqrt{\tilde{v}_{2,t}}\right)^T \left(\omega_{2,t}^T \omega_{1,t} - \omega_{2,*}^T \omega_{1,*}\right), \tag{B.25}$$

where $\odot$ is an element-wise multiplication operation.

PROOF. From step 10 in Algorithm 5, v_{2t} is a matrix with same value in the same column, which in turn yields

$$\left(\sqrt{\hat{v}_{2,t}} \odot \omega_{2,t}\right)^T \omega_{1t} = \left(\sqrt{\tilde{v}_{2,t}}\right)^T \omega_{2,t}^T \omega_{1,t},$$

where $\tilde{v}_{2,t}$ is a diagonal matrix with $\tilde{v}_{2,t} = \operatorname{diag}\left(\left[\hat{v}_{2,t}\right]_{1,:}\right)$, and $\left[\hat{v}_{2,t}\right]_{1,:}$ is the 1^{st} row of matrix $\hat{v}_{2,t}$. Applying the same argument for $\left(\sqrt{\hat{v}_{2,t}} \odot \omega_{2,*}\right)^T \omega_{1,*}$ yields (B.25). Next, let us show that $\tilde{v}_{2,t}$ is bounded. It is sufficient to show that $\hat{v}_{2,t}$ is bounded. From steps 12 and 9, we conclude that

$$\hat{v}_{2,t} \leq \max\left(v_{2,1}, v_{2,2}, \cdots, v_{2,t}\right).$$

Therefore, it is sufficient to show that $v_{2,t}$ is bounded for all t. For each entry in the matrix, since $\left|\left[g_{2,t}\right]_{ij}\right| \leq G_{2,\infty}$, we obtain

$$\left|\left[v_{2,t}\right]_{ik}\right| = \left|(1-\beta_{22}) \sum_{j=1}^{t} \beta_{22}^{t-j} \left(\max_p \left[g_{2,j}\right]_{pk}^2\right)\right| \leq \left|(1-\beta_{22}) \sum_{j=1}^{t} \beta_{22}^{t-j} G_{2,\infty}^2\right| \leq G_{2,\infty}^2. \tag{B.26}$$

By combining with the fact that g_2 is bounded due to step 5 and the boundedness of $\omega_{1,t}, \omega_{2,t}, z^t$ and y^t from condition 3 in Assumption 2, Lemma 7 follows. □

Lemma 8. *In Algorithm 5, given* $m_{1,t-1}, m_{1,t}, \hat{v}_{1,t} \in \mathbb{R}^n$ *and* $m_{2,t}, \hat{v}_{2,t} \in \mathbb{R}^{n\times d}$ *for any* t*, and* $\beta_{111}, \beta_{121}, \beta_{21}$ *and* β_{22} *are constants between* 0 *and* 1 *such that* $\lambda_1 := \frac{\beta_{111}}{\beta_{21}} < 1$ *and* $\lambda_2 := \frac{\beta_{121}}{\beta_{22}} < 1$*, then*

$$\left\| \frac{1}{\sqrt{\hat{v}_{1,t}}} \odot m_{1,t-1} \right\|^2 \leq \frac{n}{(1-\beta_{111})(1-\beta_{21})(1-\lambda_1)} \tag{B.27}$$

$$\left\| \frac{1}{\sqrt{\hat{v}_{1,t}}} \odot m_{1,t} \right\|^2 \leq \frac{n}{(1-\beta_{111})(1-\beta_{21})(1-\lambda_1)} \tag{B.28}$$

$$\left\| \frac{1}{\sqrt{\hat{v}_{2,t}}} \odot m_{2,t} \right\|^2 \leq \frac{nd}{(1-\beta_{121})(1-\beta_{21})(1-\lambda_2)} \tag{B.29}$$

$$\left\| \frac{1}{\sqrt[4]{\hat{v}_{2,t}}} \odot m_{2,t} \right\|^2 \leq \frac{ndG_{2,\infty}}{(1-\beta_{121})\sqrt{1-\beta_{21}}(1-\lambda_2)}. \tag{B.30}$$

PROOF. Based on steps 6 - 12 in Algorithm 5, we obtain

$$m_{1,t} = \sum_{j=1}^{t} \left[(1-\beta_{11j}) \prod_{k=1}^{t-j} \beta_{11(t-k+1)} g_{1,j} \right] \tag{B.31}$$

$$m_{2,t} = \sum_{j=1}^{t} \left[(1-\beta_{12j}) \prod_{k=1}^{t-j} \beta_{12(t-k+1)} g_{2,j} \right] \tag{B.32}$$

$$\hat{v}_{1,t} \geq (1-\beta_{21}) \sum_{j=1}^{t} \beta_{21}^{t-j} g_{1,j} \odot g_{1,j} \tag{B.33}$$

$$\hat{v}_{2,t} \geq (1-\beta_{22}) \sum_{j=1}^{t} \beta_{22}^{t-j} g_{2,j} \odot g_{2,j}. \tag{B.34}$$

Then, combining (B.31) and (B.33) yields

$$\begin{aligned}
\left\| \frac{1}{\sqrt{\hat{v}_{1,t}}} \odot m_{1,t} \right\|^2 &\leq \sum_{i=1}^{n} \frac{\left(\sum_{j=1}^{t} \left[(1-\beta_{11j}) \prod_{k=1}^{t-j} \beta_{11(t-k+1)} \left[g_{1,j} \right]_i \right] \right)^2}{\left((1-\beta_{21}) \sum_{j=1}^{t} \beta_{21}^{t-j} \left[g_{1,j} \right]_i^2 \right)} \\
&\leq \sum_{i=1}^{n} \frac{\left(\sum_{j=1}^{t} \prod_{k=1}^{t-j} \beta_{11(t-k+1)} \left[g_{1,j} \right]_i \right)^2}{\left((1-\beta_{21}) \sum_{j=1}^{t} \beta_{21}^{t-j} \left[g_{1,j} \right]_i^2 \right)} \\
&\leq \sum_{i=1}^{n} \frac{\left(\sum_{j=1}^{t} \prod_{k=1}^{t-j} \beta_{11(t-k+1)} \right) \left(\sum_{j=1}^{t} \prod_{k=1}^{t-j} \beta_{11(t-k+1)} \left[g_{1,j} \right]_i^2 \right)}{\left((1-\beta_{21}) \sum_{j=1}^{t} \beta_{21}^{t-j} \left[g_{1,j} \right]_i^2 \right)} \\
&\leq \sum_{i=1}^{n} \frac{\left(\sum_{j=1}^{t} \beta_{111}^{t-j} \right) \left(\sum_{j=1}^{t} \beta_{111}^{t-j} \left[g_{1,j} \right]_i^2 \right)}{\left((1-\beta_{21}) \sum_{j=1}^{t} \beta_{21}^{t-j} \left[g_{1,j} \right]_i^2 \right)} \\
&\leq \frac{1}{(1-\beta_{111})(1-\beta_{21})} \sum_{i=1}^{n} \sum_{j=1}^{t} \frac{\beta_{111}^{t-j} \left[g_{1,j} \right]_i^2}{\beta_{21}^{t-j} \left[g_{1,j} \right]_i^2} \\
&\leq \frac{1}{(1-\beta_{111})(1-\beta_{21})} \sum_{i=1}^{n} \sum_{j=1}^{t} \lambda_1^{t-j} \\
&\leq \frac{n}{(1-\beta_{111})(1-\beta_{21})(1-\lambda_1)}.
\end{aligned}$$

The first inequality follows from the definition of $\hat{v}_{1,t}$, which is maximum of all $v_{1,t}$ until the current time step. The third inequality follows from the Cauchy-Schwarz inequality and the forth inequality uses the fact that $\beta_{11t} \leq \beta_{111}$ for any t. Applying the same argument to $\left\| \frac{1}{\sqrt{\hat{v}_{2,t}}} \odot m_{2,t} \right\|^2$ implies (B.29). Then, applying the fact that $\hat{v}_{1,t} \geq \hat{v_{1,t-1}}$ yields

$$\left\| \frac{1}{\sqrt{\hat{v}_{1,t}}} \odot m_{1,t-1} \right\|^2 \leq \left\| \frac{1}{\sqrt{\hat{v}_{1,t-1}}} \odot m_{1,t-1} \right\|^2 \leq \frac{n}{(1-\beta_{111})(1-\beta_{21})(1-\lambda_1)},$$

where the last inequality follows from (B.28). Lastly, $\lambda_2 = \frac{\beta_{121}}{\beta_{22}} < 1$ implies $\frac{\beta_{121}}{\sqrt{\beta_{22}}} < \lambda_2 < 1$. By combining (B.32) and (B.34), we get

