

## ABSTRACT

Title of Dissertation: **FAST PREDICTION OF FULL QUANTUM DYNAMICS  
WITH DEEP RECURRENT NEURAL NETWORKS**

**Yash Anand, Mark Gouzoulis, Linfred Kingston,  
Jander Kugelman, Jonathan Li, Dustin Pham,  
Eashan Siddalingaiah, William Taylor, George Witt**

Dissertation Directed by: **Dr. Avik Dutt  
Dr. Mohammad Hafezi  
Gemstone Honors Program**

Numerical simulations of interacting quantum systems are computationally very intensive, typically requiring resources that scale exponentially in the number of particles. A plausible approach to overcoming this unfavorable simulation time is to train deep neural networks over short timescales and use them to infer dynamics over much longer timescales. We demonstrate that such a speedup is possible using deep recurrent neural networks, including LSTM and Transformer-based networks, by predicting the quantum dynamics of multiple 'classic' systems - the Ising, Heisenberg, and Hubbard models, with up to 9 spins, at most. We observe up to 3 orders of magnitude of data generation speedup for systems that can still be simulated with full system evolution. We observe this same performance -  $\mathcal{O}(0.1)$  seconds - to generate data samples that cannot be generated with the resources we have available to use. Unique to our work, we predict the full wavefunction dynamics of the systems, which can then be used to calculate the evolution of measurable and theoretical observables over time. We present sample predictions

for our models and compare the efficacy of the different approaches with varying context-length for prediction, spin count, and Hamiltonian parameters (mixing, interaction strength, etc.), at best accurately predicting ( $< 10^{-6}$  mean square error - MSE) up to 90% of a single period with 10% of a period for context. We probe a number of frustrations, including square and triangular interaction lattices, more complex next-nearest neighbor interactions, etc. to understand what currently limits strong machine learning - ML - results in this space. Our results indicate that the primary inhibitor to fast prediction at scale is the system scale, not the complexity of the dynamics. We anticipate that our work will provide insights towards extending the coherence time of quantum systems such as qubits and spins by determining the issues that stand in the way of network training and prediction on realistic Hamiltonians. We further believe that this work has immediate application in the simulation of large-scale neutral atom arrays, like Yt-171, under so-called Lieb-Robinson bounds.

# FAST PREDICTION OF FULL QUANTUM DYNAMICS WITH DEEP RECURRENT NEURAL NETWORKS

by

Team ASQ

Dissertation submitted to the Gemstone Honors Program of the  
University of Maryland, College Park in partial fulfillment  
of the requirements for Gemstone Honors Certification  
2025

## Advisory Committee:

Dr. Avik Dutt, Mentor  
Dr. Maksym Morawski  
Dr. Fardina Alam  
Mr. Shams Mehdi  
Ms. Greeshma Oruganti

## Acknowledgments

We thank our mentors, Dr. Avik Dutt and Dr. Mohammed Hafezi, for their expert guidance and our librarian, Jordan Sly. Additionally, we thank Dr. En-Jui Kuo for his guidance and early notes on our project. Thank you to our discussants, Dr. Maksym Morawski, Dr. Fardina Alam, Greeshma Oruganti, and Shams Mehdi. We also thank the UMD Zaratán HPC systems staff for their support with data management and all of our LaunchUMD Donors. Finally, we thank Dr. David Lovell, Dr. Allison Lansverk, and everyone in the Gemstone program for their support.

## Table of Contents

|                                                   |     |
|---------------------------------------------------|-----|
| Acknowledgments                                   | ii  |
| Table of Contents                                 | iii |
| List of Abbreviations                             | v   |
| Chapter 1: Introduction                           | 1   |
| 1.1 Prior Work                                    | 1   |
| 1.1.1 Machine Learning in the Ising Model         | 2   |
| 1.1.2 Machine Learning in Open Systems            | 5   |
| 1.1.3 Transformer Applications in Quantum Systems | 6   |
| 1.1.4 Open Challenges and Gaps                    | 7   |
| 1.2 ML architectures                              | 8   |
| 1.2.1 LSTM architecture                           | 8   |
| 1.2.2 Transformer architecture                    | 11  |
| 1.3 Physical Background                           | 13  |
| 1.3.1 Quantum Mechanics                           | 14  |
| 1.3.2 Transverse Ising Model                      | 15  |
| 1.3.3 Ising Model with Next-Nearest Neighbors     | 17  |
| 1.3.4 Simulating Anisotropy                       | 17  |
| 1.3.5 Heisenberg Model                            | 18  |
| 1.3.6 Hubbard Model                               | 19  |
| Chapter 2: Methodology                            | 21  |
| 2.1 Overview                                      | 21  |
| 2.2 Data Generation                               | 22  |
| 2.2.1 State Preparation                           | 22  |
| 2.2.2 Additional Notes                            | 24  |
| 2.3 Ensuring Quality of Generated Data            | 24  |
| 2.3.1 Manual Verification of Simple Ising Model   | 25  |
| 2.3.2 Fourier Transform Analysis                  | 26  |
| 2.4 LSTMs                                         | 32  |
| 2.5 Transformers                                  | 33  |
| 2.6 Running Models                                | 34  |
| Chapter 3: Results                                | 36  |

|              |                                               |    |
|--------------|-----------------------------------------------|----|
| 3.1          | Overview                                      | 36 |
| 3.1.1        | Anisotropic Transverse Ising Model            | 37 |
| 3.1.2        | Anisotropic Ising Next-Nearest Neighbor Model | 43 |
| 3.1.3        | Heisenberg Square Lattice                     | 49 |
| 3.1.4        | Heisenberg Triangular Lattice                 | 54 |
| 3.1.5        | 1D Hubbard Model                              | 59 |
| Chapter 4:   | Conclusions                                   | 63 |
| 4.1          | Anisotropic Transverse Ising Model            | 63 |
| 4.2          | Anisotropic Ising Next-Nearest Neighbor Model | 65 |
| 4.3          | Heisenberg Models                             | 67 |
| 4.4          | 1D Hubbard Model                              | 68 |
| 4.5          | Summary                                       | 70 |
| 4.6          | Broader Societal Impacts                      | 73 |
| 4.7          | Future Steps                                  | 75 |
| Appendix A:  | Previous Experiments                          | 77 |
| A.1          | Overview                                      | 77 |
| A.2          | Echo State Network                            | 77 |
| Appendix B:  | QuTiP and Lindblad Master Equation            | 80 |
| Appendix C:  | Sample Visualizations                         | 82 |
| Bibliography |                                               | 84 |

## List of Abbreviations

|      |                                |
|------|--------------------------------|
| LSTM | Long Short-Term Memory         |
| ML   | Machine Learning (Model)       |
| RNN  | Recurrent Neural Network       |
| ESN  | Echo State Network             |
| MSE  | Mean Squared Error             |
| MAE  | Mean Averaged Error            |
| GRU  | Gated Recurrent Unit (Network) |
| RBM  | Restricted Boltzmann Machine   |
| LLM  | Large Language Model           |
| NNN  | Next-Nearest Neighbor          |

## Chapter 1: Introduction

### 1.1 Prior Work

Quantum systems have significant applications to modern engineering tasks like sensing and computing. The principle of superposition in quantum mechanics, which allows the existence of multiple states at once, is crucial to this. One such system, useful for computing, is the quantum bit (qubit) [1]. Unlike classical bits, which exist in a state of either 0 or 1 exclusively, qubits can exist in a superposition of both states simultaneously [1]. Qubits allow for unique quantum algorithms that solve classically challenging problems in tractable amounts of time [2]. Another key feature of quantum systems is entanglement, a phenomenon where qubits are strongly correlated so that measuring one can collapse the state of the other, regardless of distance [2]. This behavior enables secure quantum communication and quantum information processing [3]. Qubit systems cannot, however, be scaled easily due to experimental challenges and the need for correcting background noise and information loss to the environment – or decoherence [4,5]

This quantum decoherence, where environmental interactions rapidly degrade quantum states, imposes significant constraints on computation time and reliability, limiting practical applications [6,7]. ML techniques have been applied to enhance quantum error mitigation and correction [8,9]. By predicting quantum computation outputs before decoherence occurs, ML models could extend usable computation time and refine error mitigation strategies. Robust ML-

assisted approaches could be essential for improving the scalability and stability of quantum computing in real-world applications. We are interested in producing ML models to predict full-state dynamics for quantum systems. This work can be applied to reduce decoherence and applied to theoretical study with large scale transients (full state simulations).

One quantum system that is of great interest to us is the Ising model (Sec. 1.3.2). This system is used to study phase transitions in ferromagnetic systems. Despite its simplicity, it captures the essential features of critical phenomena. While it is fundamentally understood, simulation of large quantum systems is still challenging (see section 2 and appendix) and time-consuming. The constant-time capacity of ML for predictions makes it an extremely interesting tool to apply to the Ising and similar systems. We'll begin by understanding how ML is currently being applied to the Ising model.

### 1.1.1 Machine Learning in the Ising Model

ML models have been applied to the Ising model to study phase transitions in ferromagnetic systems [10]. These phase transitions occur when the temperature of the spin chain changes, leading to thermally disordered states with no magnetization or ordered states with aligned spins. This spontaneous switching is of great interest to study: it introduces a chaotic element to the system that could frustrate full state prediction. Various ML architectures have been successfully applied to predicting when phase transitions might occur [11, 12]. These works employed supervised, unsupervised, and generative models to the Ising model. They applied modern architectures like convolutional networks to detect phase transitions. Principle component analysis (PCA) techniques, and other related unsupervised techniques, were further able to highlight states

where these phase transitions occur. They also focused heavily on applying generative models to the system.

For example, generative models such as Restricted Boltzmann Machines (RBMs) and Variational Autoencoders (VAEs) were compared for their ability to learn probability distributions of spin configurations in a two-dimensional lattice with nearest-neighbor interactions [12]. RBMs were used to estimate the coupling strengths that govern phase transitions in both one-dimensional and two-dimensional systems [11]. They generate probability distributions over spin configurations without external magnetic field terms or requiring convolutional layers [11]. In contrast, VAEs benefited significantly from convolutional architectures to model local interaction (i.e., interactions between spins and their immediate neighbors). However, both models faced challenges in representing long-range correlations near phase transitions, a region where the critical point of the Ising model is present. The Ising model exhibits long-range spin correlations near the phase transitions that significantly influence the system's properties. Capturing these correlations requires the models to learn dependencies between the long-distance components of the system, which introduces more complexity. For RBMs, their structure and reliance on local updates during Gibbs sampling limit their ability to capture such long-range interactions effectively. VAEs, particularly those with convolutional layers, are better equipped to model local spatial features, but their ability to represent long-range dependencies diminishes as the size of the system increases.

To address these limitations, work in one of our advisor's group - Dr. Hafezi - introduced a hybrid method called Latent RNN, which combines RBMs with recurrent neural networks (RNNs). The latent RNN serves as a continuous function generator that ensures the RBM's parameters evolve smoothly with the underlying quantum dynamics. The continuous evolution

is achieved through inductive bias, a concept in machine learning that imposes constraints on the model to better match the nature of the problem being solved. By enforcing smooth changes in the RBM's parameters, the Latent RNN ensures that the system's dynamics are accurately captured, even when the Hamiltonian that governs the system is unknown. This is a significant advantage since traditional RBMs are typically static models. They are not inherently equipped to handle chaotic time-dependent systems. The Latent RNN was tested on quantum systems with random initial states and low-dimensional subspaces, where it demonstrated improved fidelity compared to static RBMs. For instance, the method was able to predict short-term quantum dynamics more accurately. This is valuable in situations where the system exhibits complex, non-local interactions, such as near critical points. However, they did not consider how the problem dynamics affect model performance. E.g. while an inductive bias of smooth evolution does well for systems that interact to a certain strength, it might perform very differently for more jagged quantum systems under strong interactions.

These works, and other similar works, emphasize either the interactions in the model or the evolution of statistical distributions in the full state over time, or they employ techniques that do not focus on using RNNs alone (not RNNs on the latent probability distribution spaces). Thus, they stand apart from our work, which emphasizes a study of ML model prediction of the full time dynamics of the system. This leaves our study room to shine in the Ising model. Furthermore, we can extend our initial studies of the Ising model to closely related models, setting our own work even farther apart from the current standard.

### 1.1.2 Machine Learning in Open Systems

Some work similar to ours has been done on predicting quantum system dynamics for *open* quantum systems. Open quantum systems are those that are driven or affected by an external environment, or 'bath' [13]. Long short-term memory recurrent neural networks (LSTM-RNNs) were employed to simulate long-term quantum dynamics based on short-term historical data [13]. This approach utilized hyperparameter optimization techniques, such as simulated annealing (SA), Bayesian optimization with a tree-structured Parzen estimator (BO-TPE), and random search (RS), to construct optimal neural network architectures [13]. Among these techniques, SA performs the best, increasing the LSTM model's accuracy and computational efficiency. To improve the inherent uncertainties in quantum simulations, two techniques were implemented: bootstrap resampling and Monte Carlo dropout methods [13]. Bootstrap resampling focused on uncertainties related to the variability in training data distributions. This is done by generating multiple models trained on resampled datasets to assess prediction variability. In contrast, the Monte Carlo dropout method captured model uncertainties by randomly switching off neurons during prediction. Combined, the techniques quantified model uncertainties and provided confidence intervals for predictions.

Yet while their work is similar to ours, it did not focus greatly on varying properties of the problem, like mixing. They instead focused on properties of the learning agents. Furthermore, [13] worked on open quantum systems, which differ fundamentally from our closed quantum systems.

This work does involve predicting long-term quantum dynamics with an LSTM, similar to ours. However, we believe our work stands apart by focusing on the following: (a) potential

speedup applications for modern theory work, (b) elements of the physical problem itself - not the ML models - that complicate learning, and (c) closed quantum systems that are driven by their own self-interactions.

### 1.1.3 Transformer Applications in Quantum Systems

LSTMs are not the only recurrent neural network of interest to us. While they have been applied in related work, they are not the most modern and well-known architecture that has been developed in recent years.

