Journal of Machine Learning Research 22 (2021) 1-38 Submitted 10/20; Revised 6/21; Published 7/21

A Greedy Algorithm for Quantizing Neural Networks

Eric Lybrand ELYBRAND@UCSD.EDU
Department of Mathematics
University of California, San Diego
San Diego, CA 92121, USA

Rayan Saab RSAAB@UCSD.EDU
Department of Mathematics, and
Halicioglu Data Science Institute
University of California, San Diego
San Diego, CA 92121, USA

Editor: Gal Elidan

Abstract

We propose a new computationally efficient method for quantizing the weights of pre-trained neural networks that is general enough to handle both multi-layer perceptrons and convolutional neural networks. Our method deterministically quantizes layers in an iterative fashion with no complicated re-training required. Specifically, we quantize each neuron, or hidden unit, using a greedy path-following algorithm. This simple algorithm is equivalent to running a dynamical system, which we prove is stable for quantizing a single-layer neural network (or, alternatively, for quantizing the first layer of a multi-layer network) when the training data are Gaussian. We show that under these assumptions, the quantization error decays with the width of the layer, i.e., its level of over-parametrization. We provide numerical experiments, on multi-layer networks, to illustrate the performance of our methods on MNIST and CIFAR10 data, as well as for quantizing the VGG16 network using ImageNet data.

Keywords: quantization, neural networks, deep learning, stochastic control, discrepancy theory

1. Introduction

Deep neural networks have taken the world by storm. They outperform competing algorithms on applications ranging from speech recognition and translation to autonomous vehicles and even games, where they have beaten the best human players at, e.g., Go (see, LeCun et al. 2015; Goodfellow et al. 2016; Schmidhuber 2015; Silver et al. 2016). Such spectacular performance comes at a cost. Deep neural networks require a lot of computational power to train, memory to store, and power to run (e.g., Han et al. 2016; Kim et al. 2016; Gupta et al. 2015; Courbariaux et al. 2015). They are painstakingly trained on powerful computing devices and then either run on these powerful devices or on the cloud. Indeed, it is well-known that the expressivity of a network depends on its architecture (Baldi and Vershynin, 2019). Larger networks can capture more complex behavior (Cybenko, 1989) and therefore, for example, they generally learn better classifiers. The trade off, of course, is that larger networks require more memory for storage as well as more power to run

computations. Those who design neural networks for the purpose of loading them onto a particular device must therefore account for the device's memory capacity, processing power, and power consumption. A deep neural network might yield a more accurate classifier, but it may require too much power to be run often without draining a device's battery. On the other hand, there is much to be gained in building networks directly into hardware, for example as speech recognition or translation chips on mobile or handheld devices or hearing aids. Such mobile applications also impose restrictions on the amount of memory a neural network can use as well as its power consumption.

This tension between network expressivity and the cost of computation has naturally posed the question of whether neural networks can be compressed without compromising their performance. Given that neural networks require computing many matrix-vector multiplications, arguably one of the most impactful changes would be to quantize the weights in the neural network. In the extreme case, replacing each 32-bit floating point weight with a single bit would reduce the memory required for storing a network by a factor of 32 and simplify scalar multiplications in the matrix-vector product. It is not clear at first glance, however, that there even exists a procedure for quantizing the weights that does not dramatically affect the network's performance.

1.1 Contributions

The goal of this paper is to propose a framework for quantizing neural networks without sacrificing their predictive power, and to provide theoretical justification for our framework. Specifically,

- We propose a novel algorithm in (2) and (3) for sequentially quantizing layers of a pre-trained neural network in a data-dependent manner. This algorithm requires no retraining of the network, requires tuning only 2 hyperparameters—namely, the number of bits used to represent a weight and the radius of the quantization alphabet—and has a run time complexity of $O(Nm)$ operations per layer. Here, N is the ambient dimension of the inputs, or equivalently, the number of features per input sample of the layer, while m is the number of training samples used to learn the quantization. This $O(Nm)$ bound is optimal in the sense that any data-dependent quantization algorithm requires reading the Nm entries of the training data matrix. Furthermore, this algorithm is parallelizable across neurons in a given layer.

- We establish upper bounds on the relative training error in Theorem 2 and the generalization error in Theorem 3 when quantizing the first layer of a neural network that hold with high probability when the training data are Gaussian. Additionally, these bounds make explicit how the relative training error and generalization error decay as a function of the overparametrization of the data.

- We provide numerical simulations in Section 6 for quantizing networks trained on the benchmark data sets MNIST and CIFAR10 using both multilayer perceptrons and convolutional neural networks. We quantize all layers of the neural networks in these numerical simulations to demonstrate that the quantized networks generalize very well even when the data are not Gaussian.

2. Notation

Throughout the paper, we will use the following notation. Universal constants will be denoted as C, c and their values may change from line to line. For real valued quantities x, y, we write $x \lesssim y$ when we mean that $x \leq Cy$ and $x \propto y$ when we mean $cy \leq x \leq Cy$. For any natural number $m \in \mathbb{N}$, we denote the set $\{1, \ldots, m\}$ by $[m]$. For column vectors $u, v \in \mathbb{R}^m$, the Euclidean inner product is denoted by $\langle u, v \rangle = u^T v = \sum_{j=1}^m u_j v_j$, the ℓ_2-norm by $\|u\|_2 = \sqrt{\sum_{j=1}^m u_j^2}$, the ℓ_1-norm by $\|u\|_1 = \sum_{j=1}^m |u_j|$, and the ℓ_∞-norm by $\|u\|_\infty = \max_{j \in [m]} |u_j|$. $B(x, r)$ will denote the ℓ_2-ball centered at x with radius r and we will use the notation $B_2^m := B(0, 1) \subset \mathbb{R}^m$. For a sequence of vectors $u_t \in \mathbb{R}^m$ with $t \in \mathbb{Z}$, the backwards difference operator Δ acts by $\Delta u_t = u_t - u_{t-1}$. For a matrix $X \in \mathbb{R}^{m \times N}$ we will denote the rows using lowercase characters x_t and the columns with uppercase characters X_t. For two matrices $X, Y \in \mathbb{R}^{m \times N}$ we denote the Frobenius norm by $\|X - Y\|_F := \sqrt{\sum_{i,j} |X_{i,j} - Y_{i,j}|^2}$. Φ will denote a L-layer neural network, or multilayer perceptron, which acts on data $x \in \mathbb{R}^{N_0}$ via

$$\Phi(x) := \varphi \circ A^{(L)} \circ \cdots \circ \varphi \circ A^{(1)}(x).$$

Here, $\varphi : \mathbb{R} \to \mathbb{R}$ is a rectifier which acts on each component of a vector, $A^{(\ell)}$ is an affine operator with $A^{(\ell)}(v) = v^T W^{(\ell)} + b^{(\ell)T}$ and $W^{(\ell)} \in \mathbb{R}^{N_\ell \times N_{\ell+1}}$ is the ℓ^{th} layer's weight matrix, $b^{(\ell)} \in \mathbb{R}^{N_{\ell+1}}$ is the bias.

3. Background

While there are a handful of empirical studies on quantizing neural networks, the mathematical literature on the subject is still in its infancy. In practice there appear to be three different paradigms for quantizing neural networks. These include quantizing the gradients during training, quantizing the activation functions, and quantizing the weights either during or after training. Guo (2018) presents an overview of these different paradigms. Any quantization that occurs during training introduces issues regarding the convergence of the learning algorithm. In the case of using quantized gradients, it is important to choose an appropriate codebook for the gradient prior to training to ensure stochastic gradient descent converges to a local minimum. When using quantized activation functions, one must suitably modify backpropagation since the activation functions are no longer differentiable. Further, enforcing the weights to be discrete during training also causes problems for backpropagation which assumes no such restriction. In any of these cases, it will be necessary to carefully choose hyperparameters and modify the training algorithm beyond what is necessary to train unquantized neural networks. In contrast to these approaches, our result allows the practitioner to train neural networks in any fashion they choose and quantizes the trained network afterwards. Our quantization algorithm only requires tuning the number of bits that are used to represent a weight and the radius of the quantization alphabet. We now turn to surveying approaches similar to ours which quantize weights after training.

A natural question to ask is whether or not for every neural network there exists a quantized representation that approximates it well on a given data set. It turns out that a partial answer to this question lies in the field of discrepancy theory. Ignoring bias

terms for now, let's look at quantizing the first layer. There we have some weight matrix $W \in \mathbb{R}^{N_0 \times N_1}$ which acts on input $x \in \mathbb{R}^{N_0}$ by $x^T W$ and this quantity is then fed through the rectifier. Of course, a layer can act on a collection of $m > 0$ inputs stored as the rows in a matrix $X \in \mathbb{R}^{m \times N_0}$ where now the rectifier acts componentwise. Focusing on just one neuron w, or column of W, rather than viewing the matrix vector product Xw as a collection of inner products $\{x_i^T w\}_{i \in [m]}$, we can think about this as a linear combination of the columns of X, namely $\sum_{t \in [N_0]} w_t X_t$. This elementary linear algebra observation now lends the quantization problem a rather elegant interpretation: is there some way of choosing quantized weights q_t from a fixed alphabet $\mathcal{A}$, such as $\{-1, 0, 1\}$, so that the walk $Xq = \sum_{t=1}^{N_0} q_t X_t$ approximates the walk $Xw = \sum_{t=1}^{N_0} w_t X_t$?

As we mentioned above, the study of the existence of such a q when $Xw = 0$ has a rather rich history from the discrepancy theory literature. Spencer (1985) in Corollary 18 was able to prove the following surprising claim. There exists an absolute constant $c > 0$ so that given N vectors $X_1, \ldots, X_N \in \mathbb{R}^m$ with $\sup_{t \in [N]} \|X_t\|_2 \leq 1$ there exists a vector $q \in \{-1, 1\}^N$ so that $\|Xw - Xq\|_\infty = \|Xq\|_\infty \leq c \log(m)$. What makes this so remarkable is that the upper bound is *independent* of N, or the number of vectors in the walk. Spencer further remarks that János Komlós has conjectured that this upper bound can be reduced to simply c. The proof of the Komlós conjecture seems to be elusive except in special cases. One special case where it is true is if we require $N < m$ and now allow $q \in \{-1, 0, 1\}^N$. Theorem 16 in Spencer (1985) then proves that there exists universal constants $c \in (0, 1)$ and $K > 0$ so that for every collection of vetors $X_1, \ldots, X_N \in \mathbb{R}^m$ with $\max_{i \in [N]} \|X_i\|_2 \leq 1$ there is some $q \in \{-1, 0, 1\}^N$ with $|\{i \in [N] : q_i = 0\}| < cN$ and $\|Xq\|_\infty \leq K$.

Spencer's result inspired others to attack the Komlós conjecture and variants thereof. Banaszczyk (1990) was able to prove a variant of Spencer's result for vectors X_t chosen from an ellipsoid. In the special case where the ellipsoid is the unit ball in $\mathbb{R}^m$, Banaszczyk's result says for any $X_1, \ldots, X_N \in B_2^m$ there exists $q \in \{-1, 1\}^N$ so that $\|Xq\|_2 \leq \sqrt{m}$. This bound is tight, as it is achieved by the walk with $N = m$ and when the vectors X_t form an orthonormal basis. Later works consider a more general notion of boundedness. Rather than controlling the infinity or Euclidean norm one might instead wonder if there exists a bit string q so that the quantized walk never leaves a sufficiently large convex set containing the origin. The first such result, to the best of our knowledge, was proven by Giannopoulos (1997). Giannopoulos proved there that for any origin-symmetric convex set $K \subset \mathbb{R}^m$ with standard Gaussian measure $\gamma(K) \geq 1/2$ and for any collection of vectors $X_1, \ldots, X_m \in B_2^m$ there exists a bit string $q \in \{-1, 1\}^m$ so that $Xq \in c \log(m) K$. Notice here that the number of vectors in this result is equal to the dimension. Banaszczyk (1998) strengthened this result by allowing the number of vectors to be arbitrary and further showed that, under the same assumption $\gamma(K) \geq 1/2$, there exists a $q \in \{-1, 1\}^N$ which satisfies $Xq \in cK$. Scaling the hypercube appropriately, this immediately implies that the bound in Spencer's result can be reduced from $c \log(m)$ to $c\sqrt{1 + \log(m)}$. Though the above results were formulated in the special case when $w = 0$, a result by Lovasz et al. (1986) proves that results in this special case naturally extend to results in the *linear discrepancy* case when $\|w\|_\infty \leq 1$, though the universal constant scales by a factor of 2.

While all of these works are important contributions towards resolving the Komlós conjecture, many important questions remain, particularly pertaining to their applicability to our problem of quantizing neural networks. Naturally the most important question

remains on how to construct such a q given X, w. A naïve first guess towards answering both questions would be to solve an integer least squares problem. That is, given a data set X, a neuron w, and a quantization alphabet $\mathcal{A}$, such as {-1, 1}, solve

$$\begin{aligned} \underset{q}{\text{minimize}} \quad & \|Xw - Xq\|_2^2 \\ \text{subject to} \quad & q_i \in \mathcal{A}, \;\; i = 1, \ldots, m. \end{aligned} \tag{1}$$

It is well-known, however, that solving (1) is NP-Hard. See, for example, Ajtai (1998). Nevertheless, there have been many iterative constructions of vectors $q \in \{-1, 1\}^N$ which satisfy the bounds in the aforementioned works. A non-comprehensive list of such works includes Bansal (2010); Lovett and Meka (2015); Rothvoss (2017); Harvey et al. (2014); Eldan and Singh (2014). Constructions of q which satisfy the bound in the result of Banaszczyk (1998) include the works of Dadush et al. (2016); Bansal et al. (2018). These works also generalize to the linear discrepancy setting. In fact, Bansal et al. (2018) prove a much more general result which allows the use of more arbitrary alphabets other than $\{-1, 1\}$. Their algorithm is random though, so their result holds with high probability on the execution of the algorithm. This is in contrast, as we will see, with our result which will hold with high probability on the draw of Gaussian data. Beyond this, the computational complexity of the algorithms in Dadush et al. (2016); Bansal et al. (2018) prohibit their use in quantizing deep neural networks. For Dadush et al. (2016), this consists of looping over $O(N_0^5)$ iterations of solving a semi-definite program and computing a Cholesky factorization for a $N_0 \times N_0$ matrix. The Gram-Schmidt walk algorithm in Bansal et al. (2018) has a run-time complexity of $O(N_0(N_0 + m)^\omega)$, where $\omega \geq 2$ is the exponent for matrix multiplication. These complexities are already quite restrictive and only give the run-time for quantizing a single neuron. As the number of neurons in each layer is likely to be large for deep neural networks, these algorithms are simply infeasible for the task at hand. As we will see, our algorithm in comparison has a run-time complexity of $O(N_0 m)$ per neuron which is optimal in the sense that any data driven approach towards constructing q will require one pass over the $N_0 m$ entries of X. Using a norm inequality on Banasczyzk's bound, the result in Bansal (2010) guarantees for $\|w\|_\infty \leq 1$ the existence of a q such that $\|Xw - Xq\|_2 \leq c\sqrt{m \log(m)}$. Provided w is a generic vector in the hypercube, namely that $\|w\|_2 \propto \sqrt{N_0}$, then a simple calculation shows that with high probability on the draw of Gaussian data X with entries having variance $1/m$ to ensure that the columns are approximately unit norm, the Gram-Schmidt walk achieves a relative error bound of $\frac{\|Xw - Xq\|_2}{\|Xw\|_2} \lesssim \sqrt{m \log(m)/N_0}$. As we will see in Theorem 2, our relative training error bound for quantizing neurons in the first layer decays like $\log(N_0)\sqrt{m/N_0}$. In other words, to achieve a relative error of less than ε in the overparametrized regime where $N_0 \gg m$, the Gram-Schmidt walk requires on the order of $\frac{m^3 \log^3(m)}{\varepsilon^6}$ floating point operations as compared to our algorithm which only requires on the order of $\frac{m^2}{\varepsilon^2}$ floating point operations.

With no quantization algorithm that is both competitive from a theoretical perspective and computationally feasible, we turn to surveying what has been done outside the mathematical realm. Perhaps the simplest manner of quantizing weights is to quantize each weight within each neuron independently. The authors in Rastegari et al. (2016) consider precisely this set-up in the context of convolutional neural networks (CNNs). For each weight matrix $W^{(\ell)} \in \mathbb{R}^{N_\ell \times N_{\ell+1}}$ the quantized weight matrix $Q^{(\ell)}$ and optimal scaling factor α_ℓ are defined

as minimizers of $\|W^{(\ell)} - \alpha Q\|_F^2$ subject to the constraint that $Q_{i,j} \in \{-1, 1\}$ for all i, j. It turns out that the analytical solution to this optimization problem is $Q_{i,j}^{(\ell)} = \text{sign}(W_{i,j}^{(\ell)})$ and $\alpha_\ell = \frac{1}{mn} \sum_{i,j} |W_{i,j}^{(\ell)}|$. This form of quantization has long been known to the digital signal processing community as Memoryless Scalar Quantization (MSQ) because it quantizes a given weight independently of all other weights. While MSQ may minimize the Euclidean distance between two weight matrices, we will see that it is far from optimal if the concern is to design a matrix Q which approximates W on an overparameterized data set. Other related approaches are investigated in, e.g., Hubara et al. (2017).