$$
\begin{aligned}
\left\| \frac{1}{\sqrt[4]{\hat{v}_{2,t}}} \odot m_{2,t} \right\|^2 &\leq \sum_{p=1}^{n} \sum_{q=1}^{d} \frac{\left(\sum_{j=1}^{t} \left[(1-\beta_{12j}) \prod_{k=1}^{t-j} \beta_{12(t-k+1)} \left[g_{2,j} \right]_{pq} \right] \right)^2}{\sqrt{\left((1-\beta_{21}) \sum_{j=1}^{t} \beta_{22}^{t-j} \left[g_{2,j} \right]_{pq}^2 \right)}} \\
&\leq \sum_{p=1}^{n} \sum_{q=1}^{d} \frac{\left(\sum_{j=1}^{t} \left[\prod_{k=1}^{t-j} \beta_{12(t-k+1)} \left[g_{2,j} \right]_{pq} \right] \right)^2}{\sqrt{\left((1-\beta_{21}) \sum_{j=1}^{t} \beta_{22}^{t-j} \left[g_{2,j} \right]_{pq}^2 \right)}} \\
&\leq \sum_{p=1}^{n} \sum_{q=1}^{d} \frac{\left(\sum_{j=1}^{t} \prod_{k=1}^{t-j} \beta_{12(t-k+1)} \right) \left(\sum_{j=1}^{t} \prod_{k=1}^{t-j} \beta_{12(t-k+1)} \left[g_{2,j} \right]_{pq}^2 \right)}{\sqrt{\left((1-\beta_{21}) \sum_{j=1}^{t} \beta_{22}^{t-j} \left[g_{2,j} \right]_{pq}^2 \right)}} \\
&\leq \sum_{p=1}^{n} \sum_{q=1}^{d} \frac{\left(\sum_{j=1}^{t} \beta_{121}^{t-j} \right) \left(\sum_{j=1}^{t} \beta_{121}^{t-j} \left[g_{2,j} \right]_{pq}^2 \right)}{\sqrt{\left((1-\beta_{21}) \sum_{j=1}^{t} \beta_{22}^{t-j} \left[g_{2,j} \right]_{pq}^2 \right)}} \\
&\leq \frac{1}{(1-\beta_{121}) \sqrt{1-\beta_{21}}} \sum_{p=1}^{n} \sum_{q=1}^{d} \sum_{j=1}^{t} \frac{\beta_{121}^{t-j} \left[g_{2,j} \right]_{pq}^2}{\sqrt{\beta_{22}^{t-j} \left[g_{2,j} \right]_{pq}^2}} \\
&\leq \frac{1}{(1-\beta_{121}) \sqrt{1-\beta_{21}}} \sum_{p=1}^{n} \sum_{q=1}^{d} \sum_{j=1}^{t} \lambda_2^{t-j} \left| \left[g_{2,j} \right]_{pq} \right| \\
&\leq \frac{ndG_{2,\infty}}{(1-\beta_{121}) \sqrt{1-\beta_{21}} (1-\lambda_2)}.
\end{aligned}
$$

□

B.7. Proof of Theorem 10

PROOF. Now, let us multiply $\left\|\omega_{2,t+1}^T\omega_{1,t+1} - \omega_{2*}^T\omega_{1*}\right\|^2$ by $\sqrt{\hat{v}_{2,t}}$, then take expectation given all records until time t. Then, from steps 6 - 14, we obtain

$$\begin{aligned}
(\text{B.35}) \qquad & \mathbb{E}\left[\left\|\left(\sqrt[4]{\hat{v}_{2,t}} \odot \omega_{2,t+1}\right)^T \omega_{1,t+1} - \left(\sqrt[4]{\hat{v}_{2,t}} \odot \omega_{2,*}\right)^T \omega_{1,*}\right\|^2 \mid \mathcal{F}^t\right] \\
= & \mathbb{E}\left[\left\|\left(\sqrt[4]{\hat{v}_{2,t}} \odot \left(\omega_{2,t} - \frac{\eta}{\sqrt{\hat{v}_{2,t}}} \odot m_{2,t}\right)\right)^T \left(\omega_{1,t} - \frac{\eta}{\sqrt{\hat{v}_{1,t}}} \odot m_{1,t}\right)\right.\right. \\
& \left.\left. - \left(\sqrt[4]{\hat{v}_{2,t}} \odot \omega_{2,*}\right)^T \omega_{1,*}\right\|^2 \mid \mathcal{F}^t\right] \\
(\text{B.36})
\end{aligned}$$

$$= \mathbb{E}\left[\left\|\left(\sqrt[4]{\hat{v}_{2,t}} \odot \omega_{2,t}\right)^T \omega_{1,t} - \left(\sqrt[4]{\hat{v}_{2,t}} \odot \omega_{2,*}\right)^T \omega_{1,*}\right\|^2 \mid \mathcal{F}^t\right]$$

(B.37)

$$- 2\,\mathbb{E}\left[\left\langle \left(\sqrt{\hat{v}_{2,t}} \odot \omega_{2,t}\right)^T \omega_{1,t} - \left(\sqrt{\hat{v}_{2,t}} \odot \omega_{2,*}\right)^T \omega_{1*}, \omega_{2,t}^T \frac{\eta}{\sqrt{\hat{v}_{1,t}}} \odot m_{1,t} \right\rangle \mid \mathcal{F}^t\right]$$

(B.38)

$$- 2\,\mathbb{E}\left[\left\langle \left(\sqrt{\hat{v}_{2,t}} \odot \omega_{2,t}\right)^T \omega_{1,t} - \left(\sqrt{\hat{v}_{2,t}} \odot \omega_{2,*}\right)^T \omega_{1,*}, \left(\frac{\eta}{\sqrt{\hat{v}_{2,t}}} \odot m_{2,t}\right)^T \omega_{1,t} \right\rangle \mid \mathcal{F}^t\right]$$

$$+ 2\eta^2\,\mathbb{E}\left[\left\langle \left(\sqrt{\hat{v}_{2,t}} \odot \omega_{2,t}\right)^T \omega_{1,t} - \left(\sqrt{\hat{v}_{2,t}} \odot \omega_{2,*}\right)^T \omega_{1,*},\right.\right.$$

(B.39)

$$\left.\left. \left(\frac{\eta}{\sqrt{\hat{v}_{2,t}}} \odot m_{2,t}\right)^T \left(\frac{\eta}{\sqrt{\hat{v}_{1,t}}} \odot m_{1,t}\right) \right\rangle \mid \mathcal{F}^t\right]$$

$$+ \eta^2\,\mathbb{E}\left[\left\|\left(\sqrt[4]{\hat{v}_{2,t}} \odot \omega_{2,t}\right)^T \left(\frac{\eta}{\sqrt{\hat{v}_{1,t}}} \odot m_{1,t}\right) + \left(\sqrt[4]{\hat{v}_{2,t}} \odot \left(\frac{\eta}{\sqrt{\hat{v}_{2,t}}} \odot m_{2,t}\right)\right)^T \omega_{1,t} + \right.\right.$$

(B.40)

$$\left.\left. \left(\sqrt[4]{\hat{v}_{2,t}} \odot \left(\frac{\eta}{\sqrt{\hat{v}_{2,t}}} \odot m_{2,t}\right)\right)^T \left(\frac{\eta}{\sqrt{\hat{v}_{1,t}}} \odot m_{1,t}\right)\right\|^2 \mid \mathcal{F}^t\right].$$

Let us first consider the expectations in (B.39) and (B.40). From (B.26), we conclude that $\hat{v}_{2,t}$ is bounded. Similarly, given $\beta_{11t} = \beta_{111}\gamma_1^t$ and $\beta_{12t} = \beta_{121}\gamma_2^t$ with $0 < \gamma_1, \gamma_2 < 1$, for each

entry, we attain

$$\left|[m_{1,t}]_i\right| \leq \left|(1-\beta_{111})\sum_{j=1}^{t}\beta_{111}^{t-j}\left[g_{1,j}\right]_i\right| \leq \max_j \left|[g_{1,j}]_i\right|$$

$$\left|[m_{2,t}]_{ik}\right| \leq \left|(1-\beta_{121})\sum_{j=1}^{t}\beta_{121}^{t-j}\left[g_{2,j}\right]_{ik}\right| \leq \max_j \left|[g_{2,j}]_{ik}\right|.$$