The transformer model has opened up doors for development in many sectors, such as natural language processing, computer vision, and scientific computing [14–16]. Specifically for quantum applications, there have been several attempts to use transformers to solve different physical problems. One such problem is the Many-Body problem, where researchers aim to understand the behavior of multiple interacting particles in a quantum system. To address this, a Transformer Quantum State (TQS) was developed [17]. TQS was able to generalize across different physical parameters, predict field strengths from experimental measurements, and transfer learned knowledge to new quantum systems. It acts like an ‘LLM’ for quantum systems, with predictions that can generalize to as-yet-unseen quantum systems. The model was fine-tuned to accurately determine the ground state of the Transverse Field Ising (TFI) model, demonstrating its efficacy in predicting phase transitions and field strengths with minimal computational cost. Yet, once again, there was no emphasis on studying physical model frustration on ML learning. We hope to stand apart by employing similar transformer-encoder (only) models, yet with the goal of understanding the physical frustrations that stand in the way of accurate long-term

predictions.

Stepping away from our proposed line of study, transformers have also been explored for applications beyond traditional computational physics problems. Researchers have improved classical transformer-based neural network architectures for image analysis by incorporating quantum mechanics. This advancement led to the development of Quantum Vision Transformers [18]. The study proposed three quantum transformer architectures: the Orthogonal Patch-wise Transformer, Quantum Orthogonal Transformer, and Direct Quantum Attention. Each of these transformers leverages quantum circuits to enhance computational efficiency. These architectures use trainable quantum orthogonal layers to process visual data. They have shown competitive performance and, in some cases, outperformed classical vision transformers on standard medical image datasets. While studying quantum systems can improve ML tasks - a potential application of our work - we will not discuss how our current work can be applied to improving quantum systems.

#### 1.1.4 Open Challenges and Gaps

Throughout the prior sections, we have emphasized the current gaps in the literature. First, most of the work reviewed focused on predicting statistical distributions or observables rather than full wavefunctions. Second, while deep learning architectures like DNNs and generative models have been used, recurrent structures such as LSTMs and transformers have seen limited application in closed systems. Many metrics could be compared to derive what works best for the application of predicting quantum systems: accuracy, computation time, hardware resource requirements, context provided to the model, etc. Further, different environmental settings could

be compared to determine the fundamental physical effects that make a system more challenging to predict accurately. Certain model settings, like the amount of context provided to the model, have not been fully explored. In this work, we fill this gap by evaluating recurrent neural networks such as LSTMs and Encoder Transformers, similar to some of the sources reviewed above.

## 1.2 ML architectures

### 1.2.1 LSTM architecture

Recurrent neural networks (RNNs) are a category of ML model designed for processing sequential data [19]. LSTMs are a type of RNN designed to address the vanishing gradient problem, [20] arising when gradients in an RNN shrink exponentially over long sequences. As a result, RNNs that fail to address this issue struggle to model complex data with long-term dependencies [20]. LSTMs are thus a clear candidate for predicting the time evolution of full wavefunctions, as they capture long-term temporal dependencies present in the dynamics of quantum spin systems.

The LSTM consists of memory cells and gates designed to mitigate the vanishing gradient problem during backpropagation [20]. Memory cells can maintain information for longer periods, allowing the LSTM to retain information across large time gaps [20]. These are called cell states and are the main novel approach of the LSTM model. The LSTM architecture utilizes multiple gates that control what information is stored in the cell states. The gates and states can be seen in figure 1.1.

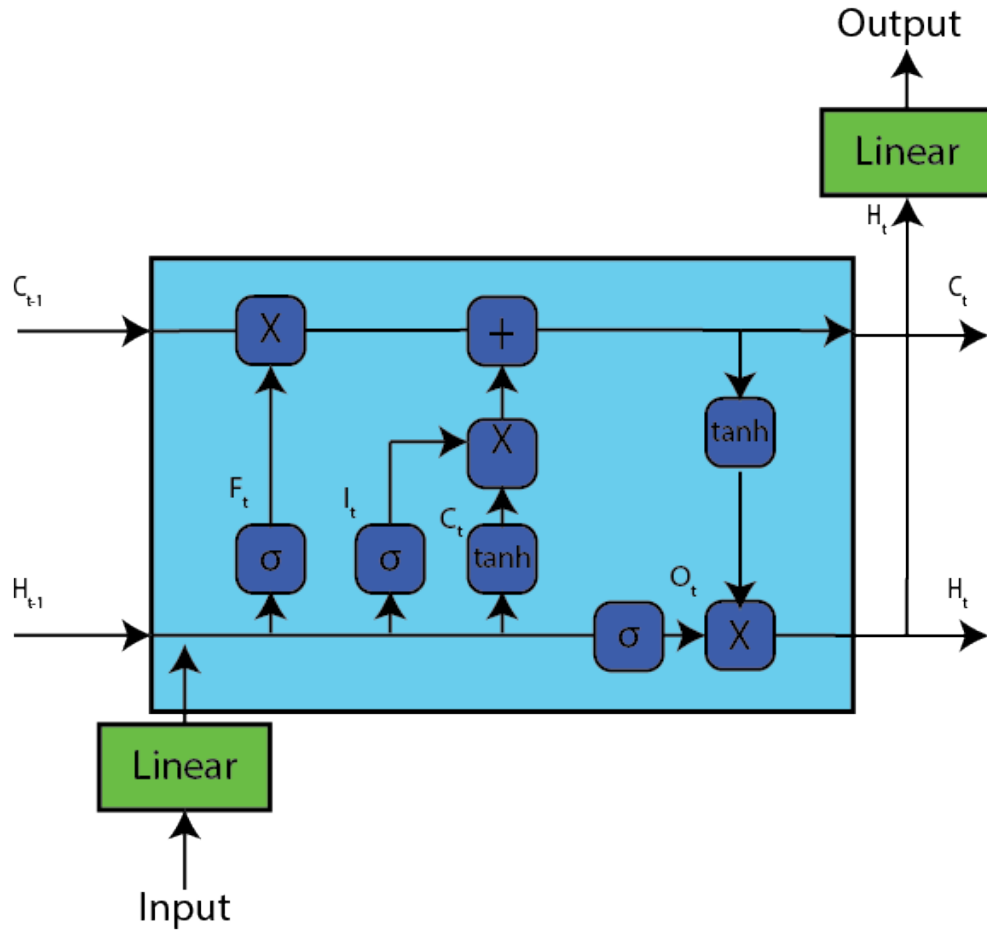


Figure 1.1: This is similar to a base LSTM architecture but with a linear layer for the input and output to encode and decode information. In the base model, the cell states and hidden states pass through multiple activation functions to eventually make an output and return the updated cell states and hidden states. The hidden state, once iterations through time have finished, can be used as the output.

The cell states in figure 1.1 take input from the previous cell state and then undergo a series of modifications depending on the new information and the learning algorithm. The cell states act as the long-term memory, but the information given to them is controlled by the input, forget, and output gates. The input gate will take any new pertinent information and add it to the cell state. Then, the forget gate will "forget" information from the previous cell state that is no longer important. Next, the output gate will take information from the cell state to pass to the hidden state which controls the visible memory for the model. The equations below provide more details

on the computations behind the gates and cells.

$$i_t = \sigma(W_{ii}x_t + b_{ii} + W_{hi}h_{t-1} + b_{hi})$$

$$f_t = \sigma(W_{if}x_t + b_{if} + W_{hf}h_{t-1} + b_{hf})$$

$$g_t = \tanh(W_{ig}x_t + b_{ig} + W_{hg}h_{t-1} + b_{hg})$$

$$o_t = \sigma(W_{io}x_t + b_{io} + W_{ho}h_{t-1} + b_{ho})$$

$$c_t = f_t \odot c_{t-1} + i_t \odot g_t$$

$$h_t = o_t \odot \tanh(c_t)$$

where  $i, f, g$ , and  $o$  are the input, forget, cell, and output gates at time  $t$ . Then  $c$  and  $h$  represent the cell and the hidden state. The weight for each gate and state is represented by  $W$  and the bias by  $b$ .

With this, the LSTM can capture longer term time information than original RNN approaches. Model hyperparameters, such as hidden state size, encoding layers, etc. are discussed later. Note that they will remain fixed in the thesis to ensure our comparison sticks to elements of the physical system.

An LSTM uses multiple hyperparameters to define its size and training. These hyperparameters include the number of layers, the number of hidden units, the dropout, the number of epochs, bias, learning rate, etc. Hyperparameters can greatly influence model performance and time required to reach an optimal solution. For example, increasing the number of layers may increase the complexity of patterns that the LSTM can learn but will also likely increase com-

putation time as more cells will need to be updated. The number of hidden units also increases the density of the model as they represent the feature space of the model. So, as the number of hidden units increases, the model can represent increasingly complicated patterns. However, it is important not to increase its density so much that the model overfits the training data - i.e. fails to generalize well or 'memorizing' training data. The chosen hyperparameters we use are listed in section 2.

### 1.2.2 Transformer architecture

Transformers are a relatively modern architecture that skips over the need for a hidden state entirely. The transformer architecture focuses on encoding and decoding a positionally-embedded input vector in one shot [14]. The transformer architecture does this through the 'self-attention' sub-architecture [14]. By embedding the temporal position of each element in the input vector, as well as 'paying attention' to all elements preferentially through training, the transformer is able to extract patterns from the entire sequence in one go. The architecture of the transformer encoder block is presented in Figure 1.2. While the original work [14] used both encoder and decoder blocks in the transformer, we stick to encoder-only architectures to relate more to some of our reviewed works like [17].

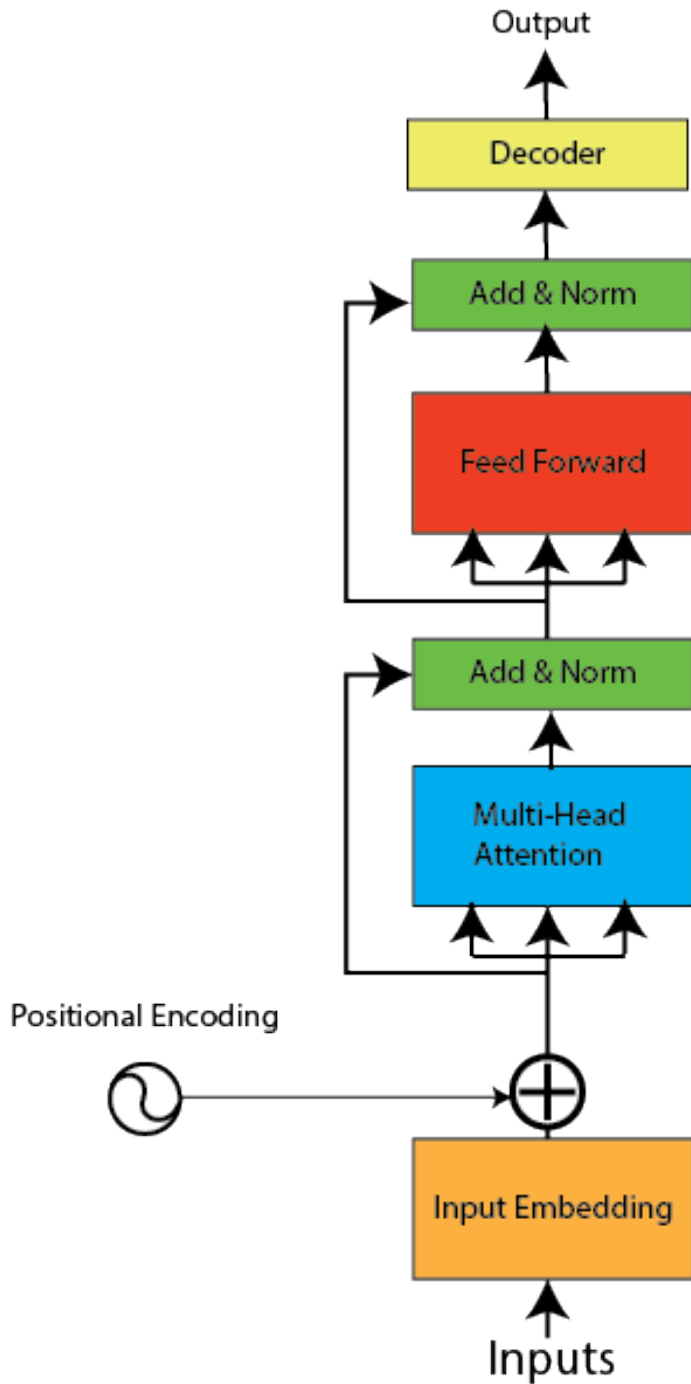


Figure 1.2: The transformer architecture consists of an encoder with a linear layer acting as the output layer and self-attention blocks mixed in. Positional encodings and self-attention are kept exactly as used in [14].

Within the encoder block, so-called 'multi-head' attention is employed. Multi-head attention simply concatenates the result from single-head attention, which is represented in equation

1.1 using an input  $X$  (already embedded).

$$A = \text{softmax} \left( \frac{(XW_Q)(XW_K)^T}{\sqrt{d_k}} \right) (XW_V) \quad (1.1)$$

where  $W_i$  represents the  $i$ th input individual weight matrix, query, key, or value, respectively, with an embedding size of  $d_k$ , resulting in a self-attention output  $A$ . The decoder head in figure 1.2 is not included in our examination, as it is primarily employed for generative tasks which we consider beyond the scope of this investigation.

The positional encoding is also shown in equation (1.2), though this is unmodified from the original in [14].

$$f(x) = \exp \left( \text{range}(d_k) * \left( -\frac{\log(P)}{d_k} \right) \right) \quad (1.2)$$

Eq. (1.2) is the positional encoding equation, where all the even elements of the input  $X$ ,  $x_i$ , are transformed as  $\sin(if(x_i))$ , and the odd elements  $x_j$  as  $\cos(jf(x_j))$ , with  $d_k$  being the set encoding dimension,  $P$  a fixed large constant, and  $\text{range}$  being the function that returns a list of numbers from 0 to the specified parameter.

### 1.3 Physical Background

In the following sections, we will explain the physical background for our system. This includes both the absolute fundamentals of quantum mechanics as well as the Hamiltonians we use in our project in roughly increasing order of complexity. The more complex Hamiltonians, such as the anisotropic generalized Ising model (Sec. 1.3.4), also typically involve a larger number of

parameters, such as coupling strengths, to be specified.