In a similar vein, Wang and Cheng (2017) consider learning a quantized factorization of the weight matrix $W = XDY$, where the matrices X, Y are ternary matrices with entries $\{-1, 0, 1\}$ and D is a full-precision diagonal matrix. While in general this is a NP-hard problem authors use a greedy approach for constructing X, Y, D inspired by the work of Kolda and O'leary (1998). They provide simulations to demonstrate the efficacy of this method on a few pre-trained models, yet no theoretical analysis is provided for the efficacy of this framework. We would like to remark that the work Kueng and Tropp (2019) gives a framework for computing factorizations of W when $\text{rank}(W) = r$ as $W = SA \in \mathbb{R}^{n \times m}$, where $S \in \{-1, 1\}^{n \times r}, A \in \mathbb{R}^{r \times m}$. The reason this work is intruiging is that it does offer a means for compressing the weight matrix W by storing a smaller analog matrix A and a binarized matrix S though it does not offer nearly as much compression as if we were to replace W by a fully quantized matrix Q. Indeed, the matrix A is not guaranteed to be binary or admit a representation with a low-complexity quantization alphabet. Nevertheless, Kueng and Tropp (2019) give conditions under which such a factorization exists and propose an algorithm which provably constructs S, A using semi-definite programming. They extend this analysis to the case when W is the sum of a rank r matrix and a sparse matrix but do not establish robustness of their factorization to more general noise models.

Extending beyond the case where the quantization alphabet is fixed a priori, Gong et al. (2014) propose learning a codebook through vector quantization to quantize only the dense layers in a convolutional neural network. This stands in contrast to our work where we quantize all layers of a network. They consider clustering weights using k-means clustering and using the centroids of the clusters as the quantized weights. Moreover, they consider three different methods of clustering, which include clustering the neurons as vectors, groups of neurons thought of as sub-matrices of the weight matrix, and quantizing the neurons and the successive residuals between the cluster centers and the neurons. Beyond the fact that this work does not consider quantizing the convolutional layers, there is the additional shortcoming that clustering the neurons or groups thereof requires choosing the number of clusters in advance and choosing a maximal number of iterations to stop after. Our algorithm gives explicit control over the alphabet in advance, requires tuning only the radius of the quantization alphabet, and runs in a fixed number of iterations. Similar to the above work, we make special mention of Deep Compression by Han et al. (2016). Deep Compression seems to enjoy compressing large networks like AlexNet without sacrificing their empirical performance on data sets like ImageNet. There, authors consider first pruning the network and quantizing the values of the (scalar-valued) weights in a given layer using k-means clustering. This method applies both to fully connected and convolutional layers. An important drawback of this quantization procedure is that the network must

be retrained, perhaps multiple times, to fine tune these learned parameters. Once these parameters have been fine tuned, the weight clusters for each layer are further compressed using Huffman coding. We further remark that quantizing in this fashion is sensitive to the initialization of the cluster weights.

4. Algorithm and Intuition

Going forward we will consider neural networks without bias vectors. This assumption may seem restrictive, but in practice one can always use MSQ with a big enough bit budget to control the quantization error for the bias. Even better, one may simply embed the m dimensional data/activations x and weights w into an $m+1$ dimensional space via $x \mapsto (x, 1)$ and $w \mapsto (w, b)$ so that $w^T x + b = (w, b)^T (x, 1)$. In other words, the bias term can simply be treated as an extra dimension to the weight vector, so we will henceforth ignore it. Given a trained neural network Φ with its associated weight matrices $W^{(\ell)}$ and a data set $X \in \mathbb{R}^{m \times N_0}$, our goal is to construct quantized weight matrices $Q^{(\ell)}$ to form a new neural network $\widetilde{\Phi}$ for which $\|\Phi(X) - \widetilde{\Phi}(X)\|_F$ is small. For simplicity and ease of exposition, we will focus on the extreme case where the weights are constrained to the ternary alphabet $\{-1, 0, 1\}$, though there is no reason that our methods cannot be applied to arbitrary alphabets.

Our proposed algorithm will quantize a given neuron independently of other neurons. Beyond making the analysis easier this has the practical benefit of allowing us to easily parallelize quantization across neurons in a layer. If we denote a neuron as $w \in \mathbb{R}^{N_\ell}$, we will sucessively quantize the weights in w in a greedy data-dependent way. Let $X \in \mathbb{R}^{m \times N_0}$ be our data matrix, and let $\Phi^{(\ell-1)}, \widetilde{\Phi}^{(\ell-1)}$ denote the analog and quantized neural networks up to layer $\ell - 1$ respectively.

In the first layer, the aim is to achieve $Xq = \sum_{t=1}^{N_0} q_t X_t \approx \sum_{t=1}^{N_0} w_t X_t = Xw$ by selecting, at the t^{th} step, q_t so the running sum $\sum_{j=1}^{t} q_j X_j$ tracks its analog $\sum_{j=1}^{t} w_j X_j$ as well as possible in an ℓ_2 sense. That is, at the t^{th} iteration, we set

$$q_t := \underset{p \in \{-1,0,1\}}{\arg\min} \| \sum_{j=1}^{t} w_j X_j - \sum_{j=1}^{t-1} q_j X_j - p X_t \|_2^2.$$

It will be more amenable to analysis, and to implementation, to instead consider the equivalent dynamical system where we quantize neurons in the first layer using

$$\begin{aligned} u_0 &:= 0 \in \mathbb{R}^m, \\ q_t &:= \underset{p \in \{-1,0,1\}}{\arg\min} \|u_{t-1} + w_t X_t - p X_t\|_2^2, \\ u_t &:= u_{t-1} + w_t X_t - q_t X_t. \end{aligned} \tag{2}$$

One can see, using a simple substitution, that $u_t = \sum_{j=1}^{t} (w_j X_j - q_j X_j)$ is the error vector at the t^{th} step. Controlling it will be a main focus in our error analysis. An interesting way of thinking about (2) is by imagining the analog, or unquantized, walk as a drunken walker

and the quantized walk is a concerned friend chasing after them. The drunken walker can stumble with step sizes w_t along an avenue in the direction X_t but the friend can only move in steps whose lengths are encoded in the alphabet $\mathcal{A}$.

In the subsequent hidden layers, we follow a slightly modified version of (2). Letting $Y := \Phi^{(\ell-1)}(X)$, $\widetilde{Y} := \widetilde{\Phi}^{(\ell-1)}(X) \in \mathbb{R}^{m\times N_\ell}$, we quantize neurons in layer ℓ by

$$\begin{aligned} u_0 &:= 0 \in \mathbb{R}^m, \\ q_t &:= \operatorname*{arg\,min}_{p\in\{-1,0,1\}} \|u_{t-1} + w_t Y_t - p\widetilde{Y}_t\|_2^2, \\ u_t &:= u_{t-1} + w_t Y_t - q_t \widetilde{Y}_t. \end{aligned} \tag{3}$$

We say the vector $q \in \mathbb{R}^{N_\ell}$ is the quantization of w. In this work, we will provide a theoretical analysis for the behavior of (2) and leave analysis of (3) for future work. To that end, we re-emphasize the critical role played by *the state variable* u_t defined in (2). Indeed, we have the identity $\|Xw - Xq\|_2 = \|u_{N_0}\|_2$. That is, the two neurons w, q act approximately the same on the batch of data X only provided the state variable $\|u_{N_0}\|_2$ is well-controlled. Given bounded input $\{(w_t, X_t)\}_t$, systems which admit uniform upper bounds on $\|u_t\|_2$ will be referred to as *stable*. When the X_t are random, and in our theoretical considerations they will be, we remark that this is a much stronger statement than proving convergence to a limiting distribution which is common, for example, in the Markov chain literature. For a broad survey of such Markov chain techniques, one may consult Meyn and Tweedie (2012). The natural question remains: is the system (2) stable? Before we dive into the machinery of this dynamical system, we would like to remark that there is a concise form solution for q_t. Denote the greedy ternary quantizer by $\mathcal{Q} : \mathbb{R} \to \{-1, 0, 1\}$ with

$$\mathcal{Q}(z) = \operatorname*{arg\,min}_{p\in\{-1,0,1\}} |z - p|.$$

Then we have the following.

Lemma 1 *In the context of* (2)*, we have for any* $X_t \neq 0$ *that*

$$q_t = \mathcal{Q}\left(w_t + \frac{X_t^T u_{t-1}}{\|X_t\|_2^2}\right). \tag{4}$$

Proof This follows simply by completing a square. Provided $X_t \neq 0$, we have by the definition of q_t

$$\begin{aligned} q_t &= \operatorname*{arg\,min}_{p\in\{-1,0,1\}} \|u_{t-1} + (w_t - p)X_t\|_2^2 = \operatorname*{arg\,min}_{p\in\{-1,0,1\}} (w_t - p)^2 + 2(w_t - p)\frac{X_t^T u_{t-1}}{\|X_{t-1}\|_2^2} \\ &= \operatorname*{arg\,min}_{p\in\{-1,0,1\}} \left((w_t - p) + \frac{X_t^T u_{t-1}}{\|X_{t-1}\|_2^2}\right)^2 - \left(\frac{X_t^T u_{t-1}}{\|X_{t-1}\|_2^2}\right)^2. \end{aligned}$$

Because the former term is always non-negative, it must be the case that the minimizer is $\mathcal{Q}\left(w_t + \frac{X_t^T u_{t-1}}{\|X_t\|_2^2}\right)$. ■

Any analysis of the stability of (2) must necessarily take into account how the vectors X_t are distributed. Indeed, one can easily cook up examples which give rise to sequences of u_t which diverge rapidly. For the sake of illustration consider restricting our attention to the case when $\|X_t\|_2 = 1$ for all t. The triangle inequality gives us the crude upper bound

$$\|X(w-q)\|_2 \leq \sum_{t=1}^{N_0} |w_t - q_t| \|X_t\|_2 = \|w-q\|_1.$$

Choosing q to minimize $\|w-q\|_1$, or any p-norm for that matter, simply reduces back to the MSQ quantizer where the weights within w are quantized independently of one another, namely $q_t = \mathcal{Q}(w_t)$. It turns out that one can effectively attain this upper bound by adversarially choosing X_t to be orthogonal to u_{t-1} for all t. Indeed, in that setting we have exactly the MSQ quantizer

$$\begin{aligned} q_t &:= \mathcal{Q}\left(w_t + X_t^T u_{t-1}\right) = \mathcal{Q}(w_t), \\ u_t &:= u_{t-1} + (w_t - q_t) X_t. \end{aligned}$$

Consequentially, by repeatedly appealing to orthogonality,

$$\|u_t\|_2^2 = \|u_{t-1} + (w_t - q_t)X_t\|_2^2 = \|u_{t-1}\|_2^2 + (w_t - q_t)^2 \|X_t\|_2^2 = \sum_{j=1}^{t} (w_j - q_j)^2.$$

Thus, for generic vectors w, and adversarially chosen X_t, the error $\|u_t\|_2$ scales like $\sqrt{t}$. Importantly, this adversarial construction requires knowledge of u_{t-1} at "time" t, in order to construct an orthogonal X_t. In that sense, this extreme case is rather contrived. In an opposite (but also contrived) extreme case, all of the X_t are equal, and therefore X_t is parallel to u_{t-1} for all t, the dynamical system reduces to a first order greedy $\Sigma\Delta$ quantizer

$$\begin{aligned} q_t &= \mathcal{Q}\left(w_t + X_t^T u_{t-1}\right) = \mathcal{Q}\left(w_t + \sum_{j=1}^{t-1} w_j - q_j\right), \\ u_t &= u_{t-1} + (w_t - q_t) X_t. \end{aligned} \tag{5}$$

Here, when $w_t \in [-1, 1]$, one can show by induction that $\|u_t\|_2 \leq 1/2$ for all t, a dramatic contrast with the previous scenario. For more details on $\Sigma\Delta$ quantization, see for example Inose et al. (1962); Daubechies and DeVore (2003).

Recall that in the present context the signal we wish to approximate is not the neuron w itself, but rather Xw. The goal therefore is not to minimize the error $\|w-q\|_2$ but rather to minimize $\|X(w-q)\|_2$, which by construction is the same as $\|u_{N_0}\|_2$. Algebraically that means carefully selecting q so that $w - q$ is in or very close to the kernel, or null-space, of the data matrix X. This immediately suggests how overparameterization may lead to better quantization. Given m data samples stored as rows in $X \in \mathbb{R}^{m \times N_0}$, having $N_0 \gg m$ or alternatively having $\dim(\text{Span}\{x_1, \ldots, x_m\}) \ll N_0$ ensures that the kernel of X is large, and one may attempt to design q so that the vector $w - q$ lies as close as possible to the kernel of X.

5. Main Results

We are now ready to state our main result which shows that (2) is stable when the input data X are Gaussian. The proofs of the following theorems are deferred to Section 9, as the proofs are quite long and require many supporting lemmata.

Theorem 2 *Suppose $X \in \mathbb{R}^{m\times N_0}$ has independent columns $X_t \sim \mathcal{N}(0, \sigma^2 I_{m\times m})$, $w \in \mathbb{R}^{N_0}$ is independent of X and satisfies $w_t \in [-1, 1]$ and $\text{dist}(w_t, \{-1, 0, 1\}) > \varepsilon$ for all t. Then, with probability at least $1 - C\exp(-cm\log(N_0))$ on the draw of the data X, if q is selected according to (2) we have that*

$$\frac{\|Xw - Xq\|_2}{\|Xw\|_2} \lesssim \frac{\sqrt{m}\log(N_0)}{\|w\|_2}, \tag{6}$$

where $C, c > 0$ are constants that depend on ε in a manner that is made explicit in the statement of Theorem 14.

Proof Without loss of generality, we'll assume $\sigma = 1/\sqrt{m}$ since this factor appears in both numerator and denominator of (6). Theorem 14 guarantees with probability at least $1 - Ce^{-cm\log(N_0)}$ that $\|u_{N_0}\|_2 = \|Xw - Xq\|_2 \lesssim \sqrt{m}\log(N_0)$. Using Lemma 8, we have $\|Xw\|_2 \gtrsim \|w\|_2$ with probability at least $1 - 2\exp(-c_{norm}m)$. Combining these two results gives us the desired statement. ■

For generic vectors w we have $\|w\|_2 \propto \sqrt{N_0}$, so in this case Theorem 2 tells us that up to logarithmic factors the relative error decays like $\sqrt{m/N_0}$. As it stands, this result suggests that it is sufficient to have $N_0 \gg m$ to obtain a small relative error. In Section 9, we address the case where the feature data X_t lay in a d-dimensional subspace to get a bound in terms of d rather than m. In other words, this suggests that the relative training error depends not on the number of training samples m but on the intrinsic dimension of the features d. See Lemma 16 for details.

Our next result shows that the quantization error is well-controlled in the span of the training data so that the quantized weights generalize to new data.