Since $\left\|\frac{1}{\sqrt{\hat{v}_{1,t}}}\odot m_{1,t}\right\|^2$, $\left\|\frac{1}{\sqrt{\hat{v}_{2,t}}}\odot m_{2,t}\right\|^2$ and $\left\|\frac{1}{\sqrt[4]{\hat{v}_{2,t}}}\odot m_{2,t}\right\|^2$ are bounded from Lemma 8 and $\omega_{1,t}, \omega_{2,t}, \omega_{1,*}, \omega_{2,*}, \hat{v}_{2,t}$ are also bounded from Assumption 2 and Lemma 7, applying Lemma 8

and Cauchy-Schwarz inequality yields

$$
\begin{aligned}
&2\eta^2\,\mathbb{E}\left[\left\langle\left(\sqrt{\hat{v}_{2,t}}\odot\omega_{2,t}\right)^T\omega_{1,t}-\left(\sqrt{\hat{v}_{2,t}}\odot\omega_{2,*}\right)^T\omega_{1,*},\left(\frac{\eta}{\sqrt{\hat{v}_{2,t}}}\odot m_{2,t}\right)^T\right.\right.\\
&\left.\left.\left(\frac{\eta}{\sqrt{\hat{v}_{1,t}}}\odot m_{1,t}\right)\right\rangle\mid\mathscr{F}^t\right]+\eta^2\,\mathbb{E}\left[\left\|\left(\sqrt[4]{\hat{v}_{2,t}}\odot\omega_{2,t}\right)^T\left(\frac{\eta}{\sqrt{\hat{v}_{1,t}}}\odot m_{1,t}\right)\right.\right.\\
&+\left(\sqrt[4]{\hat{v}_{2,t}}\odot\left(\frac{\eta}{\sqrt{\hat{v}_{2,t}}}\odot m_{2,t}\right)\right)^T\omega_{1,t}+\left(\sqrt[4]{\hat{v}_{2,t}}\odot\left(\frac{\eta}{\sqrt{\hat{v}_{2,t}}}\odot m_{2,t}\right)\right)^T\\
&\left.\left.\left(\frac{\eta}{\sqrt{\hat{v}_{1,t}}}\odot m_{1,t}\right)\right\|^2\mid\mathscr{F}^t\right]\\
=&2\eta^2\,\mathbb{E}\left[\left\langle\left(\sqrt{\hat{v}_{2,t}}\odot\omega_{2,t}\right)^T\omega_{1,t}-\left(\sqrt{\hat{v}_{2,t}}\odot\omega_{2,*}\right)^T\omega_{1,*},\left(\frac{\eta}{\sqrt{\hat{v}_{2,t}}}\odot m_{2,t}\right)^T\right.\right.\\
&\left.\left.\left(\frac{\eta}{\sqrt{\hat{v}_{1,t}}}\odot m_{1,t}\right)\right\rangle\mid\mathscr{F}^t\right]+\eta^2\,\mathbb{E}\left[\left\|\left(\sqrt[4]{\hat{v}_{2,t}}\odot\omega_{2,t}\right)^T\left(\frac{\eta}{\sqrt{\hat{v}_{1,t}}}\odot m_{1,t}\right)\right.\right.\\
&\left.\left.+\left(\frac{\eta}{\sqrt[4]{\hat{v}_{2,t}}}\odot m_{2,t}\right)^T\omega_{1,t}+\left(\frac{\eta}{\sqrt[4]{\hat{v}_{2,t}}}\odot m_{2,t}\right)^T\left(\frac{\eta}{\sqrt{\hat{v}_{1,t}}}\odot m_{1,t}\right)\right\|^2\mid\mathscr{F}^t\right]\\
\le&\eta^2\,\mathbb{E}\left[\left\|\left(\sqrt{\hat{v}_{2,t}}\odot\omega_{2,t}\right)^T\omega_{1,t}-\left(\sqrt{\hat{v}_{2,t}}\odot\omega_{2,*}\right)^T\omega_{1,*},\right\|^2+\left\|\frac{\eta}{\sqrt{\hat{v}_{2,t}}}\odot m_{2,t}\right\|^2\right.\\
&\left.\cdot\left\|\frac{\eta}{\sqrt{\hat{v}_{1,t}}}\odot m_{1,t}\right\|^2\mid\mathscr{F}^t\right]+2\eta^2\,\mathbb{E}\left[\left\|\sqrt[4]{\hat{v}_{2,t}}\odot\omega_{2,t}\right\|^2\left\|\frac{\eta}{\sqrt{\hat{v}_{1,t}}}\odot m_{1,t}\right\|^2\right.\\
&\left.+\left\|\frac{\eta}{\sqrt[4]{\hat{v}_{2,t}}}\odot m_{2,t}\right\|^2\|\omega_{1,t}\|^2+\left\|\frac{\eta}{\sqrt[4]{\hat{v}_{2,t}}}\odot m_{2,t}\right\|^2\left\|\frac{\eta}{\sqrt{\hat{v}_{1,t}}}\odot m_{1,t}\right\|^2\mid\mathscr{F}^t\right]\\
\le&\,\eta^2\cdot M_1, \qquad \text{(B.41)}
\end{aligned}
$$

where M_1 is a fixed constant. Now, let us proceed to show an upper bound for the term in (B.37). Applying Lemma 7 to (B.37) yields

$$\begin{aligned}
&\mathbb{E}\left[\left\langle \left(\sqrt[2]{\hat{v}_{2,t}}\odot\omega_{2,t}\right)^T\omega_{1,t}-\left(\sqrt[2]{\hat{v}_{2,t}}\odot\omega_{2,*}\right)^T\omega_{1,*},\omega_{2,t}^T\frac{\eta}{\sqrt{\hat{v}_{1,t}}}\odot m_{1,t}\right\rangle \mid \mathcal{F}^t\right]\\
=&\mathbb{E}\left[\left\langle \left(\sqrt{\tilde{v}_{2,t}}\right)^T\left(\omega_{2,t}^T\omega_{1,t}-\omega_{2,*}^T\omega_{1,*}\right),\omega_{2,t}^T\frac{\eta}{\sqrt{\hat{v}_{1,t}}}\odot m_{1,t}\right\rangle \mid \mathcal{F}^t\right]\\
=&\mathbb{E}\left[\left\langle \left(\sqrt{\tilde{v}_{2,t}}\right)^T\left(\omega_{2,t}^T\omega_{1,t}-\omega_{2,*}^T\omega_{1,*}\right),\omega_{2,t}^T\frac{\eta}{\sqrt{\hat{v}_{1,t}}}\odot(\beta_{11t}m_{1,t-1}+(1-\beta_{11t})g_{1,t})\right\rangle \mid \mathcal{F}^t\right]
\end{aligned}$$

(B.42)
$$=\mathbb{E}\left[\left\langle \left(\sqrt{\tilde{v}_{2,t}}\right)^T\left(\omega_{2,t}^T\omega_{1,t}-\omega_{2,*}^T\omega_{1,*}\right),\omega_{2,t}^T\frac{\eta}{\sqrt{\hat{v}_{1,t}}}\odot\beta_{11t}m_{1,t-1}\right\rangle \mid \mathcal{F}^t\right]$$

(B.43)
$$+\mathbb{E}\left[\left\langle \left(\sqrt{\tilde{v}_{2,t}}\right)^T\left(\omega_{2,t}^T\omega_{1,t}-\omega_{2,*}^T\omega_{1,*}\right),\omega_{2,t}^T\frac{\eta}{\sqrt{\hat{v}_{1,t}}}\odot(1-\beta_{11t})g_{1,t}\right\rangle \mid \mathcal{F}^t\right].$$

Since $\omega_{2,t},\omega_{1,t},\omega_{2,*},\omega_{1,*},\frac{m_{1,t-1}}{\sqrt{\hat{v}_{1,t}}},\tilde{v}_{2,t}$ and $m_{1,t-1}$ are all bounded, for the term in (B.42), there exists a constant M_2 such that

$$\begin{aligned}
&\mathbb{E}\left[\left\langle \left(\sqrt{\tilde{v}_{2,t}}\right)^T\left(\omega_{2,t}^T\omega_{1,t}-\omega_{2,*}^T\omega_{1,*}\right),\omega_{2,t}^T\frac{\eta}{\sqrt{\hat{v}_{1,t}}}\odot\beta_{11t}m_{1,t-1}\right\rangle \mid \mathcal{F}^t\right]\\
=&\eta\beta_{11t}\,\mathbb{E}\left[\left(\omega_{1,t}^T\omega_{2,t}-\omega_{1,*}^T\omega_{2,*}\right)\sqrt{\tilde{v}_{2,t}}\omega_{2,t}^T\frac{1}{\sqrt{\hat{v}_{1,t}}}\odot m_{1,t-1}\mid \mathcal{F}^t\right]\\
\text{(B.44)}\quad \le&\frac{\eta\beta_{11t}}{2}\,\mathbb{E}\left[\left\|\left(\omega_{1,t}^T\omega_{2,t}-\omega_{1,*}^T\omega_{2,*}\right)\sqrt{\tilde{v}_{2,t}}\omega_{2,t}^T\right\|^2+\left\|\frac{1}{\sqrt{\hat{v}_{1,t}}}m_{1,t-1}\right\|^2\mid \mathcal{F}^t\right]\le\eta\beta_{11t}M_2.
\end{aligned}$$