### 1.3.1 Quantum Mechanics

In quantum mechanics, a state, written as  $|\psi(t)\rangle$ , can be expressed as a vector of values expressing a linear combination of basis functions. This is in contrast to standard linear algebra, where vectors are linear combinations of basis vectors. This space of basic ‘functions’ is referred to as the *Hilbert space* of the system. A simple example of a Hilbert space is a Fourier Series, which is a linear combination (Fourier coefficients) of basis functions - sines and cosines of different frequencies.

The Hamiltonian  $H$  is an operator that represents the total energy of the system. An operator is a matrix (in the Hilbert Space) that can act on a quantum state. Since  $H$  corresponds to a physical observable, energy, then taking the *expectation value* of the operator gives a measurement of the observable itself. In this case it is energy. An expectation value can be simply written here as  $\langle\psi| H |\psi\rangle$ , where  $|\psi\rangle$  is a column vector and  $\langle\psi|$  is a row vector. Following ideas from classical mechanics, the total energy of a system can be used to determine its time evolution. In quantum mechanics, the Hamiltonian evolves the total state  $|\psi\rangle$  under the Schrödinger equation.

$$i\hbar \frac{\partial}{\partial t} |\psi(t)\rangle = H |\psi(t)\rangle \quad (1.3)$$

As is clear in Eq. (1.3),  $H$  determines the entire character of the time evolution of  $|\psi\rangle$ .

So, we focus our study on the time-dependent solution  $|\psi(t)\rangle$  to this equation under various Hamiltonians ranging from different versions of the Ising model to the Hubbard model. We also restrict ourselves to situations where the *total* energy is constant, meaning that the entirety of the

dynamics we wish to predict are purely internal.

We must also discuss the idea of ‘spin’. Spin can be thought of (for particles with mass) as ‘intrinsic angular momentum’. It is a property of the particle itself that determines what spin it has. For the rest of this writing, we will use the word ‘spin’ to refer to the particles themselves. Note that this is because the only physical quality of the object we are most concerned with is the spin.

### 1.3.2 Transverse Ising Model

First is the standard Ising model. The Ising model consists of a one-dimensional chain of spins interacting with their nearest neighbors in the presence of an external magnetic field. The Hamiltonian is as follows:

$$H = -J \sum_{\langle i,j \rangle} \sigma_i^z \sigma_j^z - h \sum_i \sigma_i^x. \quad (1.4)$$

where  $\sigma^z$  represents the  $z$  axis Pauli matrix,  $\sigma^x$  the  $x$  axis Pauli matrix, and the index  $i, j$  represents which spin site it acts on. So, for  $N$  spins in a row, the total energy is the energy of the field interacting with each spin in the chain plus the energy of the interactions of each spin with its neighbors. The Pauli matrices are expressed below

$$\sigma^z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \quad (1.5)$$

$$\sigma^y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix} \quad (1.6)$$

$$\sigma^x = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \quad (1.7)$$

where all matrices are expressed in the two-state Hilbert space for a *single* spin (up, down). Of course, in a larger system, like with  $N$  spins, the overall Hilbert space is formed by taking the Kronecker product between the Hilbert spaces of each spin. So, say we have  $N$  spins, each with a Hilbert space of size 2, then  $\sigma_j^z$  has the dimensionality of  $2^N \times 2^N$ , and the chunk corresponding to the  $j$ th spin is  $\sigma^z$  while the rest of the matrix is an identity matrix.

$J$  is a coupling constant, determining the strength of interaction between neighboring spins. The constant  $h$  is the strength of the magnetic field. At  $h = 0$ , the system exhibits ferromagnetic order. While at large  $h$ , the spins align along the transverse field, leading to a paramagnetic phase. A quantum critical point exists at  $h = J$ , where the system undergoes a phase transition.  $\sigma_i^z$  and  $\sigma_i^x$  are spin operators acting on the  $i$ th particle in the  $z$  and  $x$  directions, respectively. The sum over  $\langle i, j \rangle$  indicates that in each term of the sum,  $i$  and  $j$  are nearest neighbors. This model is pivotal for studying quantum phase transitions, and easy to simulate as a starting point. It admits exact solutions in 1D, and for the  $h = 0$  case, Onsager also provided exact 2D solutions.

### 1.3.3 Ising Model with Next-Nearest Neighbors

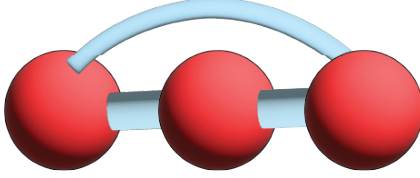


Figure 1.3: This image of a 1-dimensional next-nearest neighbor interaction represents how particles in this system interact and couple with each other. This lattice represents our anisotropic next-nearest neighbor (NNN) model. The simple Ising model is a 1D chain with nearest neighbor connections only. Hubbard is also modeled as a 1D chain.

An extension of the aforementioned Ising model with nearest-neighbor couplings consists of the addition of interactions with each spin's second-nearest neighbor. The Hamiltonian is given as follows:

$$H = -J_1 \sum_{\langle i,j \rangle} \sigma_i^z \sigma_j^z - J_2 \sum_{\langle\langle i,k \rangle\rangle} \sigma_i^z \sigma_k^z - h \sum_i \sigma_i^x. \quad (1.8)$$

$J_1$  determines the coupling strength of neighboring spins, whereas  $J_2$  determines the strength of next-nearest neighbor interactions. Other values and operators are the same as in the previous model. The corresponding sums are over pairs of nearest and next-nearest spins, respectively.

### 1.3.4 Simulating Anisotropy

To add complexity to our Ising Model simulations, we add terms in the Hamiltonian for  $\sigma_x \sigma_x$ ,  $\sigma_x \sigma_y$ , and all other permutations of  $\sigma_a \sigma_b$  with  $a, b \in \{x, y, z\}$ . We can represent this with a ‘mixing’ matrix

$$M = \begin{bmatrix} J_{xx} & J_{xy} & J_{xz} \\ J_{yx} & J_{yy} & J_{yz} \\ J_{zx} & J_{zy} & J_{zz} \end{bmatrix} \quad (1.9)$$

where each element of the matrix represents the relative strength of the interaction between those spin directions. This gives us complex interactions where interactions across  $x$  to  $x$  can be of different magnitudes than  $y$  interactions, etc. The method for generating such setups is discussed in the Methodology section.

### 1.3.5 Heisenberg Model

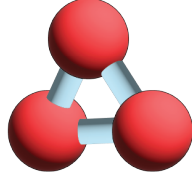


Figure 1.4: This image of a triangular lattice represents how particles in this system interact and couple with each other. As more particles enter this lattice the number of nearest neighbors for a particle can grow dramatically, which can drastically increase the complexity of their behavior. A particle can have at most 6 nearest neighbors.

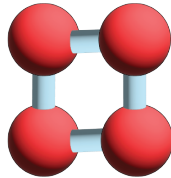


Figure 1.5: This image of a square lattice represents how particles in this system interact and couple with each other. With this lattice, a particle can have at most 4 nearest neighbors, making it more complex than the 1-dimensional lattice but less complex than the triangular lattice.

This model is very similar to the Ising model, just with lattice interactions instead of 1D interactions. The major distinction in our work is that the Heisenberg model is 2 dimensional

with isotropic interactions between the spins. The Heisenberg Hamiltonian is given as [21]:

$$H = -1/2 \sum_{\langle \vec{i}, \vec{j} \rangle} J_{xx} \sigma_i^x \sigma_j^x - 1/2 \sum_{\langle \vec{i}, \vec{j} \rangle} J_{yy} \sigma_i^y \sigma_j^y - 1/2 \sum_{\langle \vec{i}, \vec{j} \rangle} J_{zz} \sigma_i^z \sigma_j^z - h \sum_{\vec{i}} \sigma_i^x. \quad (1.10)$$

where the  $\vec{i}$  and  $\vec{j}$  represent the coordinates of the interacting spins in the lattice. Similar to the Ising model,  $J$  represents the interaction strength between two spins, and  $h$  represents an applied external field.

The 2D Heisenberg model is a great playground to study many quantum phenomena. Some phenomena of interest are the frustrations in magnetic systems, such as in triangular and Kagome lattice structures, and neutral atom arrays for quantum computing, using square lattices [22].

### 1.3.6 Hubbard Model

This model describes interacting electrons in a lattice. It captures the competition between electron mobility (hopping between sites) and on-site electron-electron repulsion. The Hamiltonian is as follows [23]:

$$H = -t \sum_{i,j,\sigma} c_{i\sigma}^\dagger c_{j\sigma} + U \sum_i n_{i\uparrow} n_{i\downarrow} \quad (1.11)$$

$t$  represents the hopping matrix element between neighboring sites. The summation over  $i$  and  $j$  is taken between nearest neighbors only.  $U$  is the Coulomb repulsion between two electrons (one spin-up and one spin-down) occupying the same site.  $c_{i\sigma}^\dagger$  creates an electron with spin  $\sigma$  at site  $i$ , whereas  $c_{j\sigma}$  removes an electron with spin  $\sigma$  at site  $j$ . These are the so-called creation and annihilation operators. If there is no electron at that site, applying the annihilation operator results in zero.  $n_{i\sigma} = c_{i\sigma}^\dagger c_{i\sigma}$  denotes the number operator at site  $i$ .

This model plays a crucial role in theories of quantum magnetism, materials science, unconventional superconductivity, and quantum many-body physics. As one related example of its utility in theoretical studies, [24] developed how the Hubbard model can be tied to Mott metal-insulator studies. It is of interest to us for its radically different creation/annihilation operator physical character and  $4^n$  scaling; we can investigate the effect of different physical mechanisms and input spaces by studying this well known model.

## Chapter 2: Methodology

### 2.1 Overview

To implement the aforementioned physical models, we have to walk through both our machine learning and data generation pipelines. The overall pipeline looks as follows

1. Prepare the Hamiltonian with a randomized mixing matrix.
2. Prepare the initial state into a random initial spin state (superposition of 'up', 'down' across all spins).
3. Simulate the evolution of the initial state over a fixed time using Python's QuTiP. QuTiP is an open-source package to simulate the dynamics of open quantum systems in Python.
4. Repeat this process a large number of times. We call the number of repetitions the number of *transients*.
5. Split the transients into some percentage for training and testing.
6. Using SLURM job arrays, spawn many training runs with Transformers or LSTMs on the training set using a variable amount of context (anywhere from 10% of the data to 90% of the timesteps), where the model predicts anything *not* used as context.

We employ a variety of RNNs in our simulated Quantum Dynamics systems. These RNNs include Long-Term Short-Term (LSTM) models, encoder-only Transformer models, and Echo State Networks (ESNs). However, our main body of work concerns only LSTMs and Transformer models, and discussion of the ESN network work is left to the appendix.

## 2.2 Data Generation

In this section, we discuss the parameters used for data generation alone. ML models are discussed in the following sections. Note that, following our anisotropy discussion in earlier sections, our model is ‘Ising-like’. However, we continue to refer to it as the Ising model simulation. Note we do use the phrase ‘Transverse’ model and ‘Ising’ model interchangeably, depending on what element of the system we wish to highlight. Fundamentally, we are pointing to the same Hamiltonian in both cases.

### 2.2.1 State Preparation

For  $N$  spins, and all the Hamiltonians, except the Hubbard model, we have  $2^N$  values to draw randomly, each with a complex amplitude. We draw these complex numbers through sampling phases for each complex number expressed with Euler’s formula. This sampling is uniform. The values are chosen such that the magnitude of  $\langle \psi | \psi \rangle = 1$ , as the squared state must represent a probability distribution. For the Hubbard model our state space has  $4^N$  values to draw randomly. We draw the initial state *randomly* for each transient.

We also prepare the Hamiltonian. We fix an interaction strength of  $J = 2$  J, and we set the field strength to 0 such that we only consider *cross* interaction in the system (for Ising models).

Heisenberg models are represented as written in the background sections. For the Ising model runs, we also generate a random coupling matrix  $M$  as described previously. This coupling matrix is randomly chosen from the standard Gaussian distribution, and then is scaled such that its Frobenius norm matches the norm set for this particular generation of  $M$ . This allows us to control the 'allowed' relative mixing by controlling the relative difference in scaling between the values in the matrix.

After solving for the quantum states at each time step, we store the states in a compressed numpy .npz file for later comparison against the predictions generated by each ML model. Our data is stored as a three-dimensional matrix where the first dimension is the number of runs, the second dimension is the number of measurements, and the last dimension holds the coefficient values for each state. Note that since each value in the matrix is imaginary, we expand this to hold 2 times the Hilbert space size as we fill out the matrix with both the real and imaginary values. These combined real and complex values put together, so-called  $C$  values, are complex numbers such that  $C \cdot C^*$  represents the probability our spins are in the corresponding state. This form of data representation is consistent with the State Representation technique. At our high level, we do not directly use density matrices. The details of the time-evolution itself are handled by QuTip 5.1.1. Each spin is described by coordinates  $\theta$  and  $\phi$  on the Bloch sphere, with each parameter generated using a random uniform distribution. We use the mesolve function from the QuTiP library to solve the Lindblad master equation to compute the time evolution of each initial state. Details of this quantum simulation are left to the appendix. Each transient is independent from every other transient. All our data is stored on the Zaratan computing cluster, in total consuming approximately 2.5 TB.

| Parameter                  | Value            |
|----------------------------|------------------|
| J Interaction Strength (J) | 2                |
| Heisenberg Model Mixing    | 0                |
| Hubbard Model Mixing       | 0                |
| Hubbard $t$ (J)            | 1                |
| Hubbard $U$ (J)            | 4                |
| Ising Spins                | $\{2, 4, 6, 8\}$ |
| Heisenberg Lattice X       | $\{2, 3\}$       |
| Heisenberg Lattice Y       | $\{2, 3\}$       |
| Hubbard Spins              | $\{2, 4\}$       |
| Transients                 | 20,000           |
| Timesteps                  | 200              |

Table 2.1: List of all parameters used by all Hamiltonians in generation. Note the Hubbard model is significantly limited in possible spins we can simulate as the system scales with  $4^N$ .