Theorem 3 *Define X, w and q as in the statement of Theorem 2 and further suppose that $N_0 \gg m$. Let $X = U\Sigma V^T$ be the singular value decomposition of X, and let $z = Vg$ where $g \sim \mathcal{N}(0, \sigma_z^2 I_{m\times m})$ is drawn independently of X, w. In other words, suppose z is a Gaussian random variable drawn from the span of the training data x_i. Then with probability at least $1 - Ce^{-cm\log(N_0)} - 3\exp(-c'm)$ we have*

$$|z^T(w - q)| \lesssim \left(\frac{\sigma_z m}{\sigma(\sqrt{N_0} - \sqrt{m})}\right)\sigma m\log(N_0). \tag{7}$$

Proof To begin, notice that the error bound in Theorem 14 easily extends to the set $X^T(B_1^{N_0}) := \{y \in \mathbb{R}^{N_0} : y = \sum_{i=1}^m a_i x_i,\ \|a\|_1 \le 1\}$ with a simple yet pessimistic argument. With probability at least $1 - Ce^{-cm\log(N_0)}$, for any $y \in X^T(B_1^{N_0})$ one has

$$\begin{aligned} |y^T(w - q)| &= \left|\sum_{i=1}^m a_i x_i^T(w - q)\right| \le \sum_{i=1}^m |a_i||x_i^T(w - q)| \\ &\lesssim \sum_{i=1}^m |a_i|\sigma m\log(N_0) \le \sigma m\log(N_0). \end{aligned} \tag{8}$$

Now for z as defined in the statement of this theorem define $p := \alpha^* z$, where

$$\alpha^* := \arg\max_{\alpha \geq 0} \quad \alpha$$
$$\text{subject to} \quad \alpha z \in X^T(B_1^{N_0}).$$

If it were the case that $\alpha^* > 0$ then we could use (8) to get the bound

$$|z^T(w-q)| = \frac{1}{\alpha^*}\|p^T X(w-q)\|_2 \lesssim \frac{\sigma m \log(N_0)}{\alpha^*}.$$

So, it behooves us to find a strictly positive lower bound on α^*. By the assumption that $z = Vg$, there exists $h \in \mathbb{R}^m$ so that $X^T h = z$. Since $N_0 > m$, X^T is injective almost surely and therefore h is unique. Setting $v := \|h\|_1^{-1} h$, observe that $X^T v = \|h\|_1^{-1} z$ and $\sum_{i=1}^m v_i = 1$. It follows that $\alpha^* \geq \|h\|_1^{-1}$. To lower bound $\|h\|_1^{-1}$, note

$$\begin{aligned}\|h\|_1^{-1}\|z\|_2 = \|X^T v\|_2 &\geq \min_{\|y\|_1=1} \|X^T y\|_2 \geq \left(\min_{\|\eta\|_2=1} \|X^T \eta\|_2\right) \min_{\|y\|_1=1} \|y\|_2 \\ &\gtrsim \sigma(\sqrt{N_0} - \sqrt{m}) \min_{\|y\|_1=1} \|y\|_2 = \frac{\sigma(\sqrt{N_0} - \sqrt{m})}{\sqrt{m}}. \end{aligned} \tag{9}$$

The penultimate inequality in the above equation follows directly from well-known bounds on the singular values of isotropic subgaussian matrices that hold with probability at least $1 - 2\exp(-c'm)$ (see Vershynin 2018). To make the argument explicit, note that $X^T = \sigma G$, where $G \in \mathbb{R}^{N_0 \times m}$ is a matrix whose rows are independent and identically distributed gaussians with $\mathbb{E}[g_i g_i^T] = I_{m\times m}$ and are thus isotropic. Using Lemma 8 we have with probability at least $1 - \exp(-c_{norm} m/4)$ that $\|z\|_2 = \|Vg\|_2 = \|g\|_2 \lesssim \sigma_z \sqrt{m}$. Substituting in (9), we have

$$\|h\|_1^{-1} \gtrsim \frac{\sigma(\sqrt{N_0} - \sqrt{m})}{\sigma_z m}.$$

Therefore, putting it all together, we have with probability at least $1 - Ce^{-cm\log(N_0)} - 3\exp(-c''m)$

$$|z^T(w-q)| \lesssim \left(\frac{\sigma_z m}{\sigma(\sqrt{N_0} - \sqrt{m})}\right) \sigma m \log(N_0).$$

■

Remark 4 *In the special case when $\sigma_z = \sigma\sqrt{N_0/m}$, i.e. when $\mathbb{E}[\|z\|_2^2|V] = \mathbb{E}\|x_i\|_2^2 = \sigma^2 N_0$ and $N_0 \gg m$, the bound in Theorem 3 reduces to*

$$\frac{\sigma\sqrt{N_0 m}}{\sigma(\sqrt{N_0} - \sqrt{m})} \sigma m \log(N_0) \lesssim \sigma m^{3/2} \log(N_0).$$

Furthermore, when the row data are normalized in expectation, or when $\sigma^2 = N_0^{-1}$, this bound becomes $\frac{m^{3/2}\log(N_0)}{\sqrt{N_0}}$.

Remark 5 *Under the low-dimensional assumptions in Lemma 16, the bound* (7) *and the discussion in Remark 4 apply when* m *is replaced with* d*.*

Remark 6 *The context of Theorem 3 considers the setting when the data are overparameterized, and there are fewer training data points used than the number of parameters. It is natural to wonder if better generalization bounds could be established if many training points were used to learn the quantization. In the extreme setting where* $m \gg N_0$*, one could use a covering or* ε*-net like argument. Specifically, if a new sample* z *were* ε *close to a training example* x*, then* $|(z-x)^T w| \leq \|z-x\|\|w\| \lesssim \varepsilon\sqrt{N_0}$*. Such an argument could be done easily when the number of training points is large enough that it leads to a small* ε*. On the other hand, the curse of dimensionality stipulates that for this argument to work it would require an exponential number of training points, e.g., of order* $(\frac{1}{\varepsilon})^d$ *if the training data were in a* d*-dimensional subspace and did not exhibit any further structure. We choose to focus on the overparametrized setting instead, but think that investigating the "intermediate" setting, where one has more training data coming from a structured* d*-dimensional set than parameters, is an interesting avenue for future work.*

Our technique for showing the stability of (2), i.e., the boundedness of $\|X(w-q)\|_2$, relies on tools from drift analysis. Our analysis is inspired by the works of Pemantle and Rosenthal (1999) and Hajek (1982). Given a real valued stochastic process $\{Y_t\}_{t\in\mathbb{N}}$, those authors give conditions on the increments $\Delta Y_t := Y_t - Y_{t-1}$ to uniformly bound the moments, or moment generating function, of the iterates Y_t. These bounds can then be transformed into a bound in probability on an individual iterate Y_t using Markov's inequality. Recall that we're interested in bounding the state variable u_t induced by the system (2) which quantizes the first layer of a neural network. In situations like ours it is natural to analyze the increments of u_t since the innovations (w_t, X_t) are jointly independent. To invoke the results of Pemantle and Rosenthal (1999); Hajek (1982) we'll consider the associated stochastic process $\{\|u_t\|_2^2\}_{t\in[N_0]}$. Beyond the fact that our intent is to control the norm of the state variable, it turns out that stability analyses of vector valued stochastic processes typically involve passing the process through a real-valued and oftentimes quadratic function known as a Lyapunov function. There is a wide variety of stability theorems which require demonstrating certain properties of the image of a stochastic process under a Lyapunov function. For example, Lyapunov functions play a critical role in analyzing Markov chains as detailed in Menshikov et al. (2016). However, there are a few details which preclude us from using one of these well-known stability results for the process $\{\|u_t\|_2^2\}_{t\in[N_0]}$. First, even though the innovations (w_t, X_t) are jointly independent the increments

$$\Delta\|u_t\|_2^2 = (w_t - q_t)^2\|X_t\|_2^2 + 2(w_t - q_t)\langle X_t, u_{t-1}\rangle \tag{10}$$

have a dependency structure encoded by the bit sequence q. In addition to this, the bigger challenge in the analysis of (2) is the discontinuity inherent in the definition of q_t. Addressing this discontinuity in the analysis requires carefully handling the increments on the events where q_t is fixed.

Based on our prior discussion, towards the end of Section 4, it would seem that for generic data sets the stability of (2) lies somewhere in between the behavior of MSQ and $\Sigma\Delta$ quantizers, and that behavior crucially depends on the "dither" terms $X_t^T u_{t-1}$. For the sake of analysis then, we will henceforth make the following assumptions.

Assumption 1 *The sequence $(w_t, X_t)_t$ defined on the probability space $(\Omega, \mathcal{F}, \mathbb{P})$ is adapted to the filtration $\mathcal{F}_t$. Further, all X_t and w_t are jointly independent.*

Assumption 2 $\|W^{(\ell)}\|_\infty = \sup_{i,j} |W^{(\ell)}_{i,j}| \leq 1$.

Assumption 1's stipulation that the X_t are independent of the weights is a simplifying relaxation, and our proof technique handles the case when the w_t are deterministic. The joint independence of the X_t could be realized by splitting the global population of training data into two populations where one is used to train the analog network and another to train the quantization. In the hypotheses of Theorem 2 it is also assumed that the entries of the weight vector w_t are sufficiently separated from the characters of the alphabet $\{-1, 0, 1\}$. We want to remark that this is simply an artifact of the proof. In succinct terms, the proof strategy relies on showing that the moment generating function of the increment $\Delta\|u_t\|_2^2$ is strictly less than 1 conditioned on the event that $\|u_{t-1}\|_2$ is sufficiently large. In the extreme case where the weights are already quantized to $\{-1, 0, 1\}$, this aforementioned event is the empty set since the state variable u_t is identically the zero function. As such, the conditioning is ill-defined. To avoid this technicality, we assume that the neural network we wish to quantize is not already quantized, namely $\text{dist}(w_t, \{-1, 0, 1\}) > \varepsilon$ for some $\varepsilon > 0$ and for all $t \in [N_\ell]$. The proof technique could easily be adapted to the case where the w_t are deterministic and this hypothesis is violated for $O(1)$ weights with only minor changes to the main result, but we do not include these modifications to keep the exposition as clear as possible. Assumption 2 is quite mild, and can be realized by scaling all neurons in a given layer by $\|W\|_\infty^{-1}$. Choosing the ternary vector q according to the scaled neuron $\|W\|_\infty^{-1} w$, any bound of the form $\|X(\|W\|_\infty^{-1} w - q)\|_2 \leq \alpha$ immediately gives the bound $\|X(w - \|W\|_\infty q)\|_2 \leq \alpha \|W\|_\infty$. In other words, at run time the network can use the scaled ternary alphabet $\{-\|W\|_\infty, 0, \|W\|_\infty\}$.

6. Numerical Simulations

We present three stylized examples which show how our proposed quantization algorithm affects classification accuracy on three benchmark data sets. In the following tables and figures, we'll refer to our algorithm as Greedy Path Following Quantization, or GPFQ for short. We look at classifying digits from the MNIST data set using a multilayer perceptron, classifying images from the CIFAR10 data set using a convolutional neural network, and finally looking at classifying images from the ILSVRC2012 data set, also known as ImageNet, using the VGG16 network (Simonyan and Zisserman, 2014). We trained both networks using Keras (Chollet et al., 2015) with the Tensorflow backend on a a 2020 MacBook Pro with a M1 chip and 16GB of RAM. Note that for the first two experiments our aim here is not to match state of the art results in training the unquantized neural networks. Rather, our goal is to demonstrate that given a trained neural network, our quantization algorithm yields a network that performs similarly. Below, we mention our design choices for the sake of completeness, and to demonstrate that our quantization algorithm does not require any special engineering beyond what is customary in neural network architectures. We have made our code available on GitHub at `https://github.com/elybrand/quantized_neural_networks`.

Our implementations for these simulations differ from the presentation of the theory in a few ways. First, we do not restrict ourselves to the particular ternary alphabet of $\{-1, 0, 1\}$. In practice, it is much more useful to replace this with the equispaced alphabet $\mathcal{A} = \alpha \times \{-1 + \frac{2j}{M-1} : j \in \{0, 1, \ldots, M-1\}\} \subset [-\alpha, \alpha]$, where M is fixed in advance and α is chosen by cross-validation. Of course, this includes the ternary alphabet $\{-\alpha, 0, \alpha\}$ as a special case. The intuition behind choosing the alphabet's radius α is to better capture the dynamic range of the true weights. For this reason we choose for every layer $\alpha_\ell = C_\alpha \text{median}(\{|W_{i,j}^{(\ell)}|\}_{i,j})$ where the constant C_α is fixed for all layers and is chosen by cross-validation. Thus, the cost associated with allowing general alphabets $\mathcal{A}$ is storing a floating point number for each layer (i.e., α_ℓ) and $N_\ell \times N_{\ell+1}$ bit strings of length $\log_2(2M+1)$ per layer as compared to $N_\ell \times N_{\ell+1}$ floats per layer in the unquantized setting.

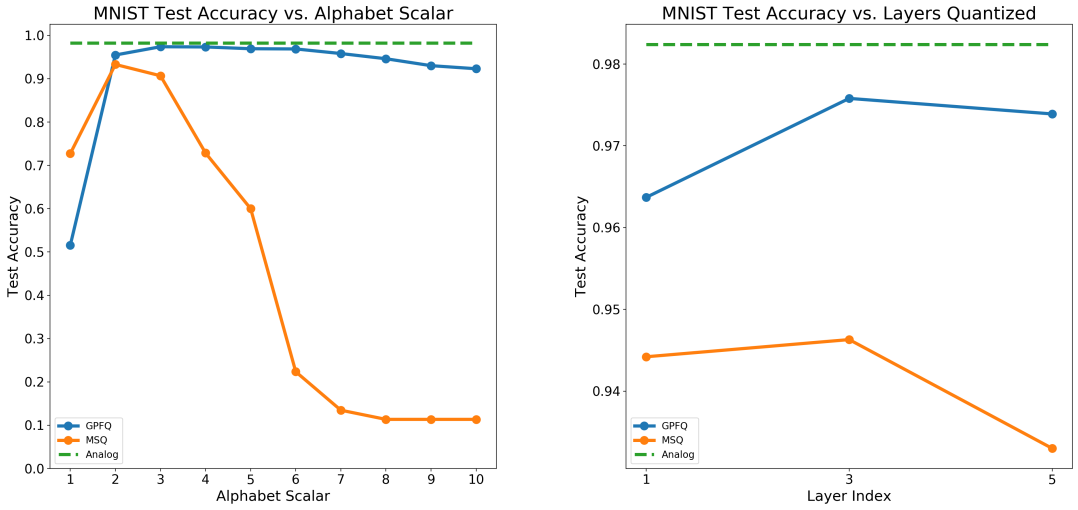
6.1 Multilayer Perceptron with MNIST

We trained a multilayer perceptron to classify MNIST digits (28×28 images) with two hidden layers. The first layer has 500 neurons, the second has 300 neurons, and the output layer has 10 neurons. We also used batch normalization layers (Ioffe and Szegedy, 2015) after each hidden layer and chose the ReLU activation function for all layers except the last where we used softmax. We trained the unquantized network on the full training set of $60,000$ digits without any preprocessing. 20% of the training data was used as validation during training. We then tested on the remaining $10,000$ images not used in the training set. We used categorical cross entropy as our loss function during training and the Adam optimizer—see Kingma and Ba (2014)—for 100 epochs with a minibatch size of 128. After training the unquantized model we used $25,000$ samples from the training set to train the quantization. We used the same data to quantize each layer rather than splitting the data for each layer. For this experiment we restricted the alphabet to be ternary and cross-validated over the alphabet scalar $C_\alpha \in \{1, 2, \ldots, 10\}$. The results for each choice of C_α are displayed in Figure 1a. As a benchmark we compared against a network quantized using MSQ, so each weight was quantized to the element of $\mathcal{A}$ that is closest to it. As we see in Figure 1a, the MSQ quantized network exhibits a high variability in its performance as a function of the alphabet scalar, whereas the GPFQ quantized network exhibits more stable behavior. Indeed, for a number of consecutive choices of C_α the performance of the GPFQ quantized network was close to its unquantized counterpart. To illustrate how accuracy was affected as subsequent layers were quantized, we ran the following experiment. First, we chose the best alphabet scalar C_α for each of the MSQ and GPFQ quantized networks separately. We then measured the test accuracy as each subsequent layer of the network was quantized, leaving the later ones unchanged. The median time it took to quantize a network was 288 seconds, or about 5 minutes. The results for MSQ and GPFQ are shown in Figure 1b. Figure 1b demonstrates that GPFQ is able to "error correct" in the sense that quantizing a later layer can correct for errors introduced when quantizing previous ones. We also remark that in this setting we replace 32 bit floating point weights with $\log_2(3)$ bit weights. Consequentially, we have compressed the network by a factor of approximately 20, and yet the drop in test accuracy for GPFQ was minimal. Further, this quick calculation assumes we use $\log_2(3)$ bits to represent those weights which are quantized to zero. However, there are other important consequences for setting weights to zero. From a hardware perspective, the

benefit is that forward propagation requires less energy due to there being fewer connections between layers. From a software perspective, multiplication by zero is an incredibly stable operation.

6.2 Convolutional Neural Network with CIFAR10

Even though our theory was phrased in the language of multilayer perceptrons it is easy to rephrase it using the vocabulary of convolutional neural networks. Here, neurons are kernels and the data are patches from the full images or their feature data in the hidden layers. These patches have the same dimensions as the kernel. Matrix convolution is defined in terms of Hilbert-Schmidt inner products between the kernel and these image patches. In other words, if we were to vectorize both the kernel and the image patches then we could take the usual inner product on vectors and reduce back to the case of a multilayer perceptron. This is exactly what we do in the quantization algorithm. Since every channel of the feature data has its own kernel we quantize each channel's kernel independently.



(a) (b)

Figure 1: Comparison of GPFQ and MSQ quantized network performance on MNIST using a ternary alphabet. Figure 1a illustrates how the top-1 accuracy on the test set behaves for various alphabet scalars C_α. Figure 1b demonstrates how the two quantized networks behave as each fully connected layer is successively quantized using the best alphabet scalar C_α for each network. We only plot the layer indices for fully connected layers as these are the only layers we quantize.

We trained a convolutional neural network to classify images from the CIFAR10 data set with the following architecture

$$2 \times 32C3 \to MP2 \to 2 \times 64C3 \to MP2 \to 2 \times 128C3 \to 128FC \to 10FC.$$

Here, $2 \times N$ C3 denotes two convolutional layers with N kernels of size 3×3, MP2 denotes a max pooling layer with kernels of size 2×2, and nFC denotes a fully connected layer with n neurons. Not listed in the above schematic are batch normalization layers which we place before every convolutional and fully connected layer except the first. During training we also use dropout layers after the max pooling layers and before the final output layer. We use the ReLU function for every layer's activation function except the last layer where we use softmax. We preprocess the data by dividing the pixel values by 255 which normalizes them in the range $[0, 1]$. We augment the data set with width and height shifts as well as horizontal flips for each image. Finally, we train the network to minimize categorical cross entropy using stochastic gradient descent with a learning rate of 10^{-4}, momentum of 0.9, and a minibatch size of 64 for 400 epochs. For more information on dropout layers and pooling layers see, for example, Hinton et al. (2012) and Weng et al. (1992), respectively.