Next, let us bound the term in (B.43). Based on Lemma 8, we have

$$\begin{aligned}
&\left|\mathbb{E}\left[\left\langle\left(\sqrt{\tilde{v}_{2,t}}\right)^T\left(\omega_{2,t}^T\omega_{1,t}-\omega_{2,*}^T\omega_{1,*}\right),\omega_{2,t}^T\frac{\eta}{\sqrt{\hat{v}_{1,t}}}\odot(1-\beta_{11t})g_{1,t}\right\rangle\mid\mathcal{F}^t\right]\right|\\
=&\eta(1-\beta_{11t})\left|\mathbb{E}\left[\left(\omega_{1,t}^T\omega_{2,t}-\omega_{1,*}^T\omega_{2,*}\right)\sqrt{\tilde{v}_{2,t}}\omega_{2,t}^T\frac{1}{\sqrt{\hat{v}_{1,t}}}\odot g_{1,t}\mid\mathcal{F}^t\right]\right|\\
\leq&\eta(1-\beta_{11t})\left\|\omega_{1,t}^T\omega_{2,t}-\omega_{1,*}^T\omega_{2,*}\right\|\mathbb{E}\left[\left\|\sqrt{\tilde{v}_{2,t}}\omega_{2,t}^T\frac{1}{\sqrt{\hat{v}_{1,t}}}\odot g_{1,t}\right\|\mid\mathcal{F}^t\right].
\end{aligned}$$

Now, let us focus on the product in the expectation. Since $\sqrt{\tilde{v}_{2,t}}\in\mathbb{R}^{d\times d}$ is a diagonal matrix, let us denote the i_{th} element on diagonal as $[\tilde{v}_{2,t}]_i$. Then,

$$\sqrt{\tilde{v}_{2,t}}\omega_{2,t}^T\frac{1}{\sqrt{\hat{v}_{1,t}}}\odot g_{1,t}=\left(\mathcal{V}_t\odot\omega_{2,t}\right)^T g_{1,t},$$

where $\mathcal{V}_{12}\in\mathbb{R}^{n\times d}$ such that $[\mathcal{V}_{12}]_{ij}=\sqrt{\frac{[\tilde{v}_{2,t}]_j}{[\hat{v}_{1,t}]_i}}$. Then, we obtain

$$\begin{aligned}
&\left|\mathbb{E}\left[\left\langle\left(\sqrt{\tilde{v}_{2,t}}\right)^T\left(\omega_{2,t}^T\omega_{1,t}-\omega_{2,*}^T\omega_{1,*}\right),\omega_{2,t}^T\frac{\eta}{\sqrt{\hat{v}_{1,t}}}\odot(1-\beta_{11t})g_{1,t}\right\rangle\mid\mathcal{F}^t\right]\right|\\
\leq&\eta(1-\beta_{11t})\left\|\omega_{1,t}^T\omega_{2,t}-\omega_{1,*}^T\omega_{2,*}\right\|\mathbb{E}\left[\|(\mathcal{V}_t\odot\omega_{2,t})\|\,\|g_{1,t}\|\mid\mathcal{F}^t\right].
\end{aligned}$$

Based on (B.26) and condition 5 from Assumption 2, we discover

$$[\mathcal{V}_{12}]_{ij}=\sqrt{\frac{[\tilde{v}_{2,t}]_j}{[\hat{v}_{1,t}]_i}}\leq\frac{G_{2,\infty}}{\mu},\tag{B.45}$$

which in turn yields

$$\begin{aligned}
&\left|\mathbb{E}\left[\left\langle\left(\sqrt{\tilde{v}_{2,t}}\right)^T\left(\omega_{2,t}^T\omega_{1,t}-\omega_{2,*}^T\omega_{1,*}\right),\omega_{2,t}^T\frac{\eta}{\sqrt{\hat{v}_{1,t}}}\odot(1-\beta_{11t})g_{1,t}\right\rangle\mid\mathcal{F}^t\right]\right| \\
\leq&\eta(1-\beta_{11t})\frac{\alpha G_{2,\infty}}{\mu}\left\|\omega_{1,t}^T\omega_{2,t}-\omega_{1,*}^T\omega_{2,*}\right\|\mathbb{E}\left[\|g_{1,t}\|\mid\mathcal{F}^t\right]. \qquad \text{(B.46)}
\end{aligned}$$

Note that in (B.45), we assume that $[\hat{v}_{1,t}]_i$ is nonzero on the i_{th} coordinate. On the other hand, if $[\hat{v}_{1,t}]_i$ is zero on the i_{th} coordinate, then it implies $[g_{1,j}]_i=0$ for $j=1,2,\cdots,t$ on the i_{th} coordinate, which in turn yields $[g_{1,t}]_i=0$. Thus, (B.46) directly follows. Then, based on step 4 in Algorithm 5, we obtain

$$\begin{aligned}
&\left|\mathbb{E}\left[\left\langle\left(\sqrt{\tilde{v}_{2,t}}\right)^T\left(\omega_{2,t}^T\omega_{1,t}-\omega_{2,*}^T\omega_{1,*}\right),\omega_{2,t}^T\frac{\eta}{\sqrt{\hat{v}_{1,t}}}\odot(1-\beta_{11t})g_{1,t}\right\rangle\mid\mathcal{F}^t\right]\right| \\
=&\eta(1-\beta_{11t})\frac{\alpha G_{2,\infty}}{\mu}\left\|\omega_{1,t}^T\omega_{2,t}-\omega_{1,*}^T\omega_{2,*}\right\|\mathbb{E}\left[\left\|\left(\omega_{1,t}^T\sigma_1\left(\omega_{2,t}z^t\right)-y^t\right)\sigma_2\left(\omega_{2,t}z^t\right)\right\|\mid\mathcal{F}^t\right] \\
\leq&\eta(1-\beta_{11t})\frac{\alpha G_{2,\infty}}{\mu}\left\|\omega_{1,t}^T\omega_{2,t}-\omega_{1,*}^T\omega_{2,*}\right\|\mathbb{E}\left[\left|\omega_{1,t}^T\sigma_1\left(\omega_{2,t}z^t\right)-\omega_{1,*}^T\sigma_1\left(\omega_{2,*}z^t\right)\right|\mid\mathcal{F}^t\right] \\
&\qquad\cdot\mathbb{E}\left[\left\|\sigma_2\left(\omega_{2,t}z^t\right)\right\|\mid\mathcal{F}^t\right] \\
=&\eta(1-\beta_{11t})\frac{\alpha G_{2,\infty}}{\mu}\left\|\omega_{1,t}^T\omega_{2,t}-\omega_{1,*}^T\omega_{2,*}\right\|\rho\left|\left(\omega_{1,t}^T\omega_{2,t}-\omega_{1,*}^T\omega_{2,*}\right)z^t\right|\rho\left\|\omega_{2,t}z^t\right\| \\
=&\eta(1-\beta_{11t})\frac{\alpha G_{2,\infty}}{\mu}\left\|\omega_{1,t}^T\omega_{2,t}-\omega_{1,*}^T\omega_{2,*}\right\|\rho^2\left|\left(\omega_{1,t}^T\omega_{2,t}-\omega_{1,*}^T\omega_{2,*}\right)z^t\right|\left\|\omega_{2,t}z^t\right\| \\
\leq&\eta(1-\beta_{11t})\frac{\alpha G_{2,\infty}}{\mu}\rho^2\left\|\omega_{2,t}\right\|\left(\omega_{1,t}^T\omega_{2,t}z^t-\omega_{1,*}^T\omega_{2,*}z^t\right)^2\frac{\left\|\omega_{1,t}^T\omega_{2,t}-\omega_{1,*}^T\omega_{2,*}\right\|\|z^t\|}{\left|\omega_{1,t}^T\omega_{2,t}z^t-\omega_{1,*}^T\omega_{2,*}z^t\right|} \\
\leq&2\eta\frac{\alpha G_{2,\infty}(1-\beta_{11t})}{\mu\cos\epsilon}\mathbb{E}\left[l_t\mid\mathcal{F}^t\right]. \qquad \text{(B.47)}
\end{aligned}$$