### 2.2.2 Additional Notes

The coupling matrix  $M$  is only generated for the Ising-like model runs. This is to investigate the effect of anisotropy and thereby complicate the simple Ising model. This is so that we can compare results more fairly between all runs, as the totally simplified Ising model is trivial to predict; small Ising systems can even be solved by hand. The data generation done here is entirely separate from fixing the amount of context to be used in training, so the explanation of our JOB ARRAY approach is left for later in the chapter.

## 2.3 Ensuring Quality of Generated Data

To confirm that our generation process matches expectations, we devised the following manual method to check our Ising model generations. This verifies our overall process as the only minor difference between the Ising generation and others is the Hamiltonian.

### 2.3.1 Manual Verification of Simple Ising Model

To verify our data generation, we solved the Lindblad Master Equation for a two-spin system by hand until the system of partial differential equations (PDEs). Then, we compared the solutions to the system of PDEs to our generated data. These spins only interacted in the  $x$  direction and had no environmental interactions. The simplified Hamiltonian used for the verification is given by:

$$H = \mathbb{I} + (\sigma_1 \otimes \sigma_1) = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix} \quad (2.1)$$

Using the Lindblad Master equation (see appendix B):

$$\frac{d\rho}{dt} = -\frac{i}{\hbar}[H, \rho] \quad (2.2)$$

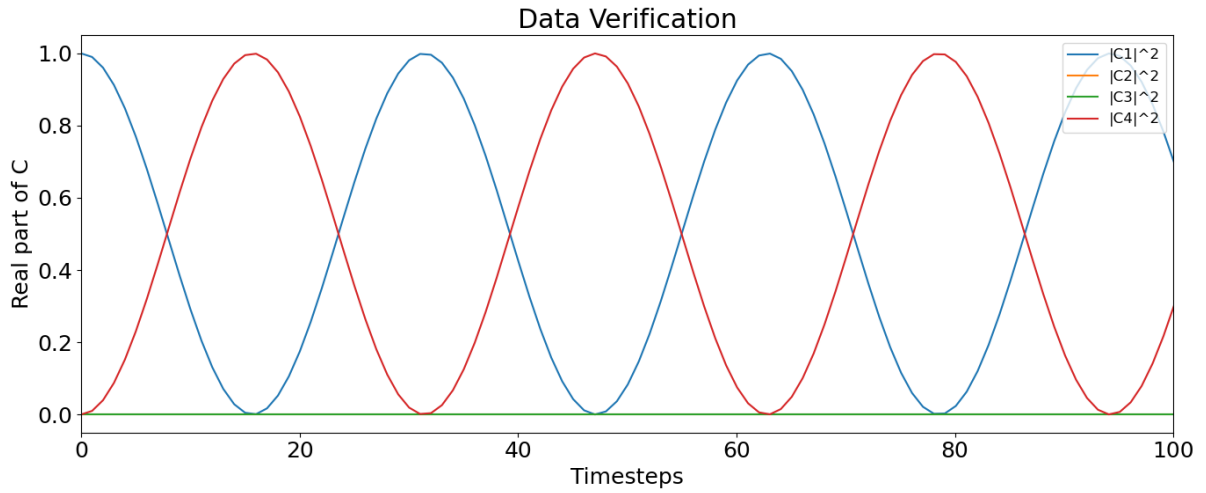
Where  $\rho$  is the density matrix with the elements given by  $\rho_{ij} = C_{ij}C_{ij}^*$ . Thus, the equation is simplified to a differential equation of matrices.

$$\frac{\partial}{\partial t} \begin{bmatrix} \rho_{00} & \rho_{01} & \rho_{02} & \rho_{03} \\ \rho_{10} & \rho_{11} & \rho_{12} & \rho_{13} \\ \rho_{20} & \rho_{21} & \rho_{22} & \rho_{23} \\ \rho_{30} & \rho_{31} & \rho_{32} & \rho_{33} \end{bmatrix} = -i \begin{bmatrix} \rho_{30} - \rho_{03} & \rho_{31} - \rho_{02} & \rho_{32} - \rho_{01} & \rho_{33} - \rho_{00} \\ \rho_{20} - \rho_{13} & \rho_{21} - \rho_{12} & \rho_{22} - \rho_{11} & \rho_{23} - \rho_{10} \\ \rho_{10} - \rho_{23} & \rho_{11} - \rho_{22} & \rho_{12} - \rho_{21} & \rho_{13} - \rho_{20} \\ \rho_{00} - \rho_{33} & \rho_{01} - \rho_{32} & \rho_{02} - \rho_{31} & \rho_{03} - \rho_{30} \end{bmatrix} \quad (2.3)$$

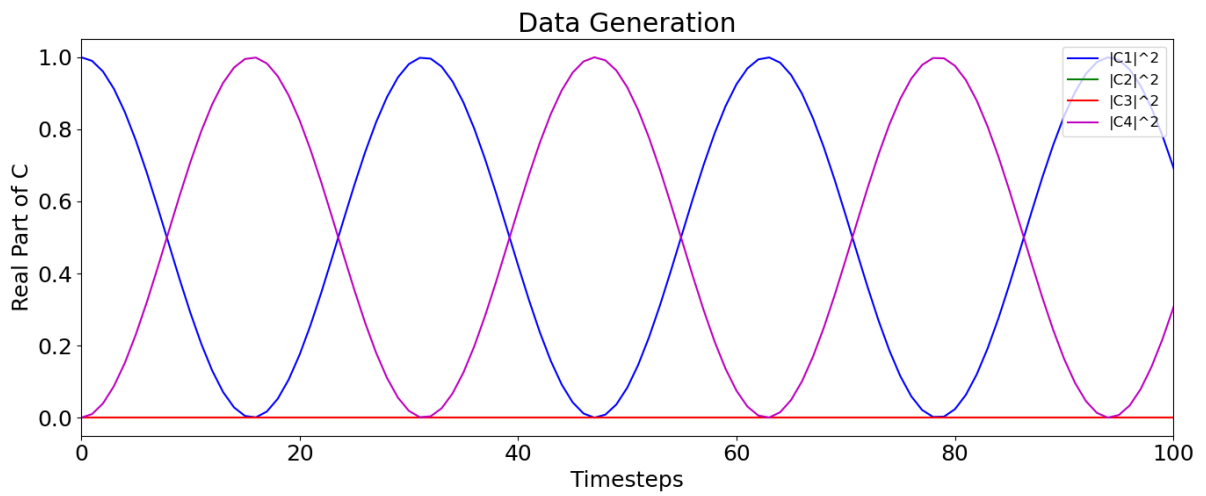
Using numerical methods to solve the equation, the solution to the system of PDEs agrees with the QuTiP solutions used for the data generation, as seen in Figure 2.1. For our data verification we used a fixed initial state of two spins given by  $\Psi = |\uparrow\uparrow\rangle$ . With this initial condition, we solved the above initial value problem of PDEs and compared it to the data generated by our code. Although this only directly verifies a simple Hamiltonian, our data generation code should hold for all other cases as only the Hamiltonian is changed and not the method to solve the time evolution.

### 2.3.2 Fourier Transform Analysis

For each of our generated datasets, we took the Fourier transform of the probability magnitude of the wavefunction,  $|\psi|^2$ , and counted the visible peaks. The probability magnitudes of the wavefunctions can be seen in Figure 2.2. Notice that in Figure 2.3, there are  $2^N$  frequency peaks present for the Ising and Heisenberg model datasets, where  $N$  is the number of spins. We expect to see  $2^N$  peaks as there are  $2^N$  different possible states if given  $N$  spins. Each of these possible state configurations correspond to a frequency and add constructively to the wavefunction, leading to peaks in the Fourier space. This can also be visualized in figures 2.2 as the wavefunction seems to be the addition of multiple different sinusoidal waves.

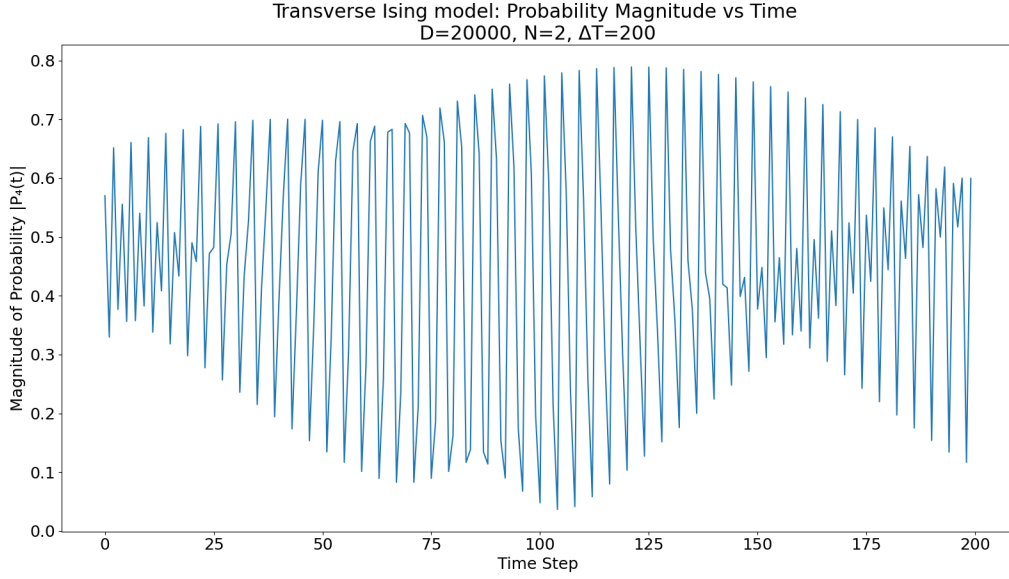


(a)



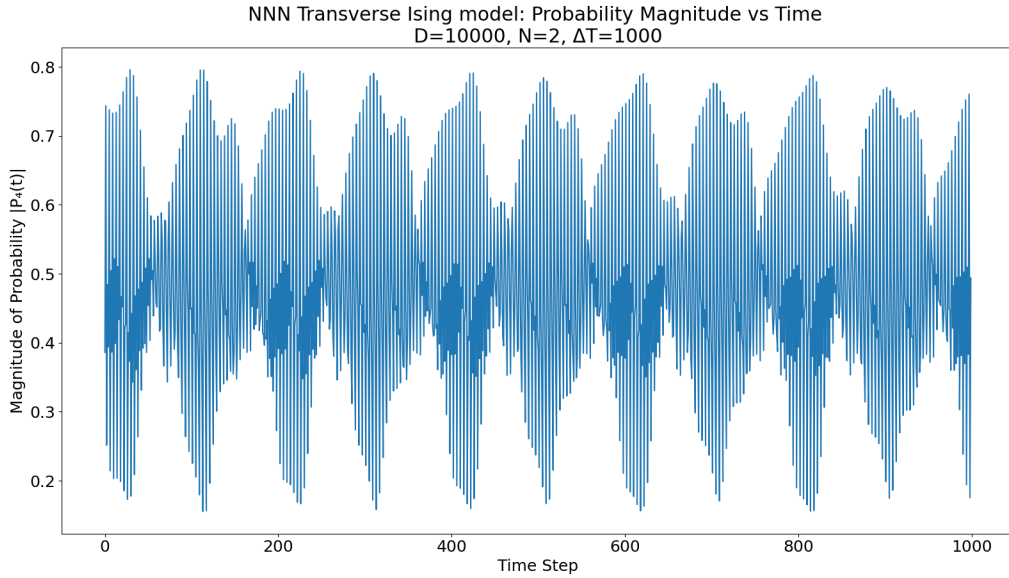
(b)

Figure 2.1: (a) Manual validation graph. (b) Data generation graph  
Notice that these graphs are identical, showing that our data generation is valid.



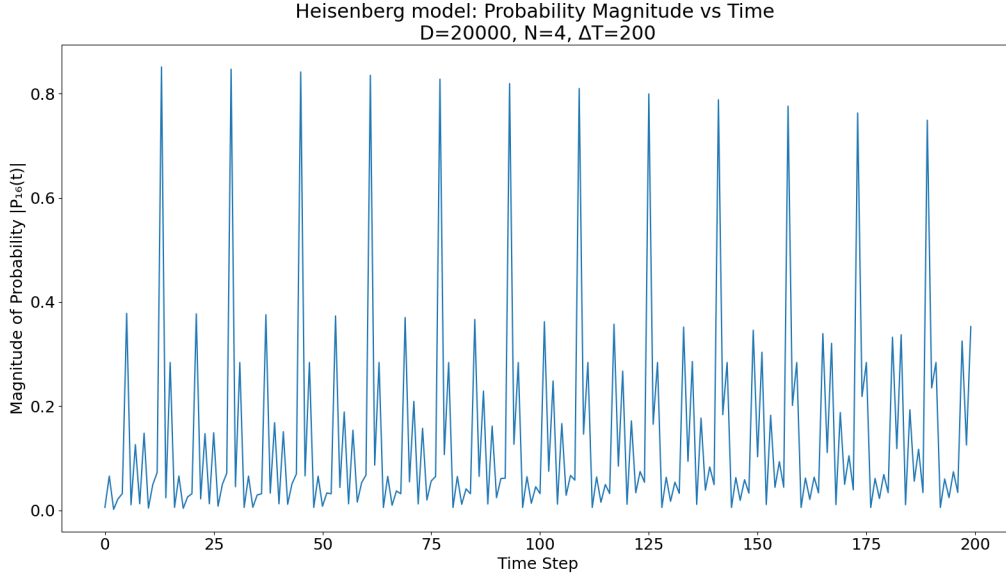
(a) Transverse Ising Model Probability Magnitude

Transverse Ising model exhibits rapid oscillations in  $|\Psi(t)|^2$  towards the early and later time steps. Note that the oscillations observed exhibit behavior of periodic dynamics that are expected of non-dissipative systems.



(b) NNN Transverse Ising Model Probability Magnitude

NNN Transverse Ising model also has rapid oscillations with the amplitude of the peaks increasing and decreasing roughly every 100 time steps. Again, the oscillations observed exhibit behavior of periodic dynamics that are expected of non-dissipative systems.