We trained the unquantized network on the full set of $50,000$ training images. For training the quantization we only used the first $5,000$ images from the training set. As we did with the multilayer perceptron on MNIST, we cross-validated the alphabet scalars C_α over the range $\{2, 3, 4, 5, 6\}$ and chose the best scalar for the benchmark MSQ network and the best GPFQ quantized network separately. Additionally, we cross-validated over the number of elements in the quantization alphabet, ranging over the set $M \in \{3, 4, 8, 16\}$ which corresponds to the set of bit budgets $\{\log_2(3), 2, 3, 4\}$. The median time it took to quantize the network using GPFQ was 1830 seconds, or about 30 minutes. The results of these experiments are shown in Table 1. In particular, the table shows that the performance of GPFQ degrades gracefully as the bit budget decreases, while the performance of MSQ drops dramatically. In this experiment, the best bit budget for both MSQ and GPFQ networks was 4 bits, or 16 characters in the alphabet. We plot the test accuracies for the best MSQ and the best GPFQ quantized network as each layer is quantized in Figure 2a. Both networks suffer from a drop in test accuracy after quantizing the second layer, but (like in the first experiment) GPFQ recovers from this dip in subsequent layers while MSQ does not. Finally, to illustrate the difference between the two sets of quantized weights in this layer we histogram the weights in Figure 2b.

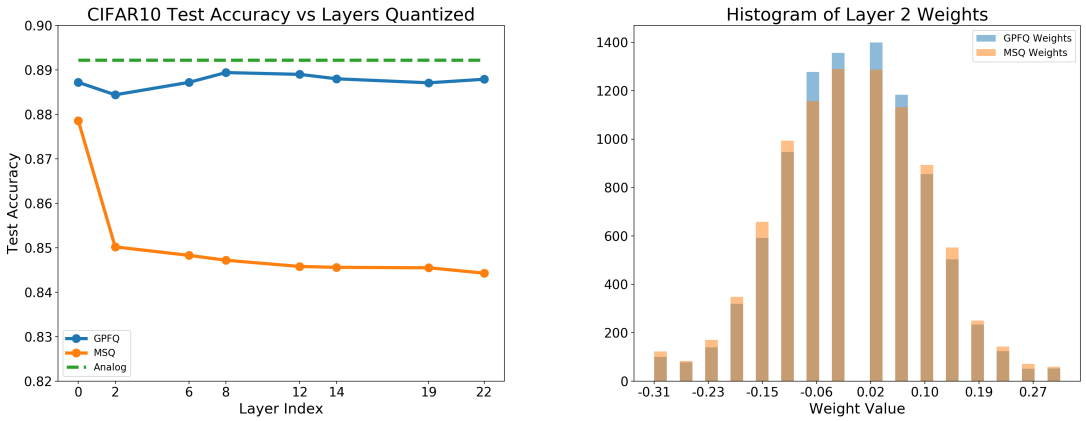
6.3 VGG16 on Imagenet Data

The previous experiments were restricted to settings where there are only 10 categories of images. To illustrate that our quantization scheme and our theory work well on more complex data sets we considered quantizing the weights of VGG16 (Simonyan and Zisserman, 2014) for the purpose of classifying images from the ILSVRC2012 validation set (Russakovsky et al., 2015). This data set contains 50,000 images with 1,000 categories. Since 90% of all weights in VGG16 are in the fully connected layers, we took a similar route as Gong et al. (2014) and only considered quantizing the weights in the fully connected layers. We preprocessed the images in the manner that the ImageNet guidelines specify. First, we resize the smallest edge of the image to 256 pixels by using bicubic interpolation over 4×4 pixel neighborhoods, and resizing the larger edge of the image to maintain the original image's aspect ratio. Next, all pixels outside the central 224×224 pixels are cropped

CIFAR10 Top-1 Test Accuracy

Bits	C_α	Analog	GPFQ	MSQ
$\log_2(3)$	2	0.8922	**0.7487**	0.1347
	3	0.8922	0.7350	0.1464
	4	0.8922	0.6919	0.0991
	5	0.8922	0.5627	0.1000
	6	0.8922	0.3515	0.1000
2	2	0.8922	0.7522	0.2209
	3	0.8922	**0.8036**	0.2800
	4	0.8922	0.7489	0.1742
	5	0.8922	0.6748	0.1835
	6	0.8922	0.5365	0.1390
3	2	0.8922	0.7942	0.4173
	3	0.8922	0.8670	0.3754
	4	0.8922	**0.8710**	0.5014
	5	0.8922	0.8567	0.5652
	6	0.8922	0.8600	0.5360
4	2	0.8922	0.8124	0.4525
	3	0.8922	0.8778	0.7776
	4	0.8922	0.8879	0.8443
	5	0.8922	**0.8888**	0.8291
	6	0.8922	0.8810	0.7831

Table 1: This table documents the test accuracies for the analog and quantized neural networks on CIFAR10 data for the various choices of alphabet scalars C_α and bit budgets.



(a) (b)

Figure 2: Figure 2a shows how the top-1 test accuracy degrades as we quantize layers successively and leave remaining layers unquantized for the best MSQ and the best GPFQ quantized networks according to the results in Table 1. We only plot the layer indices for fully connected and convolutional layers as these are the only layers we quantize. Figure 2b is a histogram of the quantized weights for the MSQ and GPFQ quantized networks at the second convolutional layer.

out. The image is then saved with red, green, blue (RGB) channel order[1]. Finally, these processed images are further preprocessed by the function specified for VGG16 in the Keras preprocessing module. For this experiment we restrict the GPFQ quantizer to the alphabet $\{-1, 0, 1\}$. We cross-validate over the alphabet scalar $C_\alpha \in \{2, 3, 4, 5\}$. 1500 images were randomly chosen to learn the quantization. To assess the quality of the quantized network we used 20000 randomly chosen images disjoint from the set of images used to perform the quantization and measured the top-1 and top-5 accuracy for the original VGG16 model, GPFQ, and MSQ networks. The median time it took to quantize VGG16 using GPFQ was 15391 seconds, or about 5 hours. The results from this experiment can be found in Table 2. Remarkably, the best GPFQ network is able to get within 0.65% and 0.42% of the top-1 and top-5 accuracy of the analog model, respectively. In contrast, the best MSQ model can do is get within 1.24% and 0.56% of the top-1 and top-5 accuracy of the analog model, respectively. Importantly, as we saw in the previous two experiments, here again we observe a notable instability of test accuracy with respect to C_α for the MSQ model whereas for the GPFQ model the test accuracy is more well-controlled. Moreover, just as in the CIFAR10 experiment, we see in these experiments that GPFQ networks uniformly outperform MSQ networks across quantization hyperparameter choices in both top-1 and top-5 test accuracy.

1. We would like to thank Caleb Robinson for outlining this procedure in his GitHub repo found at `https://github.com/calebrob6/imagenet_validation`.

ILSVRC2012 Test Accuracy

C_α	Analog Top-1	Analog Top-5	GPFQ Top-1	GPFQ Top-5	MSQ Top-1	MSQ Top-5
2	0.7073	0.8977	0.6901	0.8892	0.68755	0.88785
3	0.7073	0.8977	**0.70075**	**0.8935**	0.69485	0.8921
4	0.7073	0.8977	0.69295	0.89095	0.66795	0.8713
5	0.7073	0.8977	0.68335	0.88535	0.53855	0.77005

Table 2: This table documents the test accuracy across 20000 images for the analog and quantized VGG16 networks on ILSVRC2012 data for the various choices of alphabet scalars C_α using the alphabet $\{-1, 0, 1\}$ and 1500 training images to learn the quantized weights.

7. Future Work

Despite all of the analysis that has gone into proving stability of quantizing the first layer of a neural network using the dynamical system (2) and isotropic Gaussian data, there are still many interesting and unanswered questions about the performance of this quantization algorithm. The above experiments suggest that our theory can be generalized to account for non-Gaussian feature data which may have hidden dependencies between them. Beyond the subspace model we consider in Lemma 16, it would be interesting to extend the results to apply in the case of a manifold structure, or clustered feature data, whose intrinsic complexities can be used to improve the upper bounds in Theorem 2 and Theorem 3. Furthermore, it would be desirable to extend the analysis to address quantizing all of the hidden layers. As we showed in the experiments, our set-up naturally extends to the case of quantizing convolutional layers. Another extension of this work might consider modifying our quantization algorithm to account for other network models like recurrent networks. Finally, we observed in Theorem 2 that the relative training error for learning the quantization decays like $\log(N_0)\sqrt{m/N_0}$. We also observed in the discussion at the end of Section 4 that when all of the feature data X_t were the same our quantization algorithm reduced to a first order greedy $\Sigma\Delta$ quantizer. Higher order $\Sigma\Delta$ quantizers in the context of oversampled finite frame coefficients and bandlimited functions are known to have quantization error which decays *polynomially* in terms of the oversampling rate. One wonders if there exist extensions of our algorithm, perhaps with a modest increase in computational complexity, that achieve faster rates of decay for the relative quantization error. We leave all of these questions for future work.

8. Proofs: Supporting Lemmata

This section presents supporting lemmata that characterize the geometry of the dynamical system (2), as well as standard results from high dimensional probability which we will use in the proof of the main technical result, Theorem 14, which appears in Section 9. Outside of the high dimensional probability results, the results of Lemmas 9, 11 and 12 consider the behavior of the dynamical system under arbitrarily distributed data.

Lemma 7 *Vershynin (2018) Let $g \sim N(0, \sigma^2)$. Then for any $\alpha > 0$*

$$\mathbb{P}\left(g \geq \alpha\right) \leq \frac{\sigma}{\alpha\sqrt{2\pi}} e^{-\frac{\alpha^2}{2\sigma^2}}.$$

Lemma 8 *Vershynin (2018) Let $g \sim N(0, I_{m\times m})$ be an m-dimensional standard Gaussian vector. Then there exists some universal constant $c_{norm} > 0$ so that for any $\alpha > 0$*

$$\mathbb{P}\left(\left|\|g\|_2 - \sqrt{m}\right| \geq \alpha\right) \leq 2e^{-c_{norm}\alpha^2}.$$

Lemma 9 *Suppose that $|w_t| < 1/2$. Then*

$$\begin{aligned}
\{X_t \in \mathbb{R}^m : q_t = 1\} &= B\left(\frac{1}{1-2w_t}u_{t-1}, \frac{1}{1-2w_t}\|u_{t-1}\|_2\right), \\
&:= B(\tilde{u}_{t-1}, \|\tilde{u}_{t-1}\|_2), \\
\{X_t \in \mathbb{R}^m : q_t = -1\} &= B\left(\frac{-1}{1+2w_t}u_{t-1}, \frac{1}{1+2w_t}\|u_{t-1}\|_2\right), \\
&:= B(\hat{u}_{t-1}, \|\hat{u}_{t-1}\|_2)
\end{aligned}$$

Proof When $q_t = 1$, (4) implies that

$$\frac{X_t^T}{\|X_t\|_2^2}u_{t-1} \geq \frac{1}{2} - w_t \iff (1-2w_t)\|X_t\|_2^2 - 2X_t^T u_{t-1} \leq 0. \tag{11}$$

Since $|w_t| < 1/2$, $1 - 2w_t > 0$. After dividing both sides of (11) by this factor, and recalling that $\tilde{u}_{t-1} := (1-2w_t)^{-1}u_{t-1}$, we may complete the square to get the equivalent inequality

$$\|X_t - \tilde{u}_{t-1}\|_2^2 \leq \|\tilde{u}_{t-1}\|_2^2.$$

An analogous argument shows the claim for the level set $\{X_t : q_t = -1\}$. ■

Remark 10 *When $w > \frac{1}{2}$ or $w < -\frac{1}{2}$, the algebra in the proof tells us that the set of $X_t's$ for $q_t = 1$ (resp. $q_t = -1$) is actually the complement of $B\left(\tilde{u}_{t-1}, \|\tilde{u}_{t-1}\|_2\right)$, (resp. the complement of $B\left(\hat{u}_{t-1}, \|\hat{u}_{t-1}\|_2\right)$). For the special case when $w_t = \pm 1/2$ these level sets are half-spaces.*

Lemma 11 *Suppose $0 < w_t < 1$, and X_t a random vector in $\mathbb{R}^m \setminus \{0\}$. Then*

$$\begin{aligned}
&\mathbb{P}_{X_t}\left((w_t - q_t)^2 + 2(w_t - q_t)\frac{X_t^T u_{t-1}}{\|X_t\|_2^2} > \alpha \middle| \mathcal{F}_{t-1}\right) \\
&= \begin{cases}
\mu_y\left(\frac{\alpha-(w_t+1)^2}{2(w_t+1)}, \frac{\alpha-(w_t-1)^2}{2(w_t-1)}\right) & \alpha < -w_t - w_t^2 \\
\mu_y\left(\frac{\alpha-w_t^2}{2w_t}, \frac{\alpha-(w_t-1)^2}{2(w_t-1)}\right) & -w_t - w_t^2 \leq \alpha \leq w_t - w_t^2 \\
0 & \alpha > w_t - w_t^2
\end{cases}
\end{aligned}$$

where μ_y is the probability measure over $\mathbb{R}$ induced by the random variable $y := \frac{X_t^T u_{t-1}}{\|X_t\|_2^2}$.

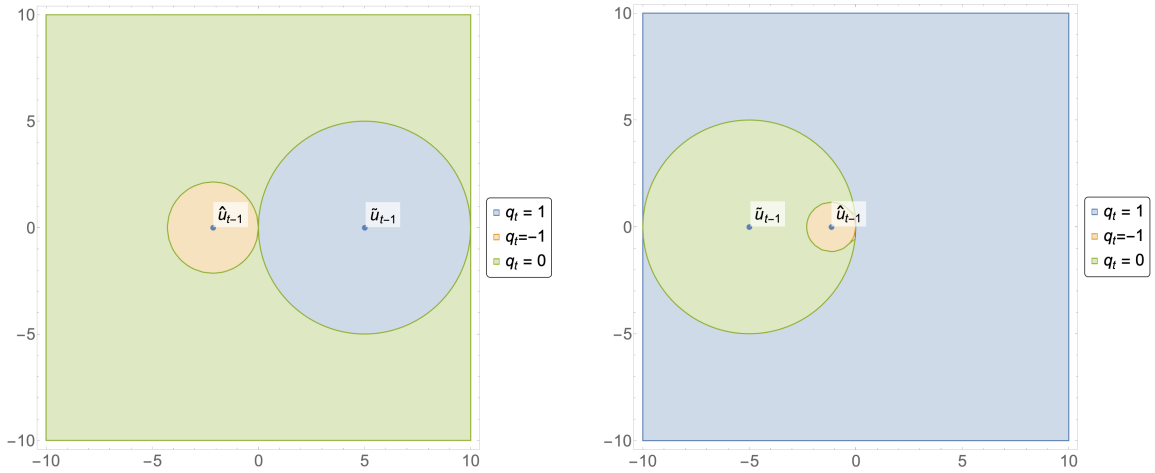


Figure 3: Visualizations of the level sets when $u_{t-1} = 3e_1$, $q_t = 1$ (blue), $q_t = -1$ (orange), and $q_t = 0$ (green) when $w_t = 0.2$ (left) and $w_t = 0.8$ (right). These are the regions that must be integrated over when calculating the moment generating function of the increment $\Delta\|u_t\|_2^2$.