The last inequality follows by applying conditions 3 and 4 in Assumption 2. Next, Let us deal with the term in (B.38). Based on step 7 in Algorithm 5, we observe

$$\mathbb{E}\left[\left\langle \left(\sqrt[2]{\hat{v}_{2,t}}\odot\omega_{2,t}\right)^T\omega_{1,t}-\left(\sqrt[2]{\hat{v}_{2,t}}\odot\omega_{2,*}\right)^T\omega_{1,*},\left(\frac{\eta}{\sqrt{\hat{v}_{2,t}}}\odot m_{2,t}\right)^T\omega_{1,t}\right\rangle \mid \mathscr{F}^t\right]$$

$$=\eta\,\mathbb{E}\left[\left\langle \omega_{2,t}^T\omega_{1,t}-\omega_{2,*}^T\omega_{1,*},m_{2,t}^T\omega_{1,t}\right\rangle \mid \mathscr{F}^t\right]$$

$$=\eta\,\mathbb{E}\left[\left\langle \omega_{2,t}^T\omega_{1,t}-\omega_{2,*}^T\omega_{1,*},\left(\beta_{12t}m_{2,t-1}+(1-\beta_{12t})\,g_{2,t}\right)^T\omega_{1,t}\right\rangle \mid \mathscr{F}^t\right]$$

$$=\eta\left[\beta_{12t}\left\langle \omega_{2,t}^T\omega_{1,t}-\omega_{2,*}^T\omega_{1,*},m_{2,t-1}^T\omega_{1t}\right\rangle+(1-\beta_{12t})\left\langle \omega_{2,t}^T\omega_{1,t}-\omega_{2,*}^T\omega_{1,*},\mathbb{E}\left[g_{2,t}^T\mid\mathscr{F}^t\right]\omega_{1,t}\right\rangle\right]$$

$$=\eta\beta_{12t}\left\langle \omega_{2,t}^T\omega_{1,t}-\omega_{2,*}^T\omega_{1,*},m_{2,t-1}^T\omega_{1t}\right\rangle+\eta\,(1-\beta_{12t})$$

$$\cdot\left\langle \omega_{2,t}^T\omega_{1,t}-\omega_{2,*}^T\omega_{1,*},\left(\mathbb{E}\left[\left(\omega_{1,t}^T\sigma_1\left(\omega_{2,t}z^t\right)-y^t\right)\omega_{1,t}\left(\sigma_2\left(z^t\right)\right)^T\mid\mathscr{F}^t\right]\right)^T\omega_{1,t}\right\rangle \tag{B.48}$$

$$=\eta\beta_{12t}\left(\omega_{1,t}^T\omega_{2,t}-\omega_{1,*}^T\omega_{2,*}\right)m_{2,t-1}^T\omega_{1,t} \tag{B.49}$$

$$+\eta\rho^2\,(1-\beta_{12t})\left\langle \omega_{2,t}^T\omega_{1,t}-\omega_{2,*}^T\omega_{1,*},\left(\omega_{1,t}^T\omega_{2,t}z^t-\omega_{1,*}^T\omega_{2,*}z^t\right)z^t\omega_{1,t}^T\omega_{1,t}\right\rangle.$$

The last equality holds true due to (B.17) in Lemma 6. By using the fact that $\omega_{1,t},\omega_{2,t},\omega_{1,*},\omega_{2,*}$ and $m_{2,t-1}$ are all bounded, for the term in (B.48), there exists a constant M_3 such that

$$\left|\eta\beta_{12t}\left(\omega_{1,t}^T\omega_{2,t}-\omega_{1,*}^T\omega_{2,*}\right)m_{2,t-1}^T\omega_{1,t}\right|\leq\eta\beta_{12t}M_3. \tag{B.50}$$

At the same time, by inserting (B.15) from Lemma 5 into (B.49) we get

$$\begin{aligned}
&\eta\rho^2\left(1-\beta_{12t}\right)\left\langle \omega_{2,t}^T\omega_{1,t}-\omega_{2,*}^T\omega_{1,*},\left(\omega_{1,t}^T\omega_{2,t}z^t-\omega_{1,*}^T\omega_{2,*}z^t\right)z^t\omega_{1,t}^T\omega_{1,t}\right\rangle \\
=&\eta\rho^2(1-\beta_{12t})\left(\omega_{1,t}^T\omega_{2,t}z^t-\omega_{1,*}^T\omega_{2,*}z^t\right)\left(\omega_{1,t}^T\omega_{2,t}-\omega_{1,*}^T\omega_{2,*}\right)z^t\omega_{1,t}^T\omega_{1,t} \\
=&2(1-\beta_{12t})\eta\left\|\omega_{1,t}\right\|^2\mathbb{E}\left[l_t\mid\mathcal{F}^t\right] \\
\geq&2(1-\beta_{121})\eta\left\|\omega_{1,t}\right\|^2\mathbb{E}\left[l_t\mid\mathcal{F}^t\right]. \qquad \text{(B.51)}
\end{aligned}$$

By inserting (B.41),(B.44),(B.47), (B.50), and (B.51), into (B.35) we obtain

$$\begin{aligned}
&2\,\mathbb{E}\left[l_t\mid\mathcal{F}^t\right]\left(\left(1-\beta_{121}\right)\left\|\omega_{1,t}\right\|^2-\frac{\alpha G_{2,\infty}(1-\beta_{11t})}{\mu\cos\epsilon}\right) \\
\leq&\frac{1}{\eta}\left\{\mathbb{E}\left[\left\|\left(\sqrt[4]{\hat{v}_{2,t}}\odot\omega_{2,t+1}\right)^T\omega_{1,t+1}-\left(\sqrt[4]{\hat{v}_{2,t}}\odot\omega_{2,*}\right)^T\omega_{1,*}\right\|^2\mid\mathcal{F}^t\right]\right. \\
&\left.-\mathbb{E}\left[\left\|\left(\sqrt[4]{\hat{v}_{2,t}}\odot\omega_{2,t}\right)^T\omega_{1,t}-\left(\sqrt[4]{\hat{v}_{2,t}}\odot\omega_{2,*}\right)^T\omega_{1,*}\right\|^2\mid\mathcal{F}^t\right]\right\} \\
&+2\left(\beta_{11t}M_2+\beta_{12t}M_3\right)+\eta M_1.
\end{aligned}$$

Since $\left\|\omega_{1,t}\right\|=\sqrt{\left[\frac{1}{2}+\xi_2\right]/\left(1-\beta_{121}\right)}=\sqrt{\left[\frac{1}{2}+\frac{\alpha G_{2,\infty}}{\mu\cos(\epsilon)}\right]/\left(1-\beta_{121}\right)}$, which in turn yields

$$2\left(\left(1-\beta_{121}\right)\left\|\omega_{1,t}\right\|^2-\frac{\alpha G_{2,\infty}(1-\beta_{11t})}{\mu\cos\epsilon}\right)\geq 1.$$

Therefore, by recalling the law of iterated expectations and summing up all loss functions for $t = p, p+1, \cdots, p+T$, we get

$$\begin{aligned}\sum_{t=p}^{T+p} \mathbb{E}\left[l_t\right] \leq \frac{1}{\eta} \sum_{t=p}^{p+T} \Bigg\{ & \mathbb{E}\left[\left\| \left(\sqrt[4]{\hat{v}_{2,t}} \odot \omega_{2,t+1} \right)^T \omega_{1,t+1} - \left(\sqrt[4]{\hat{v}_{2,t}} \odot \omega_{2,*} \right)^T \omega_{1,*} \right\|^2 \right] \\ & - \mathbb{E}\left[\left\| \left(\sqrt[4]{\hat{v}_{2,t}} \odot \omega_{2,t} \right)^T \omega_{1,t} - \left(\sqrt[4]{\hat{v}_{2,t}} \odot \omega_{2,*} \right)^T \omega_{1,*} \right\|^2 \right] \Bigg\} \\ & + 2 \sum_{t=p}^{p+T} \left(\beta_{11t} M_2 + \beta_{12t} M_3 \right) + T\eta M_1. \end{aligned} \tag{B.52}$$

Applying the definition of β_{11t} and β_{12t} implies

$$\begin{aligned} & \sum_{t=p}^{p+T} \left(\beta_{11t} M_2 + \beta_{12t} M_3 \right) = \sum_{t=p}^{p+T} \left(\beta_{111} \gamma_1^t M_2 + \beta_{121} \gamma_2^t M_3 \right) \\ = & \beta_{111} M_2 \sum_{t=p}^{p+T} \gamma_1^t + \beta_{121} M_3 \sum_{t=p}^{p+T} \gamma_2^t \leq \frac{\beta_{111} M_2}{1-\gamma_1} + \frac{\beta_{121} M_3}{1-\gamma_2}. \end{aligned} \tag{B.53}$$