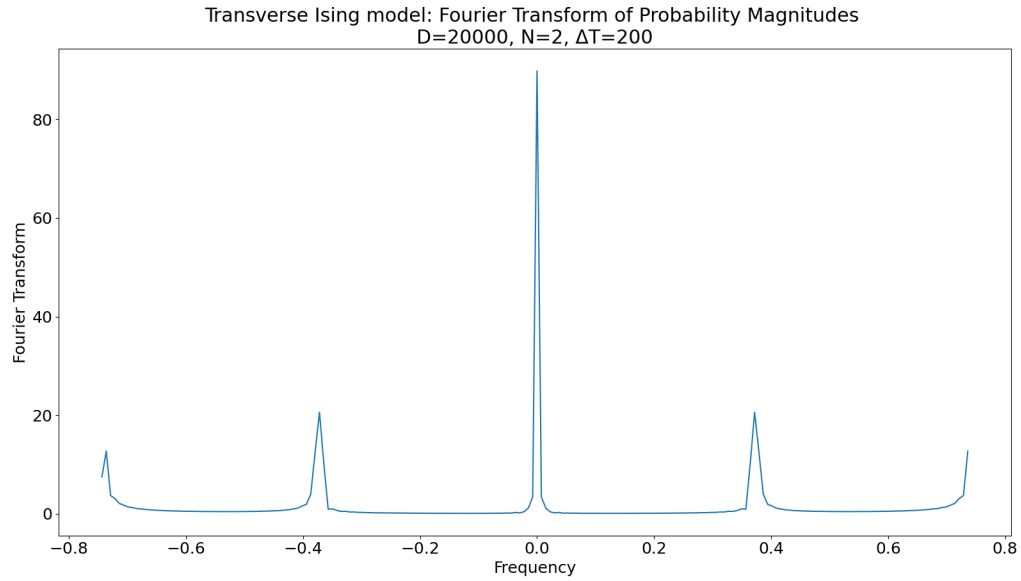


(c) Heisenberg Model Probability Magnitude

Heisenberg model oscillation contains spikes in  $|\Psi(t)|^2$  in a consistent pattern.

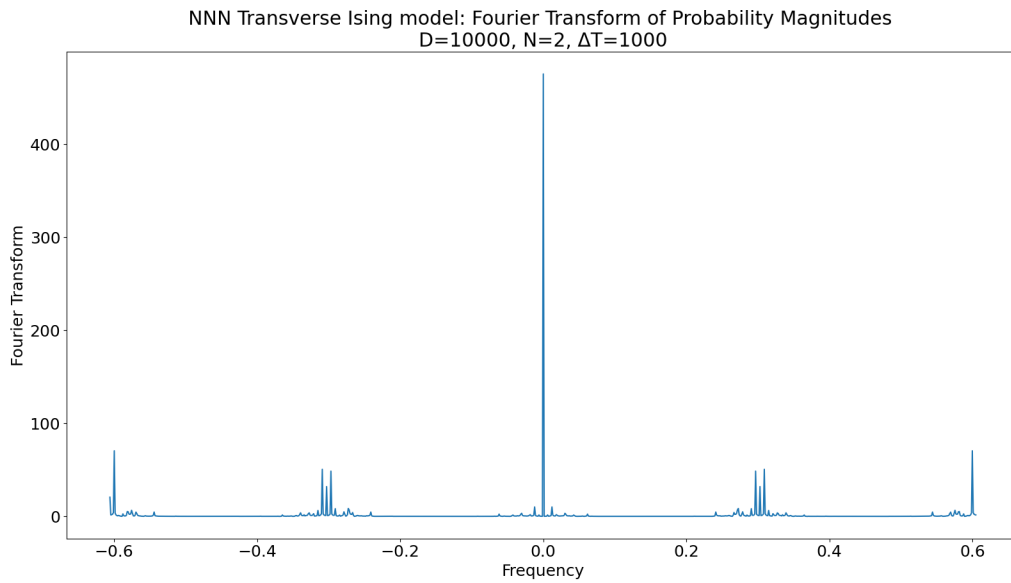
Figure 2.2: *Notes for entire Figure* - Time evolution of the magnitude of probability  $|\Psi(t)|^2$  for the a) Transverse Ising Model, b) NNN Transverse Ising Model, and c) Heisenberg Model. Periodic dynamics shown for each model match expectations for non-dissipative systems.

The time evolution of the probability magnitudes  $|\Psi(t)|^2$  for the Transverse Ising Model, NNN Transverse Ising Model, and Heisenberg Model demonstrates characteristic periodic dynamics expected from non-dissipative quantum systems, Figure 2.2. The Transverse Ising Model and NNN Transverse Ising Model both exhibit rapid oscillations, with the latter showing amplitude modulations over longer intervals. The Heisenberg Model displays a consistent oscillatory pattern with sharp spikes, reflecting the interplay of spin interactions. These results highlight the preservation of the quantum dynamics in each system we generate, consistent with theoretical expectations for closed quantum systems.



(a) Transverse Ising Model FFT peaks

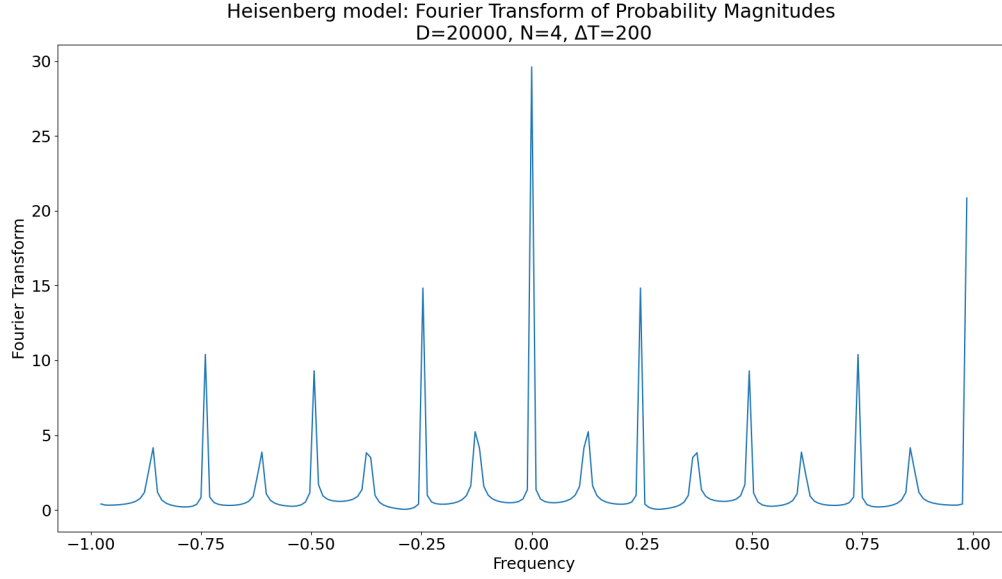
Notice that there are  $2^N = 4$  peaks clearly visible.



(b) NNN Transverse Ising Model FFT peaks

Notice that there are  $2^N = 4$  peaks clearly visible.

Irregularities around the peaks can be attributed to a lack of granularity in the data and variability introduced by increasing the number of interacting terms.



### (c) Heisenberg Model FFT Peaks

The peaks appear similar to the Transverse Ising model, but with increased frequency as there are more spins present in a basic  $2 \times 2$  lattice. The peaks appear more similar to those of the Transverse Ising Model FFT peaks as the models do not have NNN interactions.

Figure 2.3: *Notes for entire Figure* - FFT for the a) Transverse Ising model ( $D = 20000$ ,  $N = 2$ ,  $dT = 200$ ), b) NNN Transverse Ising Model ( $D = 10000$ ,  $N = 2$ ,  $dT = 1000$ ), and c) Heisenberg model ( $D = 20000$ ,  $N = 4$ ,  $dT = 200$ ). The models each have  $2^N$  peaks, which corresponds with the number of spins. We observe in the a) Transverse Ising model peaks that are symmetric, with the main dominant peak being at the zero frequency. Similarly, for the b) NNN Transverse Ising model, we observe the symmetric peaks, with a dominant peak at zero frequency. However, the b) NNN Transverse Ising model has numerous smaller peaks that are spread more widely. Finally, for the c) Heisenberg peaks, we observe 16 peaks that are widespread and not as sharp as the other two models. All of these results fit expectations and provide a rough verification of the data we use to train.

The FFT analysis of the Transverse Ising Model, NNN Transverse Ising Model, and Heisenberg Model reveals consistent patterns aligned with theoretical predictions for systems of  $N$  spins, Figure 2.3. In each case, the observed  $2^N$  peaks confirm the expected frequency components corresponding to the number of spins. The Transverse Ising Model and NNN Transverse Ising Model both exhibit symmetric peaks with a dominant zero-frequency component, indicative of coherent quantum dynamics. The NNN Transverse Ising Model, however, shows additional, less

pronounced peaks due to increased interaction complexity and reduced data granularity. The Heisenberg Model, with 16 widespread and less defined peaks, reflects the higher spin count. This subset of FFT results validates the models' expected behaviors and confirms the data's suitability for training purposes.

## 2.4 LSTMs

To test our LSTM model, we used the PyTorch library to create an LSTM model with a simple architecture with only two other linear layers to help with the input and decoding process. After finalizing the architecture, we performed small hyperparameter sweeps in order to optimize the model. We found 0.001 to be an efficient learning rate. We additionally explored minimal tuning of other hyperparameters, such as the number of layers, hidden units, and epochs - tuning to minimize training time constrained by final loss. We found that 2 LSTMs stacked together provided good results for accuracy and time. This should also allow for more complex patterns in our quantum systems to be recognized. We found that 128 hidden units was an effective size for our model in order to avoid overfitting while ensuring complex patterns can be learned. Next, with epochs, we found good results with 4 epochs. However, it is possible that we would have achieved better results by training for more epochs, which can be tested in the future. Overall, the LSTM model we created is relatively simple in order to avoid overfitting and avoid consuming too many computational resources.

Very little hyperparameter tuning was required as strong performance was achieved with the following hyperparameters. Unless otherwise stated, all the hyperparameters in the table [2.2](#) are the values we ran with the LSTM.

| Hyperparameter   | Value |
|------------------|-------|
| Train Test Split | 80%   |
| Batch size       | 32    |
| Epochs           | 4     |
| Learning Rate    | 0.001 |
| Linear Layers    | 2     |
| Hidden size      | 128   |
| Number of LSTMs  | 2     |

Table 2.2: Fixed hyperparameters for LSTM sweeps

## 2.5 Transformers

The Transformer can be trained exactly as the LSTM, where it is repeatedly used for batches across the data in the temporal dimension, with an arbitrary size for the prediction dimension. Therefore, just as for the LSTM, we train our Transformer across all generated transients predicting all  $2^N$  coefficients in the Hilbert space using  $c\%$  of a single 'period' (portion of  $J^{-1}$  over the data) and  $(1 - c)\%$  as the total amount to predict. We only include single periods in our prediction, using all timesteps of a transient for training. Training is done across transients in batches of 32 on 80% of our generated data, while 20% is left for validation. The encoder only transformer uses 4 encoder heads in a latent space of 64 dimensions, passed through 2 layers and 1 output layer. The hyperparameters are summarized in table 2.3. The input uses a positional encoding similar to the original sine based transformer [14].

Similarly to the LSTM, we found that minimal hyperparameter tuning was necessary to achieve strong performance. Therefore, unless otherwise stated, we use the hyperparameters in table 2.3 for training.

| Hyperparameter                          | Value  |
|-----------------------------------------|--------|
| Train Test Split                        | 80%    |
| Batch size                              | 32     |
| Epochs                                  | 4      |
| Learning Rate                           | 0.001  |
| Encoder Heads                           | 4      |
| Linear Layers                           | 2      |
| Linear Encoding Dimension               | 64     |
| P                                       | $10^4$ |
| Positional Encoding Dimension ( $d_k$ ) | 64     |

Table 2.3: Fixed hyperparameters for Transformer sweeps

## 2.6 Running Models

We run our models with a job array bash script that sweeps over available mixing values, spins, and context amounts and spawns all available jobs on the cluster. For all runs except the 3x3 Heisenberg model and the 8 spin Ising model, we use 4 CPU cores with 8 GB of RAM each. For the 3x3 Heisenberg model, due to the large Hilbert space size, we use 4 CPU cores with 24 GB of RAM each. For the 8-spin Ising model, we use 4 cores with 12 GB of RAM. We also repeat our training 5 - Ising models - and 3 - Heisenberg/Hubbard - times respectively across multiple random coupling matrices to provide more initial random states. We report the mean and standard deviations of our results across these repeated initializations. RAM restrictions prevent us from scaling to a higher number of spins in the Hubbard model, Heisenberg, etc., for our search. We are also restricted to only 4 epochs for efficient training that can be spread across the Zaratana system in a timely fashion. This is all to ensure we can provide as broad a search as possible for small-scale systems within reasonable timeframes.

The models are evaluated with their so-called MSE - Mean Square Error. This computes

the squared error in the model's prediction over time, averaging over a batch in training of size 32 transients and summing errors over all spin components. These models are all run, again, for 4 epochs. This indicates 4 total passes through all our training set; this is produced by the 20,000 transients in each dataset. The data batches are stochastically sampled as is typical in batch-based gradient descent.

## Chapter 3: Results

### 3.1 Overview

As mentioned in Chapter 2, we perform a large sweep over environmental settings and physical model parameters, including the employed Frobenius norm, context provided to the model, and more. Recall from Chapter 2 that context sets the percentage of time steps that are fed to the model to use. It also specifies the amount of data that remains to be predicted by our models.

Note that in the following plots, we will have data corresponding to a Frobenius norm of zero, which technically indicates a zero matrix. However, in our implementation of the code, we use this special case to indicate that there is no anisotropy among the interaction terms (i.e.,  $J_{xx}=J_{yy}=J_{zz}$ ).

Model performance will be characterized by observing (a) the distribution of the final generalization validation error in MSE and (b) the distributions of the MSE during training. By understanding both of these distributions, and how they are affected by physical parameters, we will be able to determine the primary physical model characteristics that impede either training time or final generalization performance.