Proof Let A_b denote the event that $q_t = b$ for $b \in \{-1, 0, 1\}$. Then by the law of total probability

$$\begin{aligned}
&\mathbb{P}\left((w_t - q_t)^2 + 2(w_t - q_t)y > \alpha \middle| \mathcal{F}_{t-1}\right) \\
&= \sum_{b \in \{-1,0,1\}} \mathbb{P}\left((w_t - b)^2 + 2(w_t - b)y > \alpha \text{ and } A_b \middle| \mathcal{F}_{t-1}\right).
\end{aligned}$$

Therefore, we need to look at each summand in the above sum. Well, $q_t = 0$ precisely when $-1/2 - w_t \leq y \leq 1/2 - w_t$. So we have

$$\begin{aligned}
&\mathbb{P}\left(w_t^2 + 2w_t y > \alpha \text{ and } A_0 \middle| \mathcal{F}_{t-1}\right) = \mathbb{P}\left(y > \frac{\alpha - w_t^2}{2w_t} \text{ and } -1/2 - w_t \leq y \leq 1/2 - w_t \middle| \mathcal{F}_{t-1}\right) \\
&= \begin{cases} \mu_y\left(-1/2 - w_t, 1/2 - w_t\right) & \alpha < -w_t - w_t^2 \\ \mu_y\left(\frac{\alpha - w_t^2}{2w_t}, 1/2 - w_t\right) & -w_t - w_t^2 \leq \alpha \leq w_t - w_t^2 \\ 0 & \alpha > w_t - w_t^2 \end{cases}.
\end{aligned}$$

Next, $q_t = 1$ precisely when $y > 1/2 - w_t$. Noting that $w_t - 1 < 0$, we have

$$\begin{aligned}
&\mathbb{P}\left((w_t - 1)^2 + 2(w_t - 1)y > \alpha \text{ and } A_1 \middle| \mathcal{F}_{t-1}\right) \\
&= \mathbb{P}\left(y < \frac{\alpha - (w_t - 1)^2}{2(w_t - 1)} \text{ and } y > 1/2 - w_t \middle| \mathcal{F}_{t-1}\right) \\
&= \begin{cases} \mu_y\left(1/2 - w_t, \frac{\alpha - (w_t - 1)^2}{2(w_t - 1)}\right) & \alpha \leq w_t - w_t^2 \\ 0 & \alpha > w_t - w_t^2 \end{cases}.
\end{aligned}$$

Finally, $q_t = -1$ precisely when $y < -1/2 - w_t$. So we have

$$\begin{aligned}
&\mathbb{P}\left((w_t+1)^2 + 2(w_t+1)y > \alpha \text{ and } A_{-1}\middle|\mathcal{F}_{t-1}\right) \\
&= \mathbb{P}\left(y > \frac{\alpha - (w_t+1)^2}{2(w_t+1)} \text{ and } y < -1/2 - w_t\middle|\mathcal{F}_{t-1}\right) \\
&= \begin{cases} \mu_y\left(\frac{\alpha-(w_t+1)^2}{2(w_t+1)}, -1/2 - w_t\right) & \alpha \leq -w_t - w_t^2 \\ 0 & \alpha > -w_t - w_t^2 \end{cases}.
\end{aligned}$$

Summing these three piecewise functions yields the result. ■

Lemma 12 *When $-1 < w_t < 0$, we have*

$$\begin{aligned}
&\mathbb{P}_{X_t}\left((w_t - q_t)^2 + 2(w_t - q_t)\frac{X_t^T u_{t-1}}{\|X_t\|_2^2} > \alpha\middle|\mathcal{F}_{t-1}\right) \\
&= \begin{cases} \mu_y\left(\frac{\alpha-(w_t+1)^2}{2(w_t+1)}, \frac{\alpha-(w_t-1)^2}{2(w_t-1)}\right) & \alpha < w_t - w_t^2 \\ \mu_y\left(\frac{\alpha-(w_t+1)^2}{2(w_t+1)}, \frac{\alpha-w_t^2}{2w_t}\right) & w_t - w_t^2 \leq \alpha \leq -w_t - w_t^2 \\ 0 & \alpha > -w_t - w_t^2 \end{cases}.
\end{aligned}$$

Corollary 13 *If $\|X_t\|_2^2 \leq B$ with probability 1, then $\Delta\|u_t\|_2^2 \leq B/4$ with probability 1.*

Proof Using (2), this follows from the identity

$$\Delta\|u_t\|_2^2 = \|X_t\|_2^2\left((w_t - q_t)^2 + 2(w_t - q_t)\frac{X_t^T u_{t-1}}{\|X_t\|_2^2}\right) \leq B\left((w_t - q_t)^2 + 2(w_t - q_t)\frac{X_t^T u_{t-1}}{\|X_t\|_2^2}\right).$$

Applying Lemma 11 (or Lemma 12) on the latter quantity with $\alpha = |w_t| - w_t^2$ and recognizing that $|w_t| - w_t^2 \leq 1/4$ when $w_t \in [-1, 1]$ yields the claim. ■

9. Proofs: Core Lemmata

We start by proving our main result, Theorem 14, and its extension to the case where feature vectors live in a low-dimensional subspace, Lemma 16. The proof of Theorem 14 relies on bounding the moment generating function of $\Delta\|u_t\|_2^2\middle|\mathcal{F}_{t-1}$, which in turn requires a number of results, referenced in the proof and presented thereafter. These lemmas carefully deal with bounding the above moment generating function on the events where q_t is fixed. Given u_{t-1} and $q_t = b$, Lemma 9 tells us the set of directions X_t which result in $q_t = b$ and these are the relevant events one needs to consider when bounding the moment generating function. Lemma 17 handles the case when $q_t = 0$, Lemma 18 handles the case when $q_t = 1$, and Lemma 19 handles the case when $q_t = -1$.

Theorem 14 *Suppose that for $t \in \mathbb{N}$, the vectors $X_t \sim \mathcal{N}(0, \sigma^2 I_{m\times m})$ are independent and that $w_t \in [-1,1]$ are i.i.d. and independent of X_t, and define the event*

$$A_\varepsilon := \{\mathrm{dist}(w_t, \{-1,0,1\}) < \varepsilon\}.$$

Then there exist positive constants c_{norm}, C_λ, and C_{sup}, such that with $\lambda := \frac{C_\lambda}{C_{\sup}^2 \sigma^2 m \log(N_0)}$, and $\rho, \varepsilon \in (0,1)$ satisfying $\tilde{\rho} := \rho + e^{C_\lambda/4}\mathbb{P}(A_\varepsilon) < 1$, the iteration (3) satisfies

$$\mathbb{P}\left(\|u_t\|_2^2 > \alpha\right) \le \tilde{\rho}^t e^{-\lambda\alpha} + \frac{1-\tilde{\rho}^t}{1-\tilde{\rho}} e^{\frac{C_\lambda}{4} + \lambda(\beta-\alpha)} + 2e^{-\log(N_0)(c_{norm}C_{\sup}^2 m - 1)}. \tag{12}$$

Above, $C > 0$ is a universal constant and $\beta := C\frac{e^{8C_\lambda}\sigma^2 m^2 \log^2(N_0)}{\rho^2\varepsilon^2}$.

Proof The proof technique is inspired by Hajek (1982). Define the events

$$U_t := \left\{ \sup_{j\in\{1,\dots,t\}} \|X_j\|_2 \le C_{\sup}\sigma\sqrt{m}\left(\sqrt{\log(N_0)}+1\right)\right\}.$$

Using a union bound and Lemma 8, we see that $U_{N_0}^C$ happens with low probability since

$$\begin{aligned}&\mathbb{P}\left(\sup_{t\in[N_0]} \|X_t\|_2 > C_{\sup}\sigma\sqrt{m}\left(\sqrt{\log(N_0)}+1\right)\right)\\ &\le 2N_0 e^{-c_{norm}C_{\sup}^2 m\log(N_0)} = 2e^{-\log(N_0)(c_{norm}C_{\sup}^2 m-1)}.\end{aligned}$$

We can therefore bound the probability of interest with appropriate conditioning.

$$\mathbb{P}\left(\|u_t\|_2^2 \ge \alpha\right) \le \mathbb{P}\left(\|u_t\|_2^2 \ge \alpha \Big| U_{N_0}\right)\mathbb{P}(U_{N_0}) + \mathbb{P}(U_{N_0}^C).$$

Looking at the first summand, for any $\lambda > 0$, we have by Markov's inequality

$$\begin{aligned}\mathbb{P}\left(\|u_t\|_2^2 \ge \alpha \Big| U_{N_0}\right)\mathbb{P}(U_{N_0}) &\le e^{-\lambda\alpha}\mathbb{E}[e^{\lambda\|u_t\|_2^2}\Big|U_{N_0}]\mathbb{P}(U_{N_0})\\ &= e^{-\lambda\alpha}\mathbb{E}[e^{\lambda\|u_t\|_2^2}\mathbb{1}_{U_{N_0}}]\\ &= e^{-\lambda\alpha}\mathbb{E}\left[e^{\lambda\|u_{t-1}\|_2^2}e^{\lambda\Delta\|u_t\|_2^2}\mathbb{1}_{U_{N_0}}\right]\\ &= e^{-\lambda\alpha}\mathbb{E}\left[\mathbb{E}\left[e^{\lambda\|u_{t-1}\|_2^2}e^{\lambda\Delta\|u_t\|_2^2}\mathbb{1}_{U_{N_0}}\Big|\mathcal{F}_{t-1}\right]\right].\end{aligned}$$

We expand the conditional expectation given the filtration into a sum of two parts

$$\begin{aligned}\mathbb{E}\left[e^{\lambda\|u_{t-1}\|_2^2}e^{\lambda\Delta\|u_t\|_2^2}\mathbb{1}_{U_{N_0}}\Big|\mathcal{F}_{t-1}\right] &= \mathbb{E}\left[e^{\lambda\|u_{t-1}\|_2^2}e^{\lambda\Delta\|u_t\|_2^2}\mathbb{1}_{U_{N_0}}\mathbb{1}_{A_\varepsilon^C \text{ and } \|u_{t-1}\|_2^2\ge\beta}\Big|\mathcal{F}_{t-1}\right]\\ &\quad+\mathbb{E}\left[e^{\lambda\|u_{t-1}\|_2^2}e^{\lambda\Delta\|u_t\|_2^2}\mathbb{1}_{U_{N_0}}\mathbb{1}_{A_\varepsilon \text{ or } \|u_{t-1}\|_2^2<\beta}\Big|\mathcal{F}_{t-1}\right].\end{aligned}$$

Towering expectations, the expectation over X_t of the first summand is bounded above by $\rho e^{\lambda\|u_{t-1}\|_2^2}\mathbb{1}_{U_{t-1}}$ for all w_t on the event A_ε^C using Lemmas 17, 18, and 19. Therefore, the same bound is also true for the expectation over w_t. As for the second term, we have

$$\begin{aligned}&\mathbb{E}\left[e^{\lambda\|u_{t-1}\|_2^2}e^{\lambda\Delta\|u_t\|_2^2}\mathbb{1}_{U_{N_0}}\mathbb{1}_{A_\varepsilon \text{ or } \|u_{t-1}\|_2^2<\beta}\Big|\mathcal{F}_{t-1}\right] =\\ &\mathbb{E}\left[e^{\lambda\|u_{t-1}\|_2^2}e^{\lambda\Delta\|u_t\|_2^2}\mathbb{1}_{U_{N_0}}\mathbb{1}_{\|u_{t-1}\|_2^2<\beta}\Big|\mathcal{F}_{t-1}\right] + \mathbb{E}\left[e^{\lambda\|u_{t-1}\|_2^2}e^{\lambda\Delta\|u_t\|_2^2}\mathbb{1}_{U_{N_0}}\mathbb{1}_{A_\varepsilon \text{ and } \|u_{t-1}\|_2^2\ge\beta}\Big|\mathcal{F}_{t-1}\right]\end{aligned}$$

For both terms, we can use the uniform bound on the increments as proven in Corollary 13. The first term we can bound by $e^{\lambda C_{\sup}^2 \sigma^2 m \log(N_0)/4} e^{\lambda\beta} \leq e^{C_\lambda/4} e^{\lambda\beta}$. As for the second, expecting over the draw of w_t gives us

$$\begin{aligned}\mathbb{E}\left[e^{\lambda\|u_{t-1}\|_2^2} e^{\lambda\Delta\|u_t\|_2^2} \mathbb{1}_{U_{N_0}} \mathbb{1}_{A_\varepsilon \text{ and } \|u_{t-1}\|_2^2 \geq \beta} \middle| \mathcal{F}_{t-1}\right] &\leq e^{\lambda C_{\sup}^2 \sigma^2 m \log(N_0)/4} e^{\lambda\|u_{t-1}\|_2^2} \mathbb{1}_{U_{t-1}} \mathbb{P}(A_\varepsilon) \\ &\leq e^{C_\lambda/4} e^{\lambda\|u_{t-1}\|_2^2} \mathbb{1}_{U_{t-1}} \mathbb{P}(A_\varepsilon).\end{aligned}$$

Therefore, we have

$$\begin{aligned}&\mathbb{P}\left(\|u_t\|_2^2 > \alpha\right) \\ &\leq e^{-\lambda\alpha}\left(\left(\rho + e^{C_\lambda/4}\mathbb{P}\left(A_\varepsilon\right)\right)\mathbb{E}[e^{\lambda\|u_{t-1}\|_2^2}\mathbb{1}_{U_{t-1}}] + e^{\lambda\beta + C_\lambda/4}\right) + 2e^{-\log(N_0)(c_{norm}C_{\sup}^2 m - 1)} \\ &= e^{-\lambda\alpha}\left(\tilde{\rho}\mathbb{E}[e^{\lambda\|u_{t-1}\|_2^2}\mathbb{1}_{U_{t-1}}] + e^{\lambda\beta + C_\lambda/4}\right) + 2e^{-\log(N_0)(c_{norm}C_{\sup}^2 m - 1)}.\end{aligned}$$

Proceeding inductively on $\mathbb{E}[e^{\lambda\|u_{t-1}\|_2^2}]$ yields the claim. ■

Remark 15 *To simplify the bound in* (12)*, assuming we have* $\tilde{\rho}, \varepsilon \propto 1$ *and* $\alpha \gtrsim \beta \propto \sigma^2 m^2 \log^2(N_0)$ *we have*

$$\begin{aligned}\mathbb{P}\left(\|u_t\|_2^2 \geq \alpha\right) &\leq e^{-\lambda\alpha} + e^{\frac{C_\lambda}{4} + \lambda(\beta - \alpha)} + 2e^{-\log(N_0)(c_{norm}C_{\sup}^2 m - 1)}, \\ &= e^{-\frac{C_\lambda m \log(N_0)}{C_{\sup}^2}} + e^{\frac{C_\lambda}{4} - C'\frac{C_\lambda m \log(N_0)}{C_{\sup}^2}} + 2e^{-\log(N_0)(c_{norm}C_{\sup}^2 m - 1)}, \\ &\leq Ce^{-cm\log(N_0)}.\end{aligned}$$

This matches the bound on the probability of failure we give in Theorem 2.

Lemma 16 *Suppose* $X = ZA$ *where* $Z \in \mathbb{R}^{m\times d}$ *satisfies* $Z^T Z = I$*, and* $A \in \mathbb{R}^{d\times N_0}$ *has i.i.d.* $\mathcal{N}(0, \sigma^2)$ *entries. In other words, suppose the feature data* X_t *are Gaussians drawn from a* d*-dimensional subspace of* $\mathbb{R}^m$*. Then with the remaining hypotheses as Theorem 14 we have with probability at least* $1 - Ce^{-cd\log(N_0)} - 3\exp(-c''d)$

$$\|Xw - Xq\|_2 \lesssim \sigma d \log(N_0).$$

Proof We will show that running the dynamical system (2) with X_t is equivalent to running a modified version of (2) with the columns of A, denoted A_t. Then we can apply the result of Theorem 14. By definition, we have

$$\begin{aligned}u_0 &:= 0 \in \mathbb{R}^m, \\ q_t &:= \mathcal{Q}\left(w_t + \frac{X_t^T u_{t-1}}{\|X_t\|_2^2}\right), \\ u_t &:= u_{t-1} + w_t X_t - q_t X_t.\end{aligned}$$

In anticipation of subsequent applications of change of variables, let $Z = U\Sigma V^T \in \mathbb{R}^{m\times d}$ be the singular value decomposition of Z where $U \in \mathbb{R}^{m\times m}$ and $V \in \mathbb{R}^{d\times d}$ are orthogonal matrices and $\Sigma \in \mathbb{R}^{m\times d}$ decomposes as

$$\Sigma = \begin{bmatrix} I_{d\times d} \\ 0 \end{bmatrix}.$$

Since u_t is a linear combination of $X_1, \dots, X_t$ for all t it follows that u_t is in the column space of Z. In other words, $u_t = Z(Z^TZ)^{-1}Z^Tu_t := Z\eta_t$. We may rewrite the above dynamical system in terms of A_t, η_t as

$$\begin{aligned} u_0 &:= 0 \in \mathbb{R}^m, \\ q_t &:= \mathcal{Q}\left(w_t + \frac{A_t^TZ^TZ\eta_{t-1}}{\|ZA_t\|_2^2}\right) \\ &= \mathcal{Q}\left(w_t + \frac{A_t^T\eta_{t-1}}{\|A_t\|_2^2}\right) \\ Z\eta_t &:= Z\eta_{t-1} + w_tZA_t - q_tZA_t \\ \iff \eta_t &= \eta_{t-1} + w_tA_t - q_tA_t \end{aligned}$$

So, in other words, we've reduced to running (2) but now with the state variables $\eta_{t-1} \in \mathbb{R}^d$ in place of u_{t-1} and with A_t in place of X_t. Applying the result of Theorem 14 yields the claim. ∎

Lemma 17 *Let $X_t \sim \mathcal{N}(0, \sigma^2 I_{m\times m})$ and $\operatorname{dist}(w_t, \{-1, 0, 1\}) \geq \varepsilon$. Define the event*

$$U := \left\{\|X_t\|_2 \leq C_{\sup}\sigma\sqrt{m\log(N_0)}\right\},$$

and set $\lambda := \frac{C_\lambda}{C_{\sup}^2\sigma^2 m\log(N_0)}$, where $C_\lambda \in (0, \frac{c_{norm}}{12})$ is some constant and c_{norm} is as in Lemma 8. Then there exists a universal constant $C > 0$ so that with $\beta := \frac{Ce^{C_\lambda}\sigma^2 m\log(N_0)}{\rho\varepsilon\sigma}$

$$\mathbb{E}\left[e^{\lambda\Delta\|u_t\|_2^2}\mathbb{1}_{q_t=0}\mathbb{1}_U\mathbb{1}_{\|u_{t-1}\|_2\geq\beta}\middle|\mathcal{F}_{t-1}\right] \leq \rho.$$