Since $z \in \mathbb{R}^d$, we notice that $\tilde{v}_{2,t} \in \mathbb{R}^{d\times d}$. Applying Lemma 7 yields

$$
\begin{aligned}
&\sum_{t=p}^{p+T}\Bigg\{\mathbb{E}\left[\left\|\left(\sqrt[4]{\hat{v}_{2,t}}\odot\omega_{2,t+1}\right)^T\omega_{1,t+1}-\left(\sqrt[4]{\hat{v}_{2,t}}\odot\omega_{2,*}\right)^T\omega_{1,*}\right\|^2\right]\\
&\quad-\mathbb{E}\left[\left\|\left(\sqrt[4]{\hat{v}_{2,t}}\odot\omega_{2,t}\right)^T\omega_{1,t}-\left(\sqrt[4]{\hat{v}_{2,t}}\odot\omega_{2,*}\right)^T\omega_{1,*}\right\|^2\right]\Bigg\}\\
=&\sum_{t=p}^{p+T}\Bigg\{\mathbb{E}\left[\left\|\left(\sqrt[4]{\tilde{v}_{2,t}}\right)^T\left(\omega_{2,t+1}^T\omega_{1,t+1}-\omega_{2,*}^T\omega_{1,*}\right)\right\|^2\right]\\
&\quad-\mathbb{E}\left[\left\|\left(\sqrt[4]{\tilde{v}_{2,t}}\right)^T\left(\omega_{2,t}^T\omega_{1,t}-\omega_{2,*}^T\omega_{1,*}\right)\right\|^2\right]\Bigg\}\\
=&\sum_{t=p}^{p+T}\Bigg\{\mathbb{E}\left[\sum_{i=1}^d\left[\sqrt{\tilde{v}_{2,t}}\right]_i\left[\omega_{2,t+1}^T\omega_{1,t+1}-\omega_{2,*}^T\omega_{1,*}\right]_i^2\right]\\
&\quad-\mathbb{E}\left[\sum_{i=1}^d\left[\sqrt{\tilde{v}_{2,t}}\right]_i\left[\omega_{2,t}^T\omega_{1,t}-\omega_{2,*}^T\omega_{1,*}\right]_i^2\right]\Bigg\}\\
=&\,\mathbb{E}\left[\sum_{i=1}^d\left[\sqrt{\tilde{v}_{2,p}}\right]_i\left[\omega_{2,p}^T\omega_{1,p}-\omega_{2,*}^T\omega_{1,*}\right]_i^2\right]+\sum_{t=p+1}^{T+p}\Bigg\{\mathbb{E}\left[\sum_{i=1}^d\left[\sqrt{\tilde{v}_{2,t}}\right]_i\left[\omega_{2,t}^T\omega_{1,t}-\omega_{2,*}^T\omega_{1,*}\right]_i^2\right]\\
&\quad-\mathbb{E}\left[\sum_{i=1}^d\left[\sqrt{\tilde{v}_{2,t-1}}\right]_i\left[\omega_{2,t}^T\omega_{1,t}-\omega_{2,*}^T\omega_{1,*}\right]_i^2\right]\Bigg\}\\
=&\,\mathbb{E}\left[\sum_{i=1}^d\left[\sqrt{\tilde{v}_{2,p}}\right]_i\left[\omega_{2,p}^T\omega_{1,p}-\omega_{2,*}^T\omega_{1,*}\right]_i^2\right]\\
&\quad+\sum_{t=p+1}^{T+p}\sum_{i=1}^d\left\{\mathbb{E}\left[\left[\sqrt{\tilde{v}_{2,t}}\right]_i\left[\omega_{2,t}^T\omega_{1,t}-\omega_{2,*}^T\omega_{1,*}\right]_i^2-\left[\sqrt{\tilde{v}_{2,t-1}}\right]_i\left[\omega_{2,t}^T\omega_{1,t}-\omega_{2,*}^T\omega_{1,*}\right]_i^2\right]\right\}\\
=&\,\mathbb{E}\left[\sum_{i=1}^d\left[\sqrt{\tilde{v}_{2,p}}\right]_i\left[\omega_{2,p}^T\omega_{1,p}-\omega_{2,*}^T\omega_{1,*}\right]_i^2\right]
\end{aligned}
$$

(B.54)

$$
+\sum_{t=p+1}^{T+p}\sum_{i=1}^d\mathbb{E}\left[\left[\left(\sqrt{\tilde{v}_{2,t}}-\sqrt{\tilde{v}_{2,t-1}}\right)\right]_i\left[\omega_{2,t}^T\omega_{1,t}-\omega_{2,*}^T\omega_{1,*}\right]_i^2\right],
$$

where $\left[\sqrt{\tilde{v}_{2,t}}\right]_i$ represents the i_{th} element on diagonal in matrix $\tilde{v}_{2,t}$ and $\left[\omega_{2,t}^T\omega_{1,t} - \omega_{2,*}^T\omega_{1,*}\right]_i$ represents the i_{th} coordinate in vector $\omega_{2,t}^T\omega_{1,t} - \omega_{2,*}^T\omega_{1,*}$. Since $\omega_{1,t}, \omega_{2,t}, \omega_{1,*}$ and $\omega_{2,*}$ are all bounded for any t, e.g. $\left|\left[\omega_{2,t}^T\omega_{1,t} - \omega_{2,*}^T\omega_{1,*}\right]_i^2\right| \leq W_\infty$ and $\tilde{v}_{2,t} \geq \tilde{v}_{2,t-1}$ due to the fact that $\hat{v}_{2,t} \geq \hat{v}_{2,t-1}$, (B.54) can be further simplified as

$$\begin{aligned}
&\sum_{t=p}^{p+T}\Bigg\{\mathbb{E}\left[\left\|\left(\sqrt[4]{\hat{v}_{2,t}}\odot\omega_{2,t+1}\right)^T\omega_{1,t+1} - \left(\sqrt[4]{\hat{v}_{2,t}}\odot\omega_{2,*}\right)^T\omega_{1,*}\right\|^2\right] \\
&\qquad - \mathbb{E}\left[\left\|\left(\sqrt[4]{\hat{v}_{2,t}}\odot\omega_{2,t}\right)^T\omega_{1,t} - \left(\sqrt[4]{\hat{v}_{2,t}}\odot\omega_{2,*}\right)^T\omega_{1,*}\right\|^2\right]\Bigg\} \\
\leq & W_\infty\sum_{i=1}^{d}\mathbb{E}\left[\left[\sqrt{\tilde{v}_{2,p}}\right]_i\right] + W_\infty\sum_{t=p+1}^{T+p}\sum_{i=1}^{d}\mathbb{E}\left[\left[\left(\sqrt{\tilde{v}_{2,t}} - \sqrt{\tilde{v}_{2,t-1}}\right)\right]_i\right] \\
= & W_\infty\sum_{i=1}^{d}\mathbb{E}\left[\left[\sqrt{\tilde{v}_{2,p+T}}\right]_i\right]. \qquad \text{(B.55)}
\end{aligned}$$

Substituting (B.53) and (B.55) in (B.52) gives

(B.56)

$$\sum_{t=p}^{T+p}\mathbb{E}\left[l_t\right] \leq \frac{1}{\eta}W_\infty\sum_{i=1}^{d}\mathbb{E}\left[\left[\sqrt{\tilde{v}_{2,p+T}}\right]_i\right] + 2\left(\frac{\beta_{111}M_2}{1-\gamma_1} + \frac{\beta_{121}M_3}{1-\gamma_2}\right) + T\eta M_1 = \mathcal{O}(\sqrt{T}).$$

The last equality uses the definition of $\eta = \frac{\eta_1}{\sqrt{T}}$. The desired result in Theorem 9 follows directly from (B.56) since it holds for any p. □

APPENDIX C

Appendix

C.1. Probability Distribution of LDA

Given the generative process of LDA, which is formally presented in (Blei et al., 2003), we obtain the marginal distribution of a document $d = \mathbf{w}$ with text only as

$$\begin{aligned} P_2(\mathbf{w} \mid \alpha, \beta) &= \int \hat{p}(\theta \mid \alpha) \left(\prod_{n=1}^{N} \sum_{z_k} \hat{p}(z_k \mid \theta) \hat{p}(w_n \mid z_k, \beta) \right) \mathrm{d}\theta \\ &= \int \hat{p}(\theta \mid \alpha) \left(\prod_{n=1}^{N} \sum_{i=1}^{K} \prod_{j=1}^{V} (\theta_i \beta_{ij})^{w_n^j} \right) \mathrm{d}\theta, \end{aligned}$$

which in turn yields

$$\begin{aligned} P_2(D \mid \alpha, \beta) &= \mathbb{E} \left[\int \hat{p}(\theta_d \mid \alpha) \left(\prod_{n=1}^{N} \sum_{z_{d_k}} \hat{p}(z_{d_k} \mid \theta_d) \hat{p}(w_{d_n} \mid z_{d_k}, \beta) \right) \mathrm{d}\theta_d \right] \\ &= \mathbb{E} \left[\int \hat{p}(\theta_d \mid \alpha) \left(\prod_{n=1}^{N} \sum_{i=1}^{K} \prod_{j=1}^{V} (\theta_i \beta_{ij})^{w_n^j} \right) \mathrm{d}\theta_d \right], \end{aligned}$$

where $\hat{p}(\theta_d \mid \alpha) = p(\theta_d \mid \alpha)$.