To observe these, we create four figures for each physical model. As accuracy is paramount to our claims in this thesis, we start by presenting a heatmap of the final validation performance

across all model runs with all model settings. This gives an excellent high level view of the performance of our runs. **Crucially**, we *must* present validation heatmap data first as our training distributions (size 3) are too small to visualize nicely. Under conclusions made from these validation heatmaps, like in Figure 3.1, we can plot the training distributions of models by Frobenius norm and spins for the transformers and LSTM models *combined for all* context values. This means that the distributions we display can increase from size 3 to size 30, and the number of lines in the graph becomes manageable to analyze. In boxplots like Figure 3.4, we summarize the overall performance of our models. Unlike the heatmaps, we employ conclusions we make later about context to group our validation data distributions by context value, as explained earlier. Switching from viewing heatmaps to boxplots gives us new insight into the behavior of the final validation distribution as a function of spins and Frobenius norm. As will be discussed later, it allows us to observe a shrinking in the spread of model validation distributions (standard deviation decreases, average stays roughly the same). These boxplots grant insight into the sudden 'phase change' in model performance where LSTMs suddenly outperform Transformers once the displayed distributions squeeze too much. These boxplots allow us to view the sudden shift in model performance as a critical point being reached in the size of these validation distributions.

We repeat these figures for all physical models with only minor changes. All minor changes are addressed in figure captions as necessary and in small text summaries presented at the start of each section.

### 3.1.1 Anisotropic Transverse Ising Model

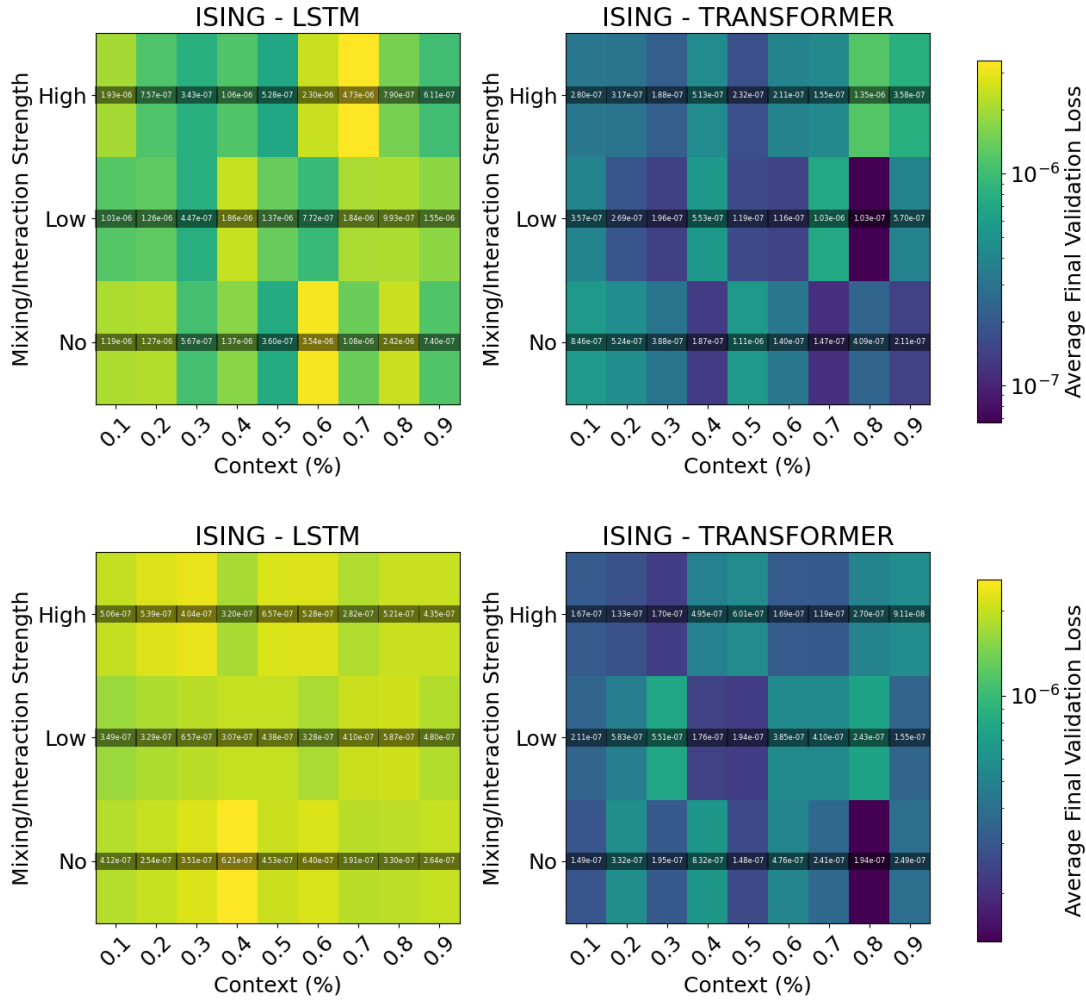


Figure 3.1: *The results are continued on the next page.* Heat maps for average final validation loss using LSTM and Transformer models. Each row corresponds to 2, 4 spins, respectively, from top to bottom. The text values present in each cell correspond to the standard deviation of the average final validation loss over the 3 runs done for each section. Note that Frobenius Norm = 0 in our case means that there is no anisotropy.

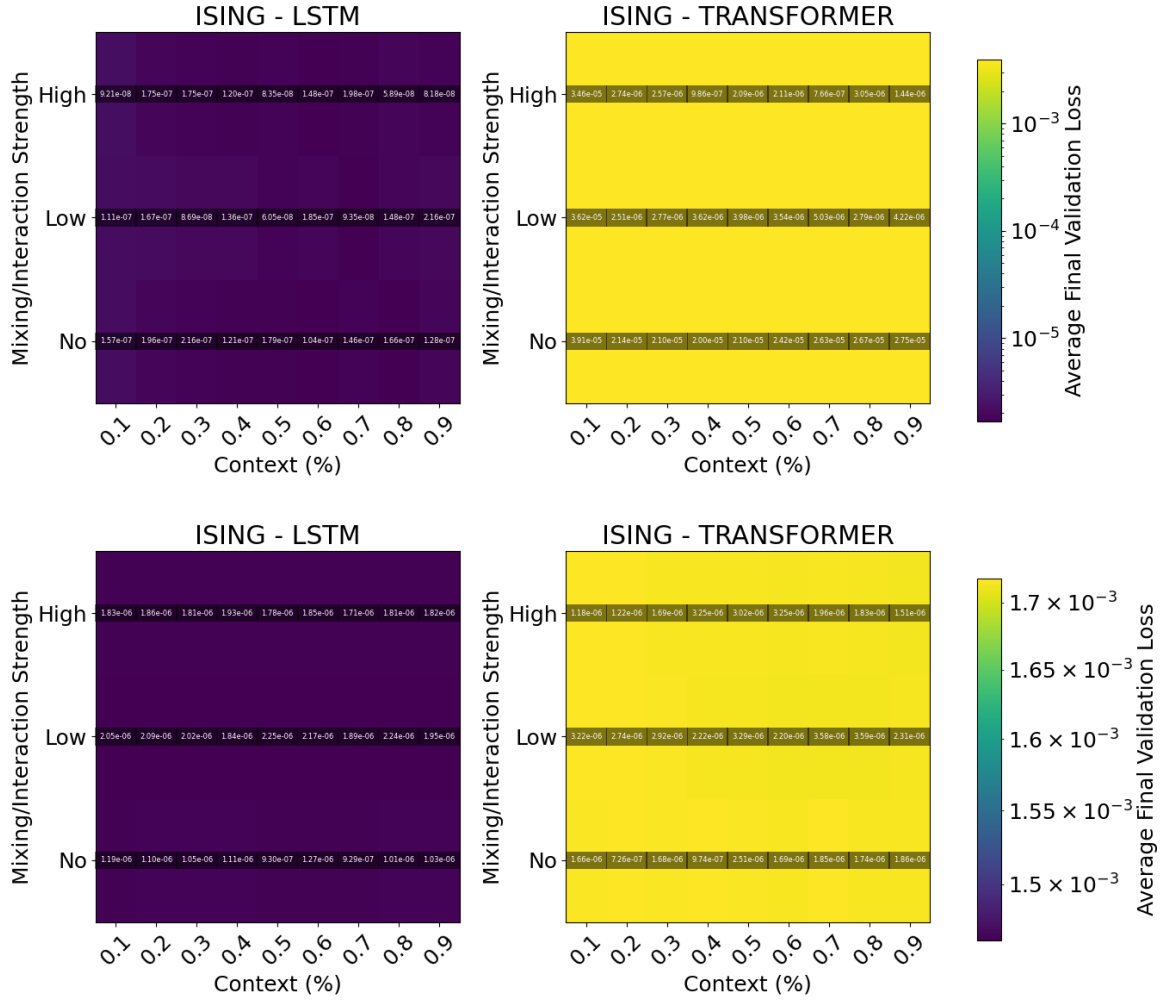


Figure 3.1: Continuation of figure 3.1. Rows correspond to 6, 8 spins, respectively, on this page.

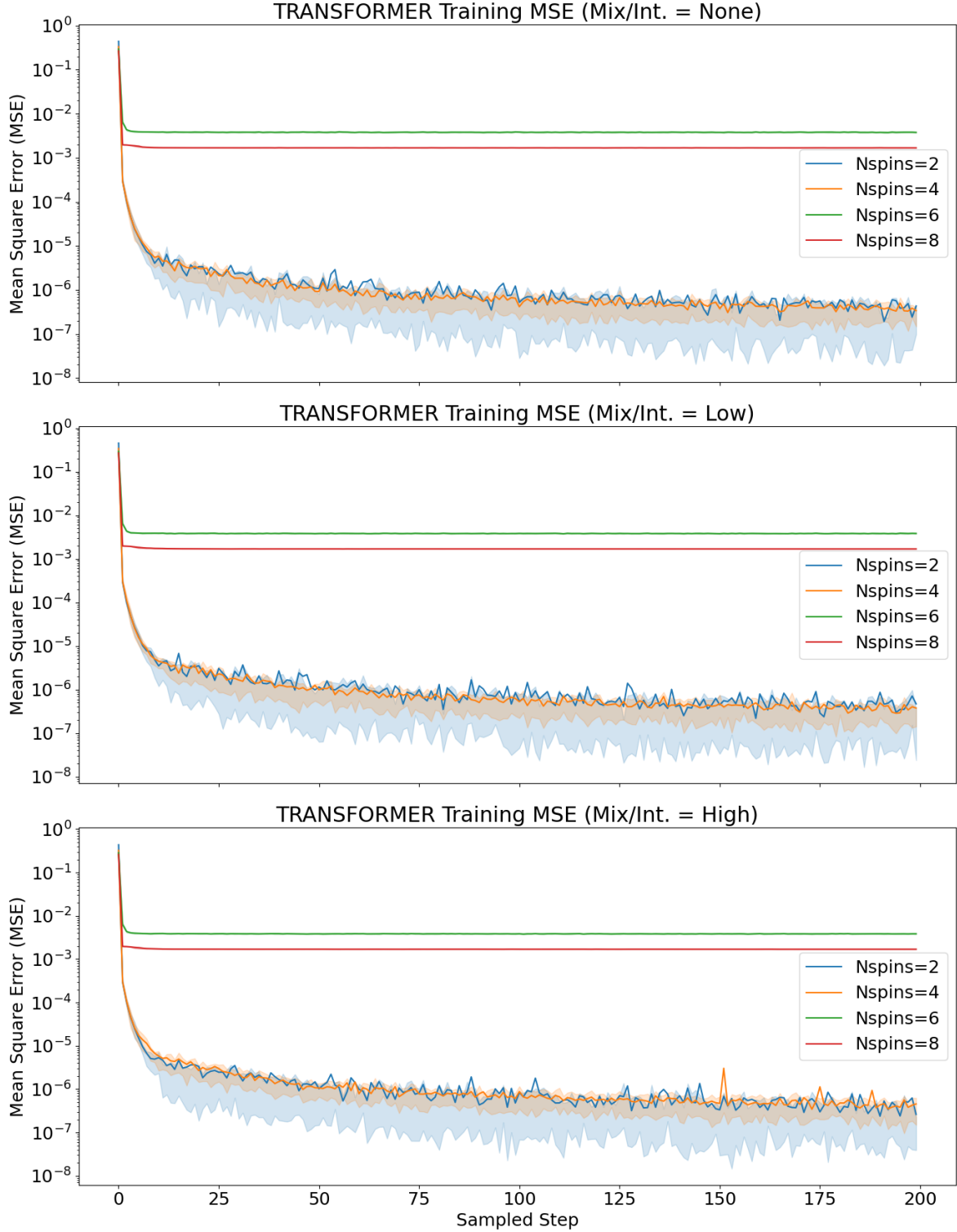


Figure 3.2: Transformer training MSE over 200 sampled steps (every 50 steps in training) for 2, 4, 6, and 8 spins with Frobenius Norm 0, 2, and 4 from top to bottom. Shaded values indicate the 25-75% quartile range, with the line indicating the average. The distribution being shaded includes all repeated runs for the given settings, *including* all context values based on results in [3.1](#).