Proof Recall $\Delta\|u_t\|_2^2 = \|u_t\|_2^2 - \|u_{t-1}\|_2^2 = (w_t - q_t)^2\|X_t\|_2^2 + 2(w_t - q_t)\langle X_t, u_{t-1}\rangle$. Let us first consider the case when $|w_t| < \frac{1}{2}$. We will further assume that $w_t > 0$, since there is the symmetry between $\hat{u}_{t-1}$ and $\tilde{u}_{t-1}$ under the mapping $w_t \to -w_t$. Before embarking on our calculus journey, let us make some key remarks. First, on the event U, we can bound the increment above by $\Delta\|u_t\|_2^2 \leq (w_t - q_t)^2C_{\sup}^2\sigma^2 m\log(N_0) + 2(w_t - q_t)\langle X_t, u_{t-1}\rangle$. So, it behooves us to find an upper bound for $\mathbb{E}\left[e^{2\lambda w_t\langle X_t, u_{t-1}\rangle}\mathbb{1}_U\mathbb{1}_{q_t=0}\mathbb{1}_{\|u_{t-1}\|_2\geq\beta}|\mathcal{F}_{t-1}\right]$. Since the exponential function is non-negative, we can always upper bound this expectation by removing the indicator on U. In other words,

$$\mathbb{E}\left[e^{2\lambda w_t\langle X_t, u_{t-1}\rangle}\mathbb{1}_U\mathbb{1}_{q_t=0}\mathbb{1}_{\|u_{t-1}\|_2\geq\beta}\middle|\mathcal{F}_{t-1}\right] \leq \mathbb{E}\left[e^{2\lambda w_t\langle X_t, u_{t-1}\rangle}\mathbb{1}_{q_t=0}\mathbb{1}_{\|u_{t-1}\|_2\geq\beta}\middle|\mathcal{F}_{t-1}\right]. \quad (13)$$

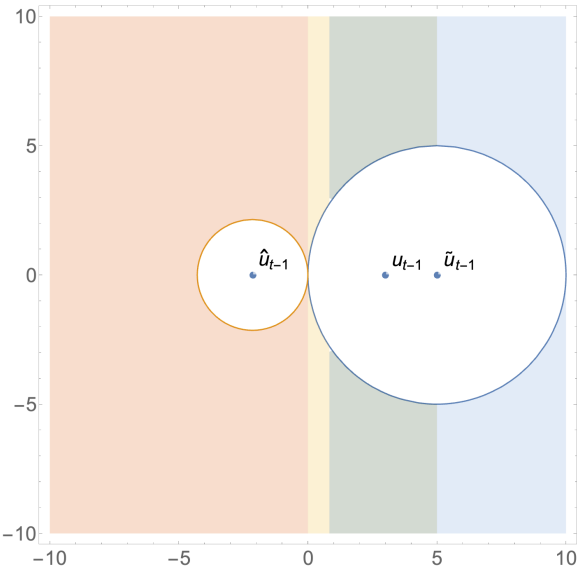


Figure 4: Plotted above is a figure depicting the various regions of integration involved in the derivation of the upper bound for Lemma 17 for the particular case when $w_t = 0.3$ and $u_t = 3e_1$. Moving from left to right, the region in red corresponds to equation (14), the region in yellow to region R as in equation (20), the region in green to region S as in equation (26), and the region in blue to region T as in equation (23).

Since we're indicating on an event where $\|u_{t-1}\|_2 \geq \beta$, we will need to handle the events where $\langle X_t, u_{t-1}\rangle > 0$ with some care, since without an *a priori* upper bound on $\|u_{t-1}\|$ the moment generating function restricted to this event could explode. Therefore, we'll divide the region of integration into 4 pieces which are depicted in Figure 4. Because of the abundance of notation in the following arguments, we will denote $\mathbb{1}_\beta := \mathbb{1}_{\|u_{t-1}\|_2 \geq \beta}$.

Let's handle the easier event first, namely where $\langle X_t, u_{t-1}\rangle \leq 0$. Here, we have

$$\mathbb{E}\left[e^{2\lambda\langle X_t, u_{t-1}\rangle}\mathbb{1}_\beta \mathbb{1}_{q_t=0}\mathbb{1}_{\langle X_t, u_{t-1}\rangle<0}\middle|\mathcal{F}_{t-1}\right] =$$
$$(2\pi\sigma^2)^{-m/2}\mathbb{1}_\beta \int_{B(\hat{u}_{t-1}, \|\hat{u}_{t-1}\|)^C \cap \{\langle x, u_{t-1}\rangle \leq 0\}} e^{2\lambda w_t \langle x, u_{t-1}\rangle} e^{\frac{-1}{2\sigma^2}\|x\|_2^2}\, dx. \tag{14}$$

By rotational invariance, we may assume without loss of generality that $u_{t-1} = \|u_{t-1}\|_2 e_1$, where $e_1 \in \mathbb{R}^m$ is the first standard basis vector. In that case, the constraint $\langle X_t, u_{t-1}\rangle < 0$ is equivalent to $X_{t,1} < 0$, where $X_{t,1}$ is the first component of X_t. Using Lemma 9, it follows that the set of X_t for which $q_t = 0$ and $X_{t,1} < 0$ is simply $\{x \in \mathbb{R}^m : x_1 \leq 0\} \cap B\left(-\|\hat{u}_{t-1}\|_2 e_1, \|\hat{u}_{t-1}\|_2\right)^C$, where the negative sign here comes from the fact that $\hat{u}_{t-1} = -(1+2w_t)u_{t-1}$. That means we can rewrite (14) as

$$(2\pi\sigma^2)^{-m/2}\mathbb{1}_\beta \int_{B(-\|\hat{u}_{t-1}\|_2 e_1, \|\hat{u}_{t-1}\|)^C \cap \{x_1 \leq 0\}} e^{2\lambda w_t \|u_{t-1}\| x_1 - \frac{x_1^2}{2\sigma^2}} e^{\frac{-1}{2\sigma^2}\sum_{j\geq 2} x_j^2}\, dx.$$

Perhaps surprisingly, we can afford to use the crude upper bound on this integral by simply removing the constraint that $x \in B(-\|\hat{u}_{t-1}\|_2 e_1, \|\hat{u}_{t-1}\|)^C$. Iterating the univariate integrals then gives us

$$\begin{aligned}
&(2\pi\sigma^2)^{-m/2}\mathbb{1}_\beta \int_{B(-\|\hat{u}_{t-1}\|_2 e_1, \|\hat{u}_{t-1}\|)^C \cap \{x_1 \leq 0\}} e^{2\lambda w_t \|u_{t-1}\| x_1 - \frac{x_1^2}{2\sigma^2}} e^{\frac{-1}{2\sigma^2}\sum_{j\geq 2} x_j^2}\, dx \\
&\leq (2\pi\sigma^2)^{-1/2}\mathbb{1}_\beta \int_{-\infty}^0 e^{2\lambda w_t \|u_{t-1}\| x_1 - \frac{x_1^2}{2\sigma^2}}\, dx_1 \int_{\mathbb{R}^{m-1}} (2\pi\sigma^2)^{-\frac{m-1}{2}} e^{\frac{-1}{2\sigma^2}\sum_{j\geq 2} x_j^2}\, dx_2 \ldots dx_m \\
&= (2\pi\sigma^2)^{-1/2}\mathbb{1}_\beta \int_{-\infty}^0 e^{2\lambda w_t \|u_{t-1}\| x_1 - \frac{x_1^2}{2\sigma^2}}\, dx_1 = (2\pi\sigma^2)^{-1/2}\mathbb{1}_\beta \int_0^\infty e^{-2\lambda w_t \|u_{t-1}\| x_1 - \frac{x_1^2}{2\sigma^2}}\, dx_1.
\end{aligned} \tag{15}$$

We complete the square and use a change of variables to reformulate (15) as

$$\begin{aligned}
&(2\pi\sigma^2)^{-1/2}\mathbb{1}_\beta e^{2\sigma^2\lambda^2 w_t^2 \|u_{t-1}\|_2^2} \int_0^\infty e^{-\frac{1}{2\sigma^2}\left(x_1 + 2\sigma^2\lambda w_t \|u_{t-1}\|_2\right)^2}\, dx_1 \\
&= (2\pi\sigma^2)^{-1/2}\mathbb{1}_\beta e^{2\sigma^2\lambda^2 w_t^2 \|u_{t-1}\|_2^2} \int_{2\sigma^2\lambda w_t \|u_{t-1}\|_2}^\infty e^{-\frac{x_1^2}{2\sigma^2}}\, dx_1.
\end{aligned} \tag{16}$$

Since the lower limit of integration is positive and large when $\|u_{t-1}\|_2$ is, we can use a tail bound as in Lemma 7 to upper bound (16) by

$$\frac{\mathbb{1}_\beta \sigma}{2\sigma^2\lambda w_t \|u_{t-1}\|_2\sqrt{2\pi}} \leq \frac{\mathbb{1}_\beta \sigma}{\sigma^2\lambda\varepsilon\|u_{t-1}\|_2} = \frac{\mathbb{1}_\beta C_{\sup}^2 \sigma m \log(N_0)}{C_\lambda \varepsilon \|u_{t-1}\|_2}, \tag{17}$$

where the first inequality follows from $|w_t| \geq \varepsilon$ and the equality follows from $\lambda = \frac{C_\lambda}{C_{\sup}^2 \sigma^2 m \log(N_0)}$.

Now we handle the moment generating function on the event that $\langle X_t, u_{t-1}\rangle \geq 0$. Again, using rotational invariance to assume $u_{t-1} = \|u_{t-1}\|_2 e_1$, we have by Lemma 9 that the event to integrate over is $\{x \in \mathbb{R}^m : x_1 \geq 0\} \cap B\left(\|\tilde{u}_{t-1}\|_2 e_1, \|\tilde{u}_{t-1}\|_2\right)^C$. Notice that iterating the integrals gives us

$$\begin{aligned}
&\mathbb{E}\left[e^{2\lambda\langle X_t, u_{t-1}\rangle}\mathbb{1}_\beta \mathbb{1}_{q_t=0}\mathbb{1}_{\langle X_t, u_{t-1}\rangle \geq 0}\middle|\mathcal{F}_{t-1}\right] \\
&\quad = (2\pi\sigma^2)^{-m/2}\mathbb{1}_\beta \int_{B(\tilde{u}_{t-1}, \|\tilde{u}_{t-1}\|)^C \cap \{x_1 \geq 0\}} e^{2\lambda w_t \|u_{t-1}\| x_1 - \frac{1}{2\sigma^2}\|x\|_2^2}\, dx \\
&\quad = (2\pi\sigma^2)^{-1/2}\mathbb{1}_\beta \int_0^\infty e^{2\lambda w_t \|u_{t-1}\| x_1 - \frac{x_1^2}{2\sigma^2}} \int_{B\left(0, \sqrt{(2x_1\|\tilde{u}_{t-1}\|_2 - x_1^2)^+}\right)^C} (2\pi\sigma^2)^{-\frac{m-1}{2}} e^{-\frac{1}{2\sigma^2}\sum_{j=2}^m x_j^2}\, dx_2 \ldots dx_m dx_1, \quad (18)
\end{aligned}$$

with the notation $(z)^+ = \max\{z, 0\}$ for $z \in \mathbb{R}$. Consequentially, we can rephrase (18) into a more probabilistic statement. Below, let $\gamma_j \sim \mathcal{N}(0,1)$ denote i.i.d. standard normal random variables. Then (18) is equal to

$$(2\pi\sigma^2)^{-1/2}\mathbb{1}_\beta \int_0^\infty e^{2\lambda w_t \|u_{t-1}\| x_1 - \frac{x_1^2}{2\sigma^2}} \mathbb{P}\left(\sigma^2 \sum_{j=1}^{m-1} \gamma_j^2 \geq 2x_1\|\tilde{u}_{t-1}\|_2 - x_1^2\right) dx_1. \quad (19)$$

The probability appearing in (19) will decay exponentially provided $2x_1\|\tilde{u}_{t-1}\|_2 - x_1^2$ is sufficiently large. To that end, we will divide up this half-space into the following regions. Let $C_0 \geq 16$ be a constant and define the sets $R := \{x \in \mathbb{R}^m : 0 \leq x_1 \leq \frac{C_0\sigma^2 m}{\|\tilde{u}_{t-1}\|_2}\}$, $S := \{x \in \mathbb{R}^m : \frac{C_0\sigma^2 m}{\|\tilde{u}_{t-1}\|_2} \leq x_1 \leq \|\tilde{u}_{t-1}\|_2\}$, and $T := \{x \in \mathbb{R}^m : \|\tilde{u}_{t-1}\|_2 \leq x_1\}$. Figure 4 gives a visual depiction of this decomposition. Then we have

$$\begin{aligned}
\mathbb{E}\left[e^{2\lambda\langle X_t, u_{t-1}\rangle}\mathbb{1}_\beta \mathbb{1}_{q_t=0}\mathbb{1}_{\langle X_t, u_{t-1}\rangle \geq 0}\middle|\mathcal{F}_{t-1}\right] &= (2\pi\sigma^2)^{-m/2}\mathbb{1}_\beta \int_{B(\tilde{u}_{t-1}, \|\tilde{u}_{t-1}\|)^C \cap R} e^{2\lambda w_t \|u_{t-1}\| x_1 - \frac{1}{2\sigma^2}\|x\|_2^2}\, dx \\
&\quad + (2\pi\sigma^2)^{-m/2}\mathbb{1}_\beta \int_{B(\tilde{u}_{t-1}, \|\tilde{u}_{t-1}\|)^C \cap S} e^{2\lambda w_t \|u_{t-1}\| x_1 - \frac{1}{2\sigma^2}\|x\|_2^2}\, dx \\
&\quad + (2\pi\sigma^2)^{-m/2}\mathbb{1}_\beta \int_{B(\tilde{u}_{t-1}, \|\tilde{u}_{t-1}\|)^C \cap T} e^{2\lambda w_t \|u_{t-1}\| x_1 - \frac{1}{2\sigma^2}\|x\|_2^2}\, dx.
\end{aligned}$$

For the integral over R, we will use the naïve upper bound

$$\mathbb{P}\left(\sigma^2 \sum_{j=1}^{m-1} \gamma_j^2 \geq 2x_1\|\tilde{u}_{t-1}\|_2 - x_1^2\right) \leq 1.$$

This gives us

$$\begin{aligned}
&(2\pi\sigma^2)^{-m/2}\mathbb{1}_\beta \int_{B(\tilde{u}_{t-1},\|\tilde{u}_{t-1}\|)^C\cap R} e^{2\lambda w_t\|u_{t-1}\|x_1-\frac{1}{2\sigma^2}\|x\|_2^2}\,dx \\
&\leq (2\pi\sigma^2)^{-1/2}\mathbb{1}_\beta \int_0^{\frac{C_0\sigma^2 m}{\|\tilde{u}_{t-1}\|_2}} e^{2\lambda w_t\|u_{t-1}\|x_1-\frac{1}{2\sigma^2}x_1^2}\,dx_1 \\
&= (2\pi\sigma^2)^{-1/2}\mathbb{1}_\beta e^{2\lambda^2\sigma^2 w_t^2\|u_{t-1}\|_2^2} \int_{-2\lambda w_t\sigma^2\|u_{t-1}\|_2}^{\frac{C_0\sigma^2 m}{\|\tilde{u}_{t-1}\|_2}-2\lambda w_t\sigma^2\|u_{t-1}\|_2} e^{-\frac{1}{2\sigma^2}x_1^2}\,dx_1. \qquad (20)
\end{aligned}$$

The upper limit of integration is negative since $\|u_{t-1}\|_2^2 \geq \frac{3C_0C_{\sup}^2\sigma^2 m\log(N_0)}{2C_\lambda\varepsilon} \geq \frac{C_0|1-2w_t|}{2\lambda w_t}$. Under this assumption, we can upper bound the integral with a Riemann sum. As the maximum of the integrand occurs at the upper limit of integration, we bound (20) with

$$(2\pi\sigma^2)^{-1/2}\mathbb{1}_\beta \frac{e^{\frac{-1}{2\sigma^2}\left(\frac{C_0^2\sigma^4 m^2}{\|\tilde{u}_{t-1}\|_2^2}-\frac{4C_0\sigma^4 m\lambda w_t\|u_{t-1}\|_2}{\|\tilde{u}_{t-1}\|_2}\right)}C_0\sigma^2 m}{\|\tilde{u}_{t-1}\|_2}. \qquad (21)$$

Recognizing that $\frac{\|u_{t-1}\|_2}{\|\tilde{u}_{t-1}\|_2} = |1-2w_t| \leq 3$ and recalling that $\lambda = \frac{C_\lambda}{C_{\sup}^2\sigma^2 m\log(N_0)}$ we can further upper bound by

$$\frac{\mathbb{1}_\beta e^{2C_0\lambda\sigma^2 m w_t|1-2w_t|}C_0\sigma m}{\|\tilde{u}_{t-1}\|_2\sqrt{2\pi}} \leq \frac{\mathbb{1}_\beta 3C_0 e^{\frac{6C_0C_\lambda}{C_{\sup}^2\log(N_0)}}\sigma m}{\|u_{t-1}\|_2}. \qquad (22)$$