C.2. Proof of Theorem 11

PROOF. By finite sample expressivity of $g(\gamma; \cdot)$, there exists a model with parameters γ_1 such that

$$g(\gamma_1; \mathbf{s}) = \alpha^*,$$

which in turn yields

$$\tilde{\hat{p}}(\theta \mid \mathbf{s}, \gamma_1) = \hat{p}(\theta \mid g(\gamma_1; \mathbf{s})) = \hat{p}(\theta \mid \alpha^*).$$

Therefore,

$$P_2(D \mid \alpha^*, \beta^*) = \tilde{\hat{P}}_1(D \mid S, \gamma_1, \beta^*) = P_1(\mu^*, \sigma^*, \gamma_1, \beta^*).$$

Since nnLDA also optimizes over the network parameter γ, we have

$$P_1(D \mid \mu^*, \sigma^*, \gamma^*, \beta^*) \geq P_1(D \mid \mu^*, \sigma^*, \gamma_1, \beta^*),$$

and thus,

$$P_1(D \mid \mu^*, \sigma^*, \gamma^*, \beta^*) \geq P_2(D \mid \alpha^*, \beta^*).$$

□

C.3. Proof of Theorem 12

PROOF. Note that

$$
\begin{aligned}
&\frac{P_1(D \mid \mu^*, \sigma^*, \gamma^*, \beta^*) - P_2(D \mid \alpha^*, \beta^*)}{P_2(D \mid \alpha^*, \beta^*)} \\
=&\frac{\mathbb{E}\left[\int \tilde{p}(\theta_d \mid \mu^*, \sigma^*, \gamma^*) \left(\prod_{n=1}^{N} \sum_{z_{d_k}} \tilde{p}(z_{d_k} \mid \theta_d) \tilde{p}(w_{d_n} \mid z_{d_k}, \beta^*)\right) \mathrm{d}\theta_d\right]}{\mathbb{E}\left[\int \hat{p}(\theta_d \mid \alpha^*) \left(\prod_{n=1}^{N} \sum_{z_{d_k}} \hat{p}(z_{d_k} \mid \theta_d) \hat{p}(w_{d_n} \mid z_{d_k}, \beta^*)\right) \mathrm{d}\theta_d\right]} \\
&- \frac{\mathbb{E}\left[\int \hat{p}(\theta_d \mid \alpha^*) \left(\prod_{n=1}^{N} \sum_{z_{d_k}} \hat{p}(z_{d_k} \mid \theta_d) \hat{p}(w_{d_n} \mid z_{d_k}, \beta^*)\right) \mathrm{d}\theta_d\right]}{\mathbb{E}\left[\int \hat{p}(\theta_d \mid \alpha^*) \left(\prod_{n=1}^{N} \sum_{z_{d_k}} \hat{p}(z_{d_k} \mid \theta_d) \hat{p}(w_{d_n} \mid z_{d_k}, \beta^*)\right) \mathrm{d}\theta_d\right]}.
\end{aligned} \tag{C.1}
$$

Since

$$
\begin{aligned}
&\tilde{p}(\theta_d \mid \mu^*, \sigma^*, \gamma^*) \left(\prod_{n=1}^{N} \sum_{z_{d_k}} \tilde{p}(z_{d_k} \mid \theta_d) \tilde{p}(w_{d_n} \mid z_{d_k}, \beta^*)\right) \\
=&\tilde{p}(\theta_d \mid \mu^*, \sigma^*, \gamma^*) \left(\prod_{n=1}^{N} \tilde{p}(w_{d_n} \mid \theta_d, \beta^*)\right) = \prod_{n=1}^{N} \tilde{p}(w_{d_n} \mid \gamma^*, \beta^*, \mu^*, \sigma^*)
\end{aligned}
$$

and

$$
\begin{aligned}
&\hat{p}(\theta_d \mid \alpha^*) \left(\prod_{n=1}^{N} \sum_{z_{d_k}} \hat{p}(z_{d_k} \mid \theta_d) \hat{p}(w_{d_n} \mid z_{d_k}, \beta^*)\right) \\
=&\hat{p}(\theta_d \mid \alpha^*) \left(\prod_{n=1}^{N} \hat{p}(w_{d_n} \mid \theta_d, \beta^*)\right) = \prod_{n=1}^{N} \hat{p}(w_{d_n} \mid \alpha^*, \beta^*),
\end{aligned}
$$

equation (C.1) could be further simplified as

$$
\begin{aligned}
&\frac{P_1(D \mid \mu^*, \sigma^*, \gamma^*, \beta^*) - P_2(D \mid \alpha^*, \beta^*)}{P_2(D \mid \alpha^*, \beta^*)} \\
=&\frac{\mathbb{E}\left[\int \prod_{n=1}^{N} \tilde{p}(w_{d_n} \mid \mu^*, \sigma^*, \gamma^*, \beta^*) \mathrm{d}\theta_d\right] - \mathbb{E}\left[\int \prod_{n=1}^{N} \hat{p}(w_{d_n} \mid \alpha^*, \beta^*) \mathrm{d}\theta_d\right]}{\mathbb{E}\left[\int \prod_{n=1}^{N} \hat{p}(w_{d_n} \mid \alpha^*, \beta^*) \mathrm{d}\theta_d\right]} \\
\geq&\frac{\mathbb{E}\left[\int C \prod_{n=1}^{N} \hat{p}(w_{d_n} \mid \alpha^*, \beta^*) \mathrm{d}\theta_d\right] - \mathbb{E}\left[\int \prod_{n=1}^{N} \hat{p}(w_{d_n} \mid \alpha^*, \beta^*) \mathrm{d}\theta_d\right]}{\mathbb{E}\left[\int \prod_{n=1}^{N} \hat{p}(w_{d_n} \mid \alpha^*, \beta^*) \mathrm{d}\theta_d\right]} \\
=&\frac{C \cdot \mathbb{E}\left[\int \prod_{n=1}^{N} \hat{p}(w_{d_n} \mid \alpha^*, \beta^*) \mathrm{d}\theta_d\right] - \mathbb{E}\left[\int \prod_{n=1}^{N} \hat{p}(w_{d_n} \mid \alpha^*, \beta^*) \mathrm{d}\theta_d\right]}{\mathbb{E}\left[\int \prod_{n=1}^{N} \hat{p}(w_{d_n} \mid \alpha^*, \beta^*) \mathrm{d}\theta_d\right]} = C - 1.
\end{aligned}
$$

□

APPENDIX D

Appendix

D.1. Proof of Theorem 13

PROOF. Given a probability distribution P for dataset (x_i, z_i), by assumption, the stochastic gradient of the loss function is unbiased, i.e.

$$\mathbb{E}_P\left[\frac{\partial \bar{L}_i(z_i, f(x_i))}{\partial f}\right] = \frac{\partial \mathscr{R}}{\partial f}, \tag{D.1}$$

with

$$\mathscr{R}[f] = \frac{1}{|\mathscr{D}|}\sum_{(x_i,z_i)\in\mathscr{D}} L(z_i, f(x_i)) = \mathbb{E}_P\left[\bar{L}_i(z_i, f(x_i))\right]$$

and

$$\bar{L}_i(z_i, f(x_i)) = \frac{1}{|\mathscr{D}|\, P(I=i)} L(z_i, f(x_i)).$$

At iterate t, in importance-sampling-based Boosting algorithms, given probability distribution P_t and $P_t^i = P_t(I = i)$, the current gradient given a subset $\mathcal{G}^t$ of samples is

$$\begin{aligned}
\bar{g}_t^{\mathcal{G}^t} &= \frac{1}{|\mathcal{G}^t|} \sum_{(x_i,z_i)\in\mathcal{G}^t} \left.\frac{\partial \bar{L}_i(z_i, f_{t-1}(x_i) + \epsilon g(x_i))}{\partial g}\right|_{\epsilon=0} \\
&= \frac{1}{|\mathcal{G}^t|} \sum_{(x_i,z_i)\in\mathcal{G}^t} \frac{1}{|\mathcal{D}|\, P_t^i} \left.\frac{\partial L_i(z_i, f_{t-1}(x_i) + \epsilon g(x_i))}{\partial g}\right|_{\epsilon=0} \\
&= \frac{1}{|\mathcal{G}^t|} \sum_{(x_i,z_i)\in\mathcal{G}^t} \frac{1}{|\mathcal{D}|\, P_t^i} g_t^i = \frac{1}{|\mathcal{G}^t|} \sum_{k=1}^{|\mathcal{G}^t|} G_k, \qquad \text{(D.2)}
\end{aligned}$$

where $g_t^i = \left.\frac{\partial L_i(z_i, f_{t-1}(x_i)+\epsilon g(x_i))}{\partial g}\right|_{\epsilon=0}$ and G_k is the random variable corresponding to sample k. Note that