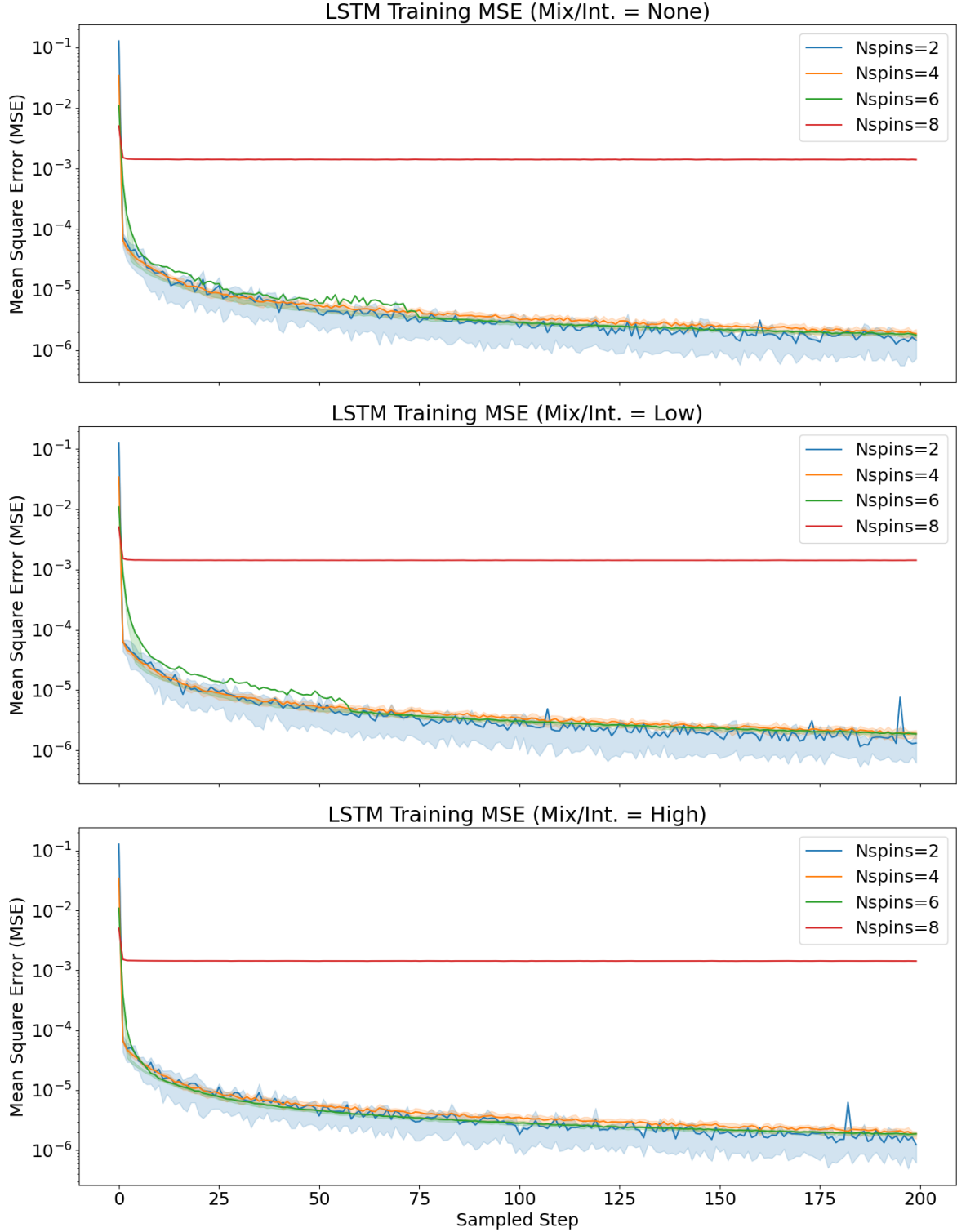


Figure 3.3: LSTM training MSE over 200 sampled steps (every 50 steps in training) for 2, 4, 6, and 8 spins with Frobenius Norm 0, 2, and 4 from top to bottom. Shaded values indicate the 25-75% quartile range, with the line indicating the average. The distribution being shaded includes all repeated runs for the given settings, *including* all context values based on results in 3.1. 6 spin models outperform the transformer figure results.

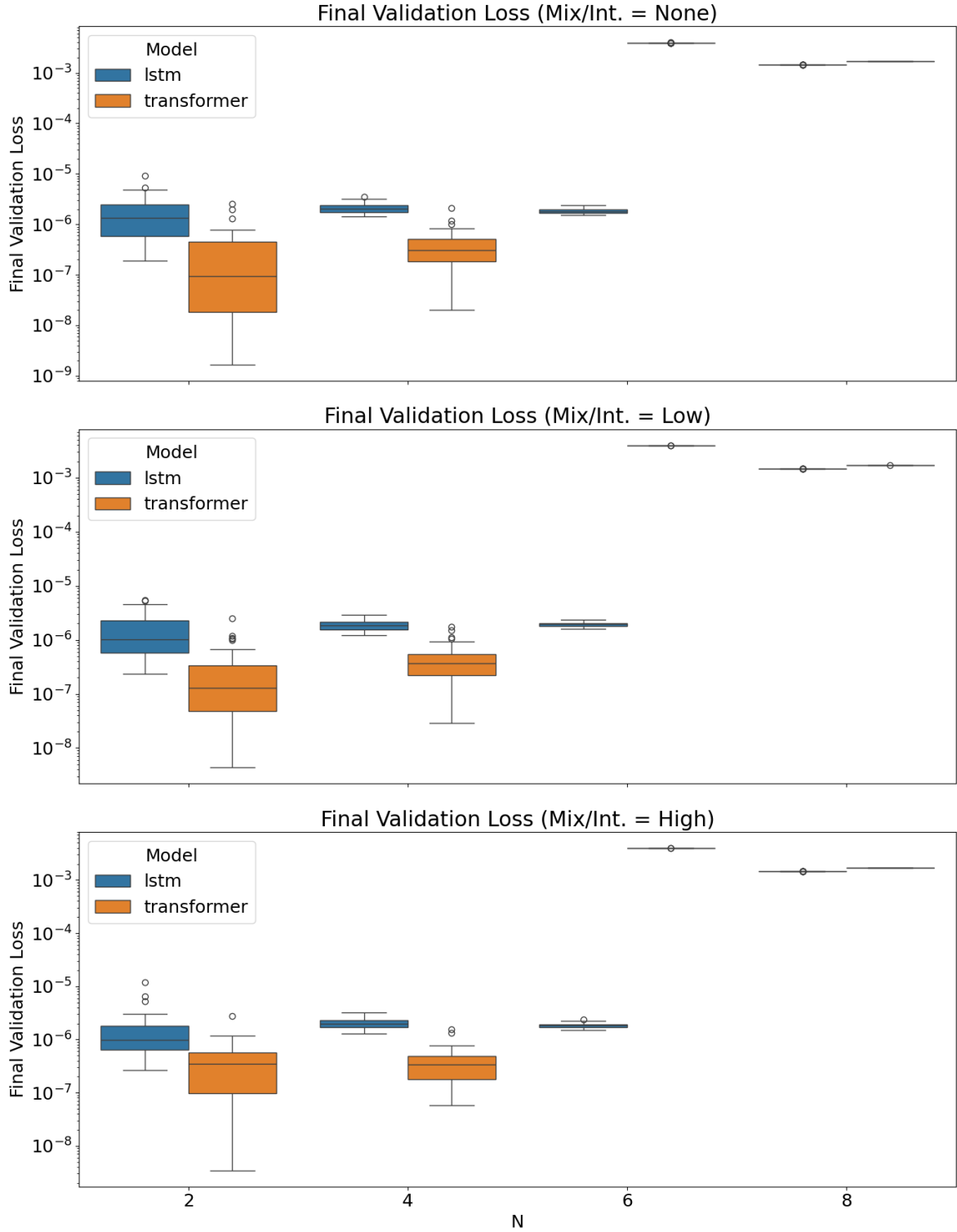


Figure 3.4: Box plot distributions displaying the min, max, quantiles, and outliers for LSTM and Transformer models at different fixed spin counts and Frobenius norms. These results are purely for the Anisotropic Ising Model. Observe how steady the transformer distribution is until a sudden flip at high spin counts. Like in earlier figures, the distribution includes all runs at these fixed settings (i.e., all context values are put together).

Figures 3.1, 3.2, 3.3, and 3.4 cover the effect of the context amount, Frobenius norm, and spin counts on the training distributions across LSTM and Transformer models both during training and for the final validation loss. As can be seen in 3.1, the Frobenius norm setting range we consider, as well as the context provided to the model, negligibly affect the final validation performance. The differences in the final results are of similar size to the standard deviations in the final validation loss (see Figure 3.1 caption). So while we consider the mean and standard deviation of a training ‘batch’ of ML models (group of ML models trained repeatedly on new initializations) as only including models with identical context settings and Frobenius norms, we lump all models with the same norm together in the same distributions in Figures 3.2, 3.3, 3.4. This same reasoning is repeated in later sections.

### 3.1.2 Anisotropic Ising Next-Nearest Neighbor Model

Figures 3.5, 3.6, 3.7, and 3.8 cover the effect of the context amount, Frobenius norm, and spin counts on the training distributions across LSTM and transformer models both during training and for the final validation loss. We see very similar results to those in the previous section. Context and Frobenius norm seem to have little effect on the performance of the models. Like in figure 3.4, we observe a ‘tightening’ of the distributions of training results until the model performances flip and the final loss dramatically increases.

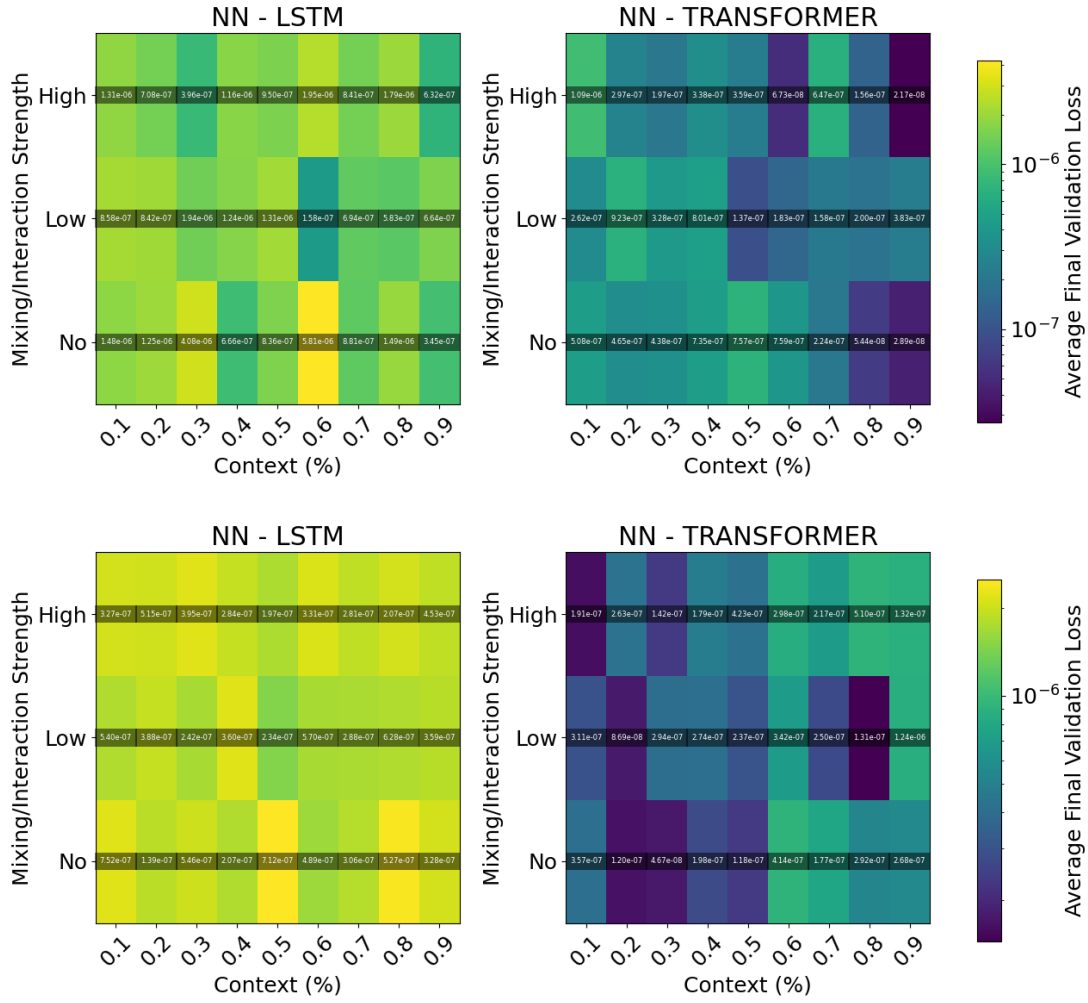


Figure 3.5: *Results are continued on the next page.* Heat maps for average final validation loss using LSTM and transformer models on the anisotropic next-nearest neighbors Ising model. Each row corresponds with 2 and 4 spins, respectively, from top to bottom. The text values present in each cell correspond to the standard deviation of the average final validation loss over the three runs done for each section. Observe similar results to Figure 3.1.

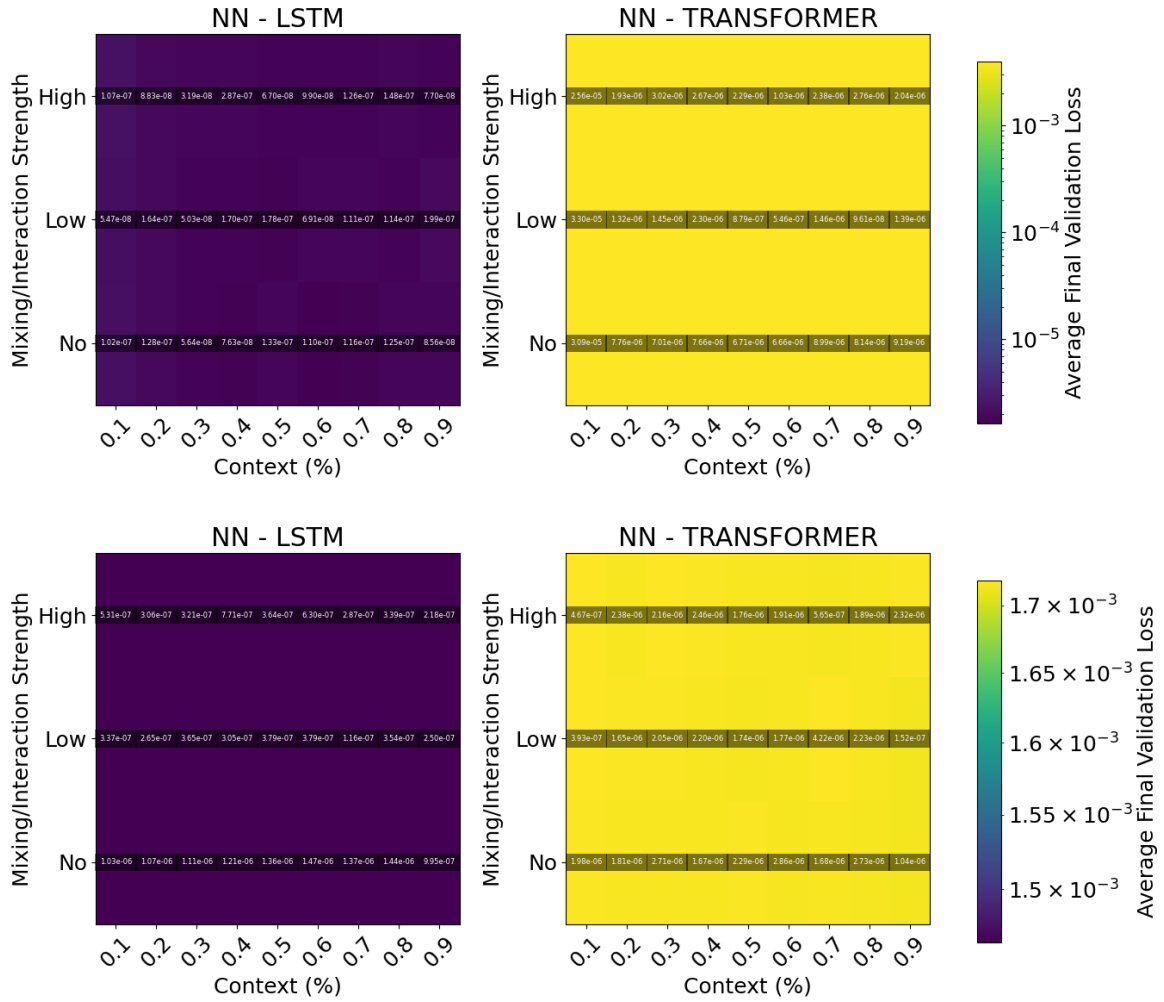


Figure 3.5: Continuation of result in figure 3.5.