As was the case for R, we can use the bound $\mathbb{P}\left(\sigma^2\sum_{j=1}^{m-1}\gamma_j^2 \geq 2x_1\|\tilde{u}_{t-1}\|_2 - x_1^2\right) \leq 1$ over T too. Completing the square in the exponent as we usually do gives us

$$\begin{aligned}
&(2\pi\sigma^2)^{-m/2}\mathbb{1}_\beta \int_{B(\tilde{u}_{t-1},\|\tilde{u}_{t-1}\|)^C\cap T} e^{2\lambda w_t\|u_{t-1}\|x_1-\frac{1}{2\sigma^2}\|x\|_2^2}\,dx \\
&\leq (2\pi\sigma^2)^{-1/2}\mathbb{1}_\beta e^{2\lambda^2 w_t^2\sigma^2\|u_{t-1}\|_2^2} \int_{\|\tilde{u}_{t-1}\|_2-2\lambda w_t\sigma^2\|u_{t-1}\|_2}^{\infty} e^{\frac{-x_1^2}{2\sigma^2}}\,dx_1. \qquad (23)
\end{aligned}$$

Since $\lambda < \frac{1}{6\sigma^2} \leq \frac{1}{2\sigma^2 w_t|1-2w_t|}$ the lower limit of integration is positive, so we can use a Gaussian tail bound as in Lemma 7 to bound (23) by

$$\begin{aligned}
&\frac{\mathbb{1}_\beta\sigma}{\sqrt{2\pi}\left(\|\tilde{u}_{t-1}\|_2-2\lambda w_t\sigma^2\|u_{t-1}\|_2\right)} e^{\frac{-1}{2\sigma^2}\left(\|\tilde{u}_{t-1}\|_2^2-4\lambda w_t\sigma^2\|u_{t-1}\|_2\|\tilde{u}_{t-1}\|_2\right)} \\
&= \frac{\mathbb{1}_\beta\sigma}{\sqrt{2\pi}\left(\|\tilde{u}_{t-1}\|_2-2\lambda w_t\sigma^2\|u_{t-1}\|_2\right)} e^{\frac{-\|u_{t-1}\|_2^2}{2\sigma^2}\left(\frac{1}{|1-2w_t|^2}-\frac{4\lambda w_t\sigma^2}{|1-2w_t|}\right)}. \qquad (24)
\end{aligned}$$

As $\lambda < \frac{1}{12\sigma^2} \leq \frac{1}{4w_t\sigma^2|1-2w_t|}$ the exponent appearing in (24) is negative. Bounding the exponential by 1 then gives us the upper bound

$$\begin{aligned}
\frac{\mathbb{1}_\beta\sigma}{\sqrt{2\pi}\left(\|\tilde{u}_{t-1}\|_2-2\lambda w_t\sigma^2\|u_{t-1}\|_2\right)} &= \frac{\mathbb{1}_\beta\sigma}{\|u_{t-1}\|_2\left(\frac{1}{|1-2w_t|}-\frac{2w_tC_\lambda\sigma^2}{C_{\sup}^2\sigma^2 m\log(N_0)}\right)} \\
&\leq \frac{\mathbb{1}_\beta\sigma}{\|u_{t-1}\|_2\left(\frac{1}{3}-\frac{2C_\lambda}{C_{\sup}^2 m\log(N_0)}\right)}. \qquad (25)
\end{aligned}$$

Now, for S we can use the exponential decay of the probability appearing in (19). To make the algebra a bit nicer, we can upper-bound this probability by $\mathbb{P}\left(\sigma^2 \sum_{j=1}^{m-1} \gamma_j^2 \geq x_1 \|\tilde{u}_{t-1}\|\right)$ since on S we have $0 \leq x_1 \leq \|\tilde{u}_{t-1}\|_2$. Setting $\nu := \frac{1}{\sigma\sqrt{m-1}}\sqrt{x_1\|\tilde{u}_{t-1}\|}$, Lemma 8 tells us for $x_1 \geq \frac{C_0\sigma^2 m}{\|\tilde{u}_{t-1}\|}$

$$\mathbb{P}\left(\sqrt{\sum_{j=1}^{m-1}\gamma_j^2} \geq \sqrt{m-1}\nu\right) \leq 2\exp(-c_{norm}(\nu-1)^2(m-1)).$$

To simplify our algebra, we remark that for any $c > 0$,

$$e^{-c(m-1)(z-1)^2} \leq e^{\frac{-c}{2}(m-1)z^2},$$

provided $z \geq 4$. By our choice of C_0, this happens to be the case on S, as $\frac{C_0\sigma^2 m}{\|\tilde{u}_{t-1}\|} \leq x_1 \leq \|\hat{u}_{t-1}\|$ and so

$$\nu^2 \geq \frac{x_1\|\tilde{u}_{t-1}\|_2}{\sigma^2 m} \geq C_0.$$

This gives us the upper bound on the probability

$$\begin{aligned}\mathbb{P}\left(\sigma^2\sum_{j=1}^{m-1}\gamma_j^2 \geq 2x_1\|\tilde{u}_{t-1}\|_2 - x_1^2\right) &\leq 2\exp(-c_{norm}(m-1)\nu^2/2)\\ &= 2\exp\left(\frac{-c_{norm}x_1\|\tilde{u}_{t-1}\|_2}{2\sigma^2}\right).\end{aligned}$$

Consequentially, we can bound the integral over S as follows

$$\begin{aligned}&(2\pi\sigma^2)^{-1/2}\mathbb{1}_\beta \int_{B(\tilde{u}_{t-1},\|\tilde{u}_{t-1}\|)^C\cap S} e^{2\lambda w_t\|u_{t-1}\|x_1 - \frac{x_1^2}{2\sigma^2}}\mathbb{P}\left(\sigma^2\sum_{j=1}^{m-1}\gamma_j^2 \geq 2x_1\|\tilde{u}_{t-1}\|_2 - x_1^2\right)\, dx_1\\ &\leq 2\cdot(2\pi\sigma^2)^{-1/2}\mathbb{1}_\beta\int_{\frac{C_0\sigma^2 m}{\|\tilde{u}_{t-1}\|}}^{\|\tilde{u}_{t-1}\|} e^{2\lambda w_t\|u_{t-1}\|x_1 - \frac{x_1^2}{2\sigma^2} - \frac{c_{norm}x_1\|\hat{u}_{t-1}\|_2}{2\sigma^2}}\, dx_1\\ &= 2\cdot(2\pi\sigma^2)^{-1/2}\mathbb{1}_\beta\int_{\frac{C_0\sigma^2 m}{\|\tilde{u}_{t-1}\|}}^{\|\tilde{u}_{t-1}\|} e^{\left(2\lambda w_t\|u_{t-1}\|_2 - \frac{c_{norm}\|\tilde{u}_{t-1}\|}{2\sigma^2}\right)x_1 - \frac{x_1^2}{2\sigma^2}}\, dx_1. \qquad (26)\end{aligned}$$

Setting $2\zeta := c_{norm}\|\tilde{u}\|_2 - 4\lambda\sigma^2 w_t\|u_{t-1}\|$, we have that (26) is equal to

$$\begin{aligned}2\cdot(2\pi\sigma^2)^{-1/2}\mathbb{1}_\beta\int_{\frac{C_0\sigma^2 m}{\|\tilde{u}_{t-1}\|}}^{\|\tilde{u}_{t-1}\|} e^{\frac{-2\zeta x_1}{2\sigma^2} - \frac{x_1^2}{2\sigma^2}}\, dx_1 &= 2\cdot(2\pi\sigma^2)^{-1/2}\mathbb{1}_\beta e^{\frac{\zeta^2}{2\sigma^2}}\int_{\frac{C_0\sigma^2 m}{\|\tilde{u}_{t-1}\|}}^{\|\tilde{u}_{t-1}\|} e^{\frac{-1}{2\sigma^2}(x_1+\zeta)^2}\, dx_1\\ &\leq 2\cdot(2\pi\sigma^2)^{-1/2}\mathbb{1}_\beta e^{\frac{\zeta^2}{2\sigma^2}}\int_{\frac{C_0\sigma^2 m}{\|\tilde{u}_{t-1}\|}+\zeta}^{\infty} e^{\frac{-x_1^2}{2\sigma^2}}\, dx_1. \qquad (27)\end{aligned}$$

We remark that $\zeta > 0$ if $-2\lambda w_t + \frac{c_{norm}}{2|1-2w_t|\sigma^2} > 0$ which holds since $\lambda < \frac{c_{norm}}{12\sigma^2} < \frac{c_{norm}}{4w_t|1-2w_t|\sigma^2}$. Therefore, the lower limit of integration is positive and we can use a Gaussian tail bound as in Lemma 7 to upper bound (27) by

$$\frac{\mathbb{1}_\beta 2\sigma e^{\frac{-1}{2\sigma^2}\left(\frac{C_0^2\sigma^4 m^2}{\|\tilde{u}_{t-1}\|^2}+2\frac{C_0\sigma^2 m\zeta}{\|\tilde{u}_{t-1}\|_2}\right)}}{\sqrt{2\pi}\left(\frac{C_0\sigma^2 m}{\|\tilde{u}_{t-1}\|}+\zeta\right)} \leq \frac{\mathbb{1}_\beta 2\sigma}{\sqrt{2\pi}\zeta} = \frac{\mathbb{1}_\beta 4\sigma}{\sqrt{2\pi}\|u_{t-1}\|_2\left(\frac{c_{norm}}{|1-2w_t|}-4\lambda\sigma^2 w_t\right)} \tag{28}$$

$$\leq \frac{\mathbb{1}_\beta 4\sigma}{\|u_{t-1}\|_2\left(\frac{c_{norm}}{3}-\frac{4C_\lambda}{C_{\sup}^2 m\log(N_0)}\right)}. \tag{29}$$

Putting it all together, and remembering to add back in the factor $e^{\lambda C_{\sup}^2\sigma^2 m\log(N_0)w_t^2} = e^{C_\lambda w_t^2} \leq e^{C_\lambda}$ we have previously ignored, we've bound $\mathbb{E}\left[e^{\lambda\Delta\|u_t\|_2^2}\mathbb{1}_\beta\mathbb{1}_{q_t=0}\mathbb{1}_U\mathbb{1}_{\|u_{t-1}\|_2\geq\beta}\middle|\mathcal{F}_{t-1}\right]$ from above with

$$\underbrace{\frac{\mathbb{1}_\beta e^{C_\lambda}C_{\sup}^2\sigma m\log(N_0)}{C_\lambda\varepsilon\|u_{t-1}\|_2}}_{(17)} + \underbrace{\frac{\mathbb{1}_\beta 3C_0 e^{\frac{6C_0C_\lambda}{C_{\sup}^2\log(N_0)}+C_\lambda}\sigma m}{\|u_{t-1}\|_2}}_{(22)} + \underbrace{\frac{\mathbb{1}_\beta e^{C_\lambda}\sigma}{\|u_{t-1}\|_2\left(\frac{1}{3}-\frac{2C_\lambda}{C_{\sup}^2 m\log(N_0)}\right)}}_{(25)}$$

$$+ \underbrace{\frac{\mathbb{1}_\beta 4\sigma e^{C_\lambda}}{\|u_{t-1}\|_2\left(\frac{c_{norm}}{3}-\frac{4C_\lambda}{C_{\sup}^2 m\log(N_0)}\right)}}_{(28)} \lesssim \frac{\mathbb{1}_\beta e^{C_\lambda}\sigma m\log(N_0)}{\|u_{t-1}\|_2\varepsilon}.$$

So, when $|w_t| < 1/2$ and $\|u_{t-1}\|_2 \geq \beta \gtrsim \frac{\sigma m\log(N_0)}{\rho\varepsilon}$ the claim follows.

Now, let's consider the case when $w_t \geq 1/2$. Then it must be, by Lemma 9, that $X_t \in B(\tilde{u}_{t-1}, \|\tilde{u}_{t-1}\|_2) \cap B(\hat{u}_{t-1}, \|\hat{u}_{t-1}\|_2)^C$. By non-negativity of the exponential function, we can always upper-bound the moment generating function by instead integrating over $X_t \in B(\hat{u}_{t-1}, \|\hat{u}_{t-1}\|_2)^C \cap \{x_1 \leq 0\}$. Pictorially, one can see this by looking at the subfigure on the right in Figure 3. In this scenario, we're integrating over the region in green. The upper bound we're proposing is derived by ignoring the constraint from the blue region on the left half-space. Using this upper bound we can retrace through the steps we took to bound the integrals over R, S, and T with only minor modifications and obtain the desired result. By symmetry, an analogous approach will work for $w_t \leq -1/2$. ■

Lemma 18 *With the same hypotheses as Lemma 17,*

$$\mathbb{E}\left[e^{\lambda\Delta\|u_t\|_2^2}\mathbb{1}_\beta\mathbb{1}_{q_t=1}\mathbb{1}_U\mathbb{1}_{\|u_{t-1}\|_2\geq\beta}\middle|\mathcal{F}_{t-1}\right] \leq \rho.$$

Proof To begin, let's consider the case when $w_t < 1/2$. Recalling that $\tilde{u}_{t-1} = \frac{1}{1-2w_t}u_{t-1}$, and arguing as we did at the beginning of the proof of Lemma 17, Lemma 9 tells us

$$\mathbb{E}\left[e^{\lambda\Delta\|u_t\|_2^2}\mathbb{1}_\beta\mathbb{1}_{q_t=1}\mathbb{1}_U\mathbb{1}_{\|u_{t-1}\|_2\geq\beta}\middle|\mathcal{F}_{t-1}\right]$$
$$\leq (2\pi\sigma^2)^{-m/2}\mathbb{1}_\beta e^{\lambda C_{\sup}^2\sigma^2 m\log(N_0)(w_t-1)^2}\int_{B(\tilde{u}_{t-1},\|\tilde{u}_{t-1}\|_2)} e^{2\lambda(w_t-1)x^T u_{t-1}}e^{\frac{-1}{2\sigma^2}\|x\|_2^2}\,dx.$$

As before, we have denoted $\mathbb{1}_\beta := \mathbb{1}_{\|u_{t-1}\|_2 \geq \beta}$ for conciseness. Using rotational invariance, we may assume that $u_{t-1} = \|u_{t-1}\|_2 e_1$. Just as we did in Lemma 17, expressing this integral as nested iterated integrals gives us the probabilistic formulation

$$\frac{\mathbb{1}_\beta e^{\lambda C_{\sup}^2 \sigma^2 m \log(N_0)(w_t-1)^2}}{\sqrt{2\pi}\sigma} \int_0^{2\|\tilde{u}_{t-1}\|_2} e^{2\lambda(w_t-1)\|u_{t-1}\|x_1 - \frac{1}{2\sigma^2}x_1^2} \mathbb{P}\left(\sigma^2 \sum_{j=1}^{m-1} \gamma_j^2 \leq 2x_1\|\tilde{u}_{t-1}\|_2 - x_1^2\right) \, dx_1,$$

where, as before, the $\gamma_j \sim \mathcal{N}(0,1)$ are i.i.d. standard normal random variables. So, consider decomposing the above integral into the following two pieces. Set $R := \{x \in \mathbb{R}^m : 0 \leq x_1 \leq \frac{C_1\sigma^2 m}{\|\tilde{u}_{t-1}\|_2}\}$ and $S := \{x \in \mathbb{R}^m : \frac{C_1\sigma^2 m}{\|\tilde{u}_{t-1}\|_2} \leq x_1 \leq 2\|\tilde{u}_{t-1}\|_2\}$ where $C_1 \in (0,1)$ is a fixed constant. Then on R we have by Lemma 8

$$\begin{aligned}
&\mathbb{P}\left(\sum_{j=1}^{m-1} g_j^2 \leq (m-1)\left(\frac{1}{\sigma^2(m-1)}(2x_1\|\tilde{u}_{t-1}\|_2 - x_1^2)\right)\right) \\
&\leq \mathbb{P}\left(\sum_{j=1}^{m-1} g_j^2 \leq (m-1)\left(\frac{1}{\sigma^2(m-1)}(2x_1\|\tilde{u}_{t-1}\|_2)\right)\right) \leq 2e^{-c(1-C_1)^2(m-1)}.
\end{aligned}$$

Setting aside the factor $e^{\lambda C_{\sup}^2 \sigma^2 m \log(N_0)(w_t-1)^2}$ for the moment, we have that the integral over R is equal to

$$\begin{aligned}
&(2\pi\sigma^2)^{-1/2}\mathbb{1}_\beta \int_0^{\frac{C_1\sigma^2 m}{\|\tilde{u}_{t-1}\|_2}} e^{2\lambda(w_t-1)\|u_{t-1}\|x_1 - \frac{1}{2\sigma^2}x_1^2} \mathbb{P}\left(\sigma^2 \sum_{j=1}^{m-1} \gamma_j^2 \leq 2x_1\|\tilde{u}_{t-1}\|_2 - x_1^2\right) \, dx \\
&\leq (2\pi\sigma^2)^{-1/2}\mathbb{1}_\beta 2e^{-c(1-C_1)^2(m-1)} \int_0^{\frac{C_1\sigma^2 m}{\|\tilde{u}_{t-1}\|_2}} e^{2\lambda(w_t-1)\|u_{t-1}\|x_1 - \frac{1}{2\sigma^2}x_1^2} dx_1 \\
&= (2\pi\sigma^2)^{-1/2}\mathbb{1}_\beta 2e^{-c(1-C_1)^2(m-1)} e^{2\sigma^2\lambda^2(w_t-1)^2\|u_{t-1}\|_2^2} \int_{2\lambda\sigma^2(1-w_t)\|u_{t-1}\|_2}^{\frac{C_1\sigma^2 m}{\|\tilde{u}_{t-1}\|_2} + 2\lambda\sigma^2(1-w_t)\|u_{t-1}\|_2} e^{-\frac{1}{2\sigma^2}x_1^2} dx_1.
\end{aligned} \tag{30}$$