$$g_t = \frac{\partial \mathcal{R}[f_{t-1}; g]}{\partial g} = \left.\frac{\partial \mathcal{R}[f_{t-1} + \epsilon g]}{\partial g}\right|_{\epsilon=0} \qquad \text{(D.3)}$$

and

$$\mathbb{E}_{P_t}(G_t) = \mathbb{E}_{P_t}\left[\bar{g}_t^{\mathcal{G}^t}\right] = g_t, \qquad \text{(D.4)}$$

due to the unbiased gradient in (D.1). Given $\bar{g}_t^{\mathcal{G}^t}$ computed on a subset $\mathcal{G}^t$ with probability distribution P_t, we consider

$$\begin{aligned}
\mathbb{E}_{P_t}\left[\Delta^{(t)}\right] &= \|f_{t-1} - f^*\|^2 - \mathbb{E}_{P_t}\left[\|f_t - f^*\|^2 \middle| \mathcal{F}^{t-1}\right] \\
&= \|f_{t-1} - f^*\|^2 - \mathbb{E}_{P_t}\left[\left\|f_{t-1} + \alpha_t \bar{g}_t^{\mathcal{G}^t} - f^*\right\|^2 \middle| \mathcal{F}^{t-1}\right] \\
&= -2\alpha_t \left\langle f_{t-1} - f^*, \mathbb{E}_{P_t}\left[\bar{g}_t^{\mathcal{G}^t} \middle| \mathcal{F}^{t-1}\right]\right\rangle - \alpha_t^2 \mathbb{E}_{P_t}\left[\left\|\bar{g}_t^{\mathcal{G}^t}\right\|^2 \middle| \mathcal{F}^{t-1}\right]. \qquad \text{(D.5)}
\end{aligned}$$

By inserting (D.4) into (D.5), we have

$$\mathbb{E}_{P_t}\left[\Delta^{(t)}\right] = -2\alpha_t \left\langle f_{t-1} - f^*, g_t \right\rangle - \alpha_t^2 \mathbb{E}_{P_t}\left[\left\|\bar{g}_t^{g^t}\right\|^2 \middle| \mathcal{F}^{t-1}\right]. \tag{D.6}$$

Thus, maximizing $\mathbb{E}_{P_t}\left[\Delta^{(t)}\right]$ is equivalent to minimizing the variance of the gradient. Consequently, consider

$$\begin{aligned}
\mathbb{E}_{P_t}\left[\left\|\bar{g}_t^{g^t}\right\|^2 \middle| \mathcal{F}^{t-1}\right] &= \mathbb{E}_{P_t}\left[\left\|\bar{g}_t^{g^t} - g_t + g_t\right\|^2 \middle| \mathcal{F}^{t-1}\right] \\
&= \mathbb{E}_{P_t}\left[\left\|\bar{g}_t^{g^t} - g_t\right\|^2 \middle| \mathcal{F}^{t-1}\right] + \mathbb{E}_{P_t}\left[2 < \bar{g}_t^{g^t} - g_t, g_t > \middle| \mathcal{F}^{t-1}\right] + \|g_t\|^2 \\
&= \mathbb{E}_{P_t}\left[\left\|\bar{g}_t^{g^t} - g_t\right\|^2 \middle| \mathcal{F}^{t-1}\right] + 2 < \mathbb{E}_{P_t}\left[\bar{g}_t^{g^t} \middle| \mathcal{F}^{t-1}\right] - g_t, g_t > + \|g_t\|^2 \\
&= \mathbb{E}_{P_t}\left[\left\|\bar{g}_t^{g^t} - g_t\right\|^2 \middle| \mathcal{F}^{t-1}\right] + \|g_t\|^2, \qquad \text{(D.7)}
\end{aligned}$$

where the last equality holds due to (D.4). Continuing, we have

$$\mathbb{E}_{P_t}\left[\left\|\bar{g}_t^{\mathcal{G}^t}-g_t\right\|^2\middle|\mathcal{F}^{t-1}\right]=\mathbb{E}_{P_t}\left[\left\|\frac{1}{|\mathcal{G}^t|}\sum_{i\in\mathcal{G}^t}\left(\frac{1}{|\mathcal{D}|\,P_t^i}g_t^i-g_t\right)\right\|^2\middle|\mathcal{F}^{t-1}\right]$$

$$=\frac{1}{|\mathcal{G}^t|^2}\mathbb{E}_{P_t}\left[\left\|\sum_{i\in\mathcal{G}^t}\left(\frac{1}{|\mathcal{D}|\,P_t^i}g_t^i-g_t\right)\right\|^2\middle|\mathcal{F}^{t-1}\right]$$

$$=\frac{1}{|\mathcal{G}^t|^2}\mathbb{E}_{P_t}\left[\sum_{i\in\mathcal{G}^t}\left\|\frac{1}{|\mathcal{D}|\,P_t^i}g_t^i-g_t\right\|^2+\sum_{(i,j)\in\mathcal{G}^t,i\neq j}<\frac{1}{|\mathcal{D}|\,P_t^i}g_t^i-g_t,\frac{1}{|\mathcal{D}|\,P_t^j}g_t^j-g_t>\middle|\mathcal{F}^{t-1}\right]$$

$$=\frac{1}{|\mathcal{G}^t|^2}\left(\mathbb{E}_{P_t}\left[\sum_{i\in\mathcal{G}^t}\left\|\frac{1}{|\mathcal{D}|\,P_t^i}g_t^i-g_t\right\|^2\middle|\mathcal{F}^{t-1}\right]\right.$$

$$\left.+\mathbb{E}_{P_t}\left[\sum_{(i,j)\in\mathcal{G}^t,i\neq j}<\frac{1}{|\mathcal{D}|\,P_t^i}g_t^i-g_t,\frac{1}{|\mathcal{D}|\,P_t^j}g_t^j-g_t>\middle|\mathcal{F}^{t-1}\right]\right)$$

$$=\frac{|\mathcal{G}^t|}{|\mathcal{G}^t|^2}\mathbb{E}_{P_t}\left[\|G_1-g_t\|^2\middle|\mathcal{F}^{t-1}\right]$$

$$+\frac{2}{|\mathcal{G}^t|^2}\binom{|\mathcal{G}^t|}{2}<\mathbb{E}_{P_t}\left[G_1-g_t|\,\mathcal{F}^{t-1}\right],\mathbb{E}_{P_t}\left[G_2-g_t|\,\mathcal{F}^{t-1}\right]>$$

$$=\frac{1}{|\mathcal{G}^t|}\mathbb{E}_{P_t}\left[\|G_1-g_t\|^2\middle|\mathcal{F}^{t-1}\right]$$

(D.8)

$$=\frac{1}{|\mathcal{G}^t|}\left(\mathbb{E}_{P_t}\left[\|G_1\|^2\middle|\mathcal{F}^{t-1}\right]-\|g_t\|^2\right).$$

The fifth equality holds since G_i and G_j are independent, moreover, the seventh equality is valid due to (D.4). Inserting (D.8) into (D.7) yields

$$\text{(D.9)}\qquad\mathbb{E}_{P_t}\left[\left\|\bar{g}_t^{\mathcal{G}^t}\right\|^2\middle|\mathcal{F}^{t-1}\right]=\frac{1}{|\mathcal{G}^t|\,|\mathcal{D}|^2}\sum_{(x_i,z_i)\in\mathcal{D}}\frac{1}{P_t^i}\left\|g_t^i\right\|^2-\frac{1}{|\mathcal{G}^t|}\|g_t\|^2+\|g_t\|^2.$$

As (D.9) shows, maximizing $\mathbb{E}_{P_t}\left[\Delta^{(t)}\right]$ is equivalent to minimizing $\frac{1}{P_t^i}\left\|g_t^i\right\|^2$. By using the Jensen's inequality, it follows that

$$\sum_{(x_i,z_i)\in\mathcal{D}} \frac{1}{P_t^i}\left\|g_t^i\right\|^2 = \sum_{(x_i,z_i)\in\mathcal{D}} P_t^i \left(\frac{\left\|g_t^i\right\|}{P_t^i}\right)^2 \geq \left(\sum_{(x_i,z_i)\in\mathcal{D}} \left\|g_t^i\right\|\right)^2, \tag{D.10}$$

and the equality holds when $P_t^i = \left\|g_t^i\right\| / \sum_{(x_j,z_j)\in\mathcal{D}} \left\|g_t^j\right\|$. Note that $g_t^i = \left.\frac{\partial L(z_i, f_{t-1}(x_i)+\epsilon g(x_i))}{\partial g}\right|_{\epsilon=0}$ is proportional to the boosting weights $w_t(x_i, z_i)$ of sample (x_i, z_i) as stated in (4.4), therefore, the claim in (4.15) follows. □