We remark that the lower limit of integration is strictly positive. Therefore, using a Riemann approximation to the integral and knowing that the maximum of the integral occurs at the lower limit of integration bounds (30) above by

$$\mathbb{1}_\beta 2e^{-c(1-C_1)^2(m-1)} \frac{C_1\sigma m}{\|\tilde{u}_{t-1}\|_2\sqrt{2\pi}}. \tag{31}$$

On S, we use the bound $\mathbb{P}\left(\sum_{j=1}^{m-1} g_j^2 \leq (m-1)\left(\frac{1}{\sigma^2(m-1)}(2x_1\|\tilde{u}_{t-1}\|_2 - x_1^2)\right)\right) \leq 1$ to get

$$(2\pi\sigma^2)^{-1/2}\mathbb{1}_\beta \times \tag{32}$$
$$\int_{\frac{C_1\sigma^2 m}{\|\tilde{u}_{t-1}\|_2}}^{2\|\tilde{u}_{t-1}\|_2} e^{2\lambda(w_t-1)\|u_{t-1}\|x_1 - \frac{1}{2\sigma^2}x_1^2} \times \mathbb{P}\left(\sum_{j=1}^{m-1} g_j^2 \leq (m-1)\left(\frac{1}{\sigma^2(m-1)}(2x_1\|\tilde{u}_{t-1}\|_2 - x_1^2)\right)\right) dx_1$$
$$\leq (2\pi\sigma^2)^{-1/2}\mathbb{1}_\beta \int_{\frac{C_1\sigma^2 m}{\|\tilde{u}_{t-1}\|_2}}^{2\|\tilde{u}_{t-1}\|_2} e^{2\lambda(w_t-1)\|u_{t-1}\|x_1 - \frac{1}{2\sigma^2}x_1^2} dx_1$$
$$= \mathbb{1}_\beta e^{2\sigma^2\lambda^2(w_t-1)^2\|u_{t-1}\|_2^2} \int_{\frac{C_1\sigma^2 m}{\|\tilde{u}_{t-1}\|_2}+2\lambda\sigma^2(1-w_t)\|u_{t-1}\|_2}^{2\|\tilde{u}_{t-1}\|_2+2\lambda\sigma^2(1-w_t)\|u_{t-1}\|_2} (2\pi\sigma^2)^{-1/2} e^{-\frac{1}{2\sigma^2}x_1^2} dx_1. \tag{33}$$

Since the lower limit of integration is strictly positive, we can use a Gaussian tail bound as in Lemma 7 to upper bound (32) by

$$\frac{\mathbb{1}_\beta\sigma e^{-\frac{1}{2\sigma^2}\left(\frac{C_1^2\sigma^4 m^2}{\|\tilde{u}_{t-1}\|_2^2}+4\lambda(1-w_t)C_1\sigma^4 m\frac{\|u_{t-1}\|_2}{\|\tilde{u}_{t-1}\|_2}\right)}}{\left(\frac{C_1\sigma^2 m}{\|\tilde{u}_{t-1}\|_2}+2\lambda\sigma^2(1-w_t)\|u_{t-1}\|_2\right)\sqrt{2\pi}} \leq \frac{\mathbb{1}_\beta\sigma}{\sqrt{2\pi}2\lambda\sigma^2(1-w_t)\|u_{t-1}\|}. \tag{34}$$

To summarize, we have shown, at least when $w_t < 1/2$, that

$$\mathbb{E}\left[e^{\lambda\Delta\|u_t\|_2^2}\mathbb{1}_\beta\mathbb{1}_{q_t=1}\mathbb{1}_U\middle|\mathcal{F}_{t-1}\right]$$
$$\leq \underbrace{\mathbb{1}_\beta e^{\lambda C_{\sup}^2\sigma^2 m\log(N_0)(w_t-1)^2} 2e^{-c(1-C_1)^2(m-1)}\frac{C_1\sigma m}{\|\tilde{u}_{t-1}\|_2\sqrt{2\pi}}}_{(31)} + \underbrace{\frac{\mathbb{1}_\beta e^{\lambda C_{\sup}^2\sigma^2 m\log(N_0)(w_t-1)^2}\sigma}{\sqrt{2\pi}2\lambda\sigma^2(1-w_t)\|u_{t-1}\|}}_{(34)}$$
$$\leq \frac{\mathbb{1}_\beta e^{\lambda C_{\sup}^2\sigma^2 m\log(N_0)(w_t-1)^2}}{\|u_{t-1}\|_2}\left(\frac{2\sigma m|1-2w_t|}{\sqrt{2\pi}}+\frac{\sigma}{\sqrt{2\pi}2\lambda\sigma^2(1-w_t)}\right)$$
$$\leq \frac{\mathbb{1}_\beta e^{\lambda C_{\sup}^2\sigma^2 m\log(N_0)(w_t-1)^2}}{\|u_{t-1}\|_2}\left(6\sigma m+\frac{\sigma}{\lambda\sigma^2\varepsilon}\right)$$
$$\leq \frac{\mathbb{1}_\beta e^{4C_\lambda}\sigma m\log(N_0)}{\|u_{t-1}\|_2}\left(\frac{6}{\log(N_0)}+\frac{1}{C_\lambda\varepsilon}\right)$$
$$\lesssim \frac{\mathbb{1}_\beta\sigma m\log(N_0)}{\|u_{t-1}\|_2\varepsilon}. \tag{35}$$

Therefore, when $\|u_{t-1}\|_2 \geq \beta \gtrsim \frac{Ce^{C_\lambda}\sigma m\log(N_0)}{\rho\varepsilon}$, (35) is bounded above by ρ as desired.

Now, let's consider the case when $w_t > 1/2$. In this scenario, we can express the expectation as

$$\mathbb{E}\left[e^{\lambda\Delta\|u_t\|_2^2}\mathbb{1}_\beta\mathbb{1}_{q_t=1}\mathbb{1}_U\middle|\mathcal{F}_{t-1}\right]$$
$$\leq e^{\lambda C_{\sup}^2\sigma^2 m\log(N_0)(w_t-1)^2}(2\pi\sigma^2)^{-m/2}\mathbb{1}_\beta \int_{B(\tilde{u}_{t-1},\|\tilde{u}_{t-1}\|_2)^C} e^{2\lambda(w_t-1)x^T u_{t-1}} e^{\frac{-1}{2\sigma^2}\|x\|_2^2}\, dx.$$

Using the exact same approach as in the proof of Lemma 17, we can partition the domain of integration into the following pieces:

$$\begin{aligned}\int_{B(\tilde{u}_{t-1},\|\tilde{u}_{t-1}\|_2)^C} e^{2\lambda(w_t-1)x^T u_{t-1}} e^{\frac{-1}{2\sigma^2}\|x\|_2^2}\, dx &= \int_{B(\tilde{u}_{t-1},\|\tilde{u}_{t-1}\|_2)^C\cap\{x_1\leq -\|\tilde{u}_{t-1}\|\}} e^{2\lambda(w_t-1)x^T u_{t-1}} e^{\frac{-1}{2\sigma^2}\|x\|_2^2}\, dx \\ &+ \int_{B(\tilde{u}_{t-1},\|\tilde{u}_{t-1}\|_2)^C\cap\{-\|\tilde{u}_{t-1}\|\leq x_1\leq \frac{-C\sigma^2 m}{\|\tilde{u}_{t-1}\|}\}} e^{2\lambda(w_t-1)x^T u_{t-1}} e^{\frac{-1}{2\sigma^2}\|x\|_2^2}\, dx \\ &+ \int_{B(\tilde{u}_{t-1},\|\tilde{u}_{t-1}\|_2)^C\cap\{-\frac{-C\sigma^2 m}{\|\tilde{u}_{t-1}\|}\leq x_1\leq 0\}} e^{2\lambda(w_t-1)x^T u_{t-1}} e^{\frac{-1}{2\sigma^2}\|x\|_2^2}\, dx \\ &+ \int_{B(\tilde{u}_{t-1},\|\tilde{u}_{t-1}\|_2)^C\cap\{0\leq x_1\}} e^{2\lambda(w_t-1)x^T u_{t-1}} e^{\frac{-1}{2\sigma^2}\|x\|_2^2}\, dx.\end{aligned}$$

The same arguments from the proof of Lemma 17 apply here with only minor modifications. Namely, an argument exactly like that given for (14) gives us

$$\int_{B(\tilde{u}_{t-1},\|\tilde{u}_{t-1}\|_2)^C\cap\{0\leq x_1\}} e^{2\lambda(w_t-1)x^T u_{t-1}} e^{\frac{-1}{2\sigma^2}\|x\|_2^2}\, dx \leq \frac{\sigma}{\lambda\sigma^2\varepsilon\|u_{t-1}\|}.$$

Similarly, the chain of logic used to derive (25) gives us

$$\int_{B(\tilde{u}_{t-1},\|\tilde{u}_{t-1}\|_2)^C\cap\{x_1\leq -\|\tilde{u}_{t-1}\|\}} e^{2\lambda(w_t-1)x^T u_{t-1}} e^{\frac{-1}{2\sigma^2}\|x\|_2^2}\, dx \leq \frac{\sigma}{\|u_{t-1}\|_2\left(\frac{1}{3}-\frac{2C_\lambda}{C_{\sup}^2 m\log(N_0)}\right)}.$$

Calculations for the derivation of (28) give us

$$\begin{aligned}\int_{B(\tilde{u}_{t-1},\|\tilde{u}_{t-1}\|_2)^C\cap\{-\|\tilde{u}_{t-1}\|\leq x_1\leq\frac{-C\sigma^2 m}{\|\tilde{u}_{t-1}\|}\}} & e^{2\lambda(w_t-1)x^T u_{t-1}} e^{\frac{-1}{2\sigma^2}\|x\|_2^2}\, dx \\ &\leq \frac{4\sigma}{\|u_{t-1}\|_2\left(\frac{c_{norm}}{3}-\frac{4C_\lambda}{C_{\sup}^2 m\log(N_0)}\right)}.\end{aligned}$$

Finally, the same reasoning that was used to derive (22) gives us

$$\int_{B(\tilde{u}_{t-1},\|\tilde{u}_{t-1}\|_2)^C\cap\{-\frac{-C\sigma^2 m}{\|\tilde{u}_{t-1}\|}\leq x_1\leq 0\}} e^{2\lambda(w_t-1)x^T u_{t-1}} e^{\frac{-1}{2\sigma^2}\|x\|_2^2}\, dx \leq \frac{3C_0 e^{\frac{6C_0C_\lambda}{C_{\sup}^2\log(N_0)}}\sigma m}{\|u_{t-1}\|_2}.$$

Following the remainder of the proof of Lemma 17 in this scenario gives us the result when $w_t > 1/2$. ■

Lemma 19 *With the same hypotheses as Lemma 17*

$$\mathbb{E}\left[e^{\lambda\Delta\|u_t\|_2^2}\mathbb{1}_{q_t=-1}\mathbb{1}_U\mathbb{1}_{\|u_{t-1}\|_2\geq\beta}\middle|\mathcal{F}_{t-1}\right]\leq\rho.$$

Proof The proof is effectively the same as that for Lemma 18. ■

Acknowledgments

This work was supported in part by National Science Foundation Grant DMS-2012546 and a UCSD senate research award.

References

Miklos Ajtai. The shortest vector problem in l2 is np-hard for randomized reductions. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 10–19, 1998.

Pierre Baldi and Roman Vershynin. The capacity of feedforward neural networks. *Neural networks*, 116:288–311, 2019.

Wojciech Banaszczyk. A beck—fiala-type theorem for euclidean norms. *European Journal of Combinatorics*, 11(6):497–500, 1990.

Wojciech Banaszczyk. Balancing vectors and gaussian measures of n-dimensional convex bodies. *Random Structures & Algorithms*, 12(4):351–360, 1998.

Nikhil Bansal. Constructive algorithms for discrepancy minimization. In *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, pages 3–10. IEEE, 2010.

Nikhil Bansal, Daniel Dadush, Shashwat Garg, and Shachar Lovett. The gram-schmidt walk: a cure for the banaszczyk blues. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, pages 587–597, 2018.

François Chollet et al. Keras. `https://keras.io`, 2015.

Matthieu Courbariaux, Yoshua Bengio, and Jean-Pierre David. Binaryconnect: Training deep neural networks with binary weights during propagations. In *Advances in neural information processing systems*, pages 3123–3131, 2015.

George Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems*, 2(4):303–314, 1989.

Daniel Dadush, Shashwat Garg, Shachar Lovett, and Aleksandar Nikolov. Towards a constructive version of banaszczyk's vector balancing theorem. *arXiv preprint arXiv:1612.04304*, 2016.

Ingrid Daubechies and Ron DeVore. Approximating a bandlimited function using very coarsely quantized data: A family of stable sigma-delta modulators of arbitrary order. *Annals of mathematics*, 158(2):679–710, 2003.

Ronen Eldan and Mohit Singh. Efficient algorithms for discrepancy minimization in convex sets. *arXiv preprint arXiv:1409.2913*, 2014.

Apostolos A Giannopoulos. On some vector balancing problems. *Studia Mathematica*, 122 (3):225–234, 1997.

Yunchao Gong, Liu Liu, Ming Yang, and Lubomir Bourdev. Compressing deep convolutional networks using vector quantization. *arXiv preprint arXiv:1412.6115*, 2014.

Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*, volume 1. MIT press Cambridge, 2016.

Yunhui Guo. A survey on methods and theories of quantized neural networks. *arXiv preprint arXiv:1808.04752*, 2018.

Suyog Gupta, Ankur Agrawal, Kailash Gopalakrishnan, and Pritish Narayanan. Deep learning with limited numerical precision. In *International Conference on Machine Learning*, pages 1737–1746, 2015.

Bruce Hajek. Hitting-time and occupation-time bounds implied by drift analysis with applications. *Advances in Applied probability*, pages 502–525, 1982.

Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding. *arXiv preprint arXiv:1510.00149, conference paper at ICLR*, 2016.

Nicholas JA Harvey, Roy Schwartz, and Mohit Singh. Discrepancy without partial colorings. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2014)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2014.

Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.

Itay Hubara, Matthieu Courbariaux, Daniel Soudry, Ran El-Yaniv, and Yoshua Bengio. Quantized neural networks: Training neural networks with low precision weights and activations. *The Journal of Machine Learning Research*, 18(1):6869–6898, 2017.

Hi Inose, Y Yasuda, and Jun Murakami. A telemetering system by code modulation-δ-σmodulation. *IRE Transactions on Space Electronics and Telemetry*, (3):204–209, 1962.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

Yong-Deok Kim, Eunhyeok Park, Sungjoo Yoo, Taelim Choi, Lu Yang, and Dongjun Shin. Compression of deep convolutional neural networks for fast and low power mobile applications. *arXiv preprint arXiv:1511.06530, conference paper at ICLR*, 2016.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Tamara G Kolda and Dianne P O'leary. A semidiscrete matrix decomposition for latent semantic indexing information retrieval. *ACM Transactions on Information Systems (TOIS)*, 16(4):322–346, 1998.

Richard Kueng and Joel A Tropp. Binary component decomposition part ii: The asymmetric case. *arXiv preprint arXiv:1907.13602*, 2019.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436, 2015.

Laszlo Lovasz, Joel Spencer, and Katalin Vesztergombi. Discrepancy of set-systems and matrices. *European Journal of Combinatorics*, 7(2):151–160, 1986.

Shachar Lovett and Raghu Meka. Constructive discrepancy minimization by walking on the edges. *SIAM Journal on Computing*, 44(5):1573–1582, 2015.

Mikhail Menshikov, Serguei Popov, and Andrew Wade. *Non-homogeneous random walks: Lyapunov function methods for near-critical stochastic systems*, volume 209. Cambridge University Press, 2016.

Sean P Meyn and Richard L Tweedie. *Markov chains and stochastic stability*. Springer Science & Business Media, 2012.

Robin Pemantle and Jeffrey S Rosenthal. Moment conditions for a sequence with negative drift to be uniformly bounded in lr. *Stochastic Processes and their Applications*, 82(1): 143–155, 1999.

Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. Xnor-net: Imagenet classification using binary convolutional neural networks. In *European conference on computer vision*, pages 525–542. Springer, 2016.

Thomas Rothvoss. Constructive discrepancy minimization for convex sets. *SIAM Journal on Computing*, 46(1):224–234, 2017.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. doi: 10.1007/s11263-015-0816-y.

Jurgen Schmidhuber. Deep learning in neural networks: An overview. *Neural networks*, 61: 85–117, 2015.

David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484, 2016.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Joel Spencer. Six standard deviations suffice. *Transactions of the American mathematical society*, 289(2):679–706, 1985.

Roman Vershynin. *High-dimensional probability: An introduction with applications in data science*, volume 47. Cambridge university press, 2018.

Peisong Wang and Jian Cheng. Fixed-point factorized networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4012–4020, 2017.

Juyang Weng, Narendra Ahuja, and Thomas S Huang. Cresceptron: a self-organizing neural network which grows adaptively. In *[Proceedings 1992] IJCNN International Joint Conference on Neural Networks*, volume 1, pages 576–581. IEEE, 1992.