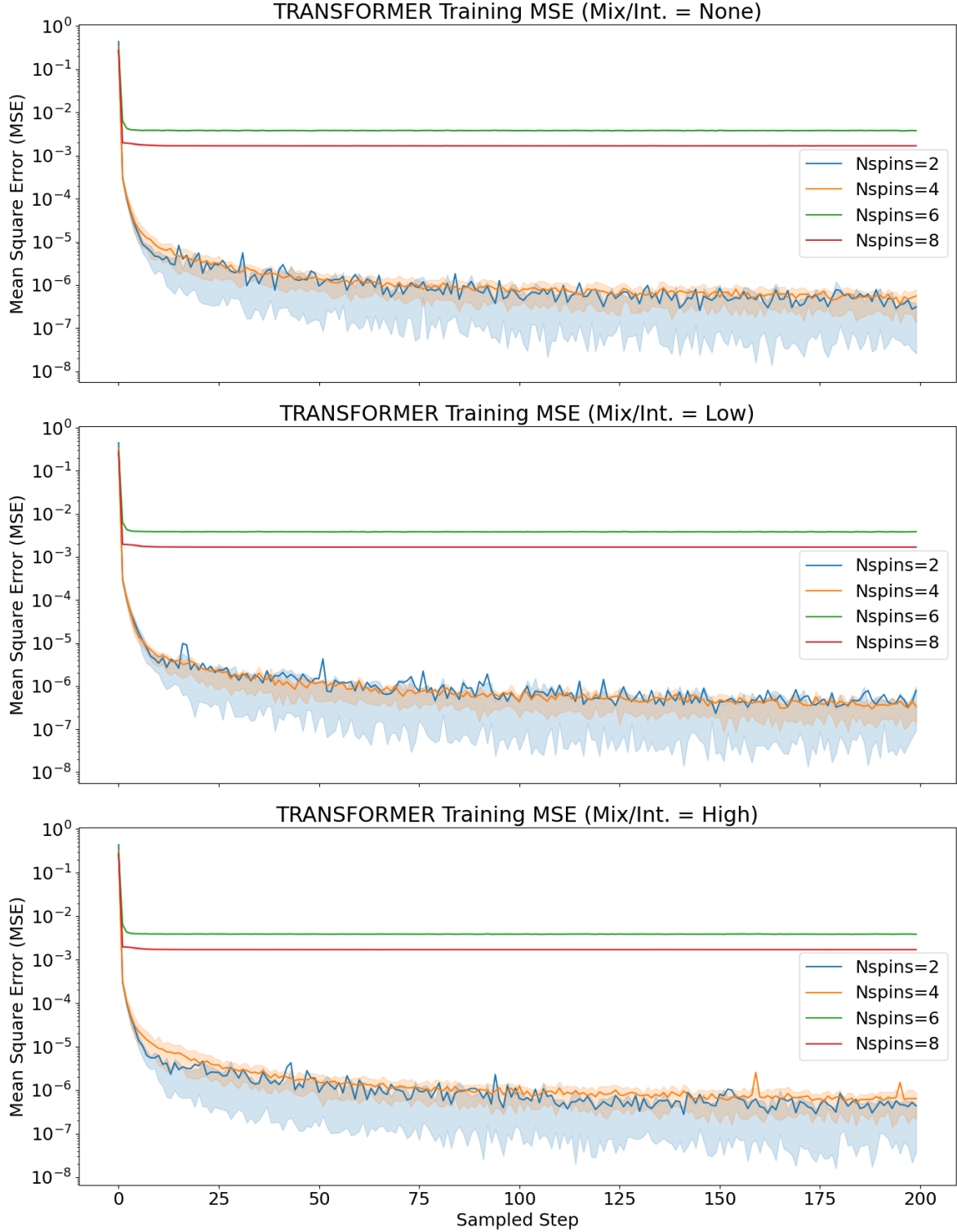


Figure 3.6: Transformer training MSE over 200 sampled steps (every 50 steps in training) for 2, 4, 6, and 8 spins with Frobenius Norm 0, 2, and 4 from top to bottom. Shaded values indicate the 25-75% quartile range, with the line indicating the average. The distribution being shaded includes all repeated runs for the given settings, *including* all context values based on results in figure 3.5. These are very similar results to figure 3.2.

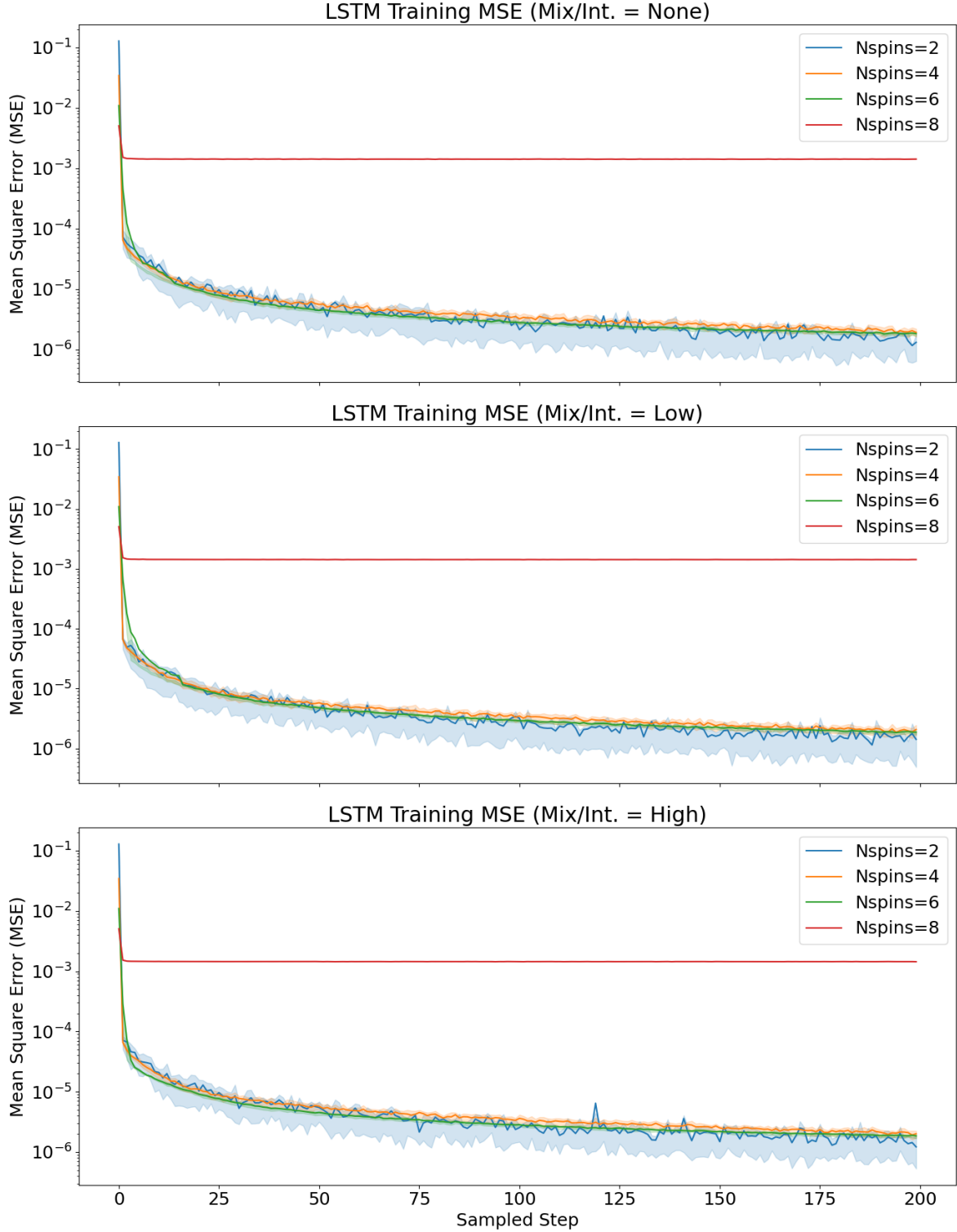


Figure 3.7: LSTM training MSE over 200 sampled steps (every 50 steps in training) for 2, 4, 6, and 8 spins with Frobenius Norm 0, 2, and 4 from top to bottom. Shaded values indicate the 25-75% quartile range, with the line indicating the average. The distribution being shaded includes all repeated runs for the given settings, *including* all context values based on results in figure 3.5. We observe very similar results to figure 3.3.

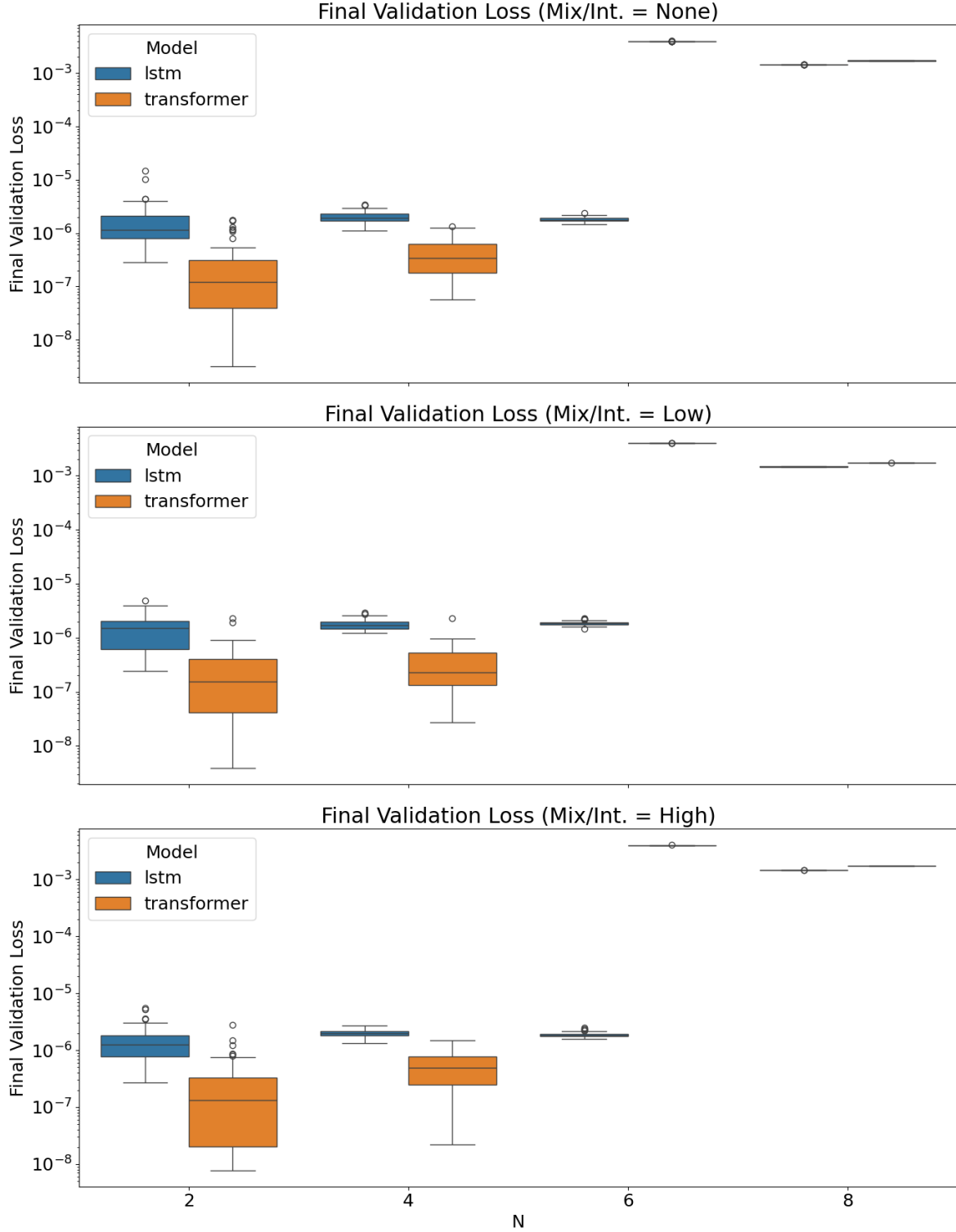


Figure 3.8: Box plot distributions displaying the min, max, quantiles, and outliers for LSTM and Transformer models at different fixed spin counts and Frobenius norms. These results are purely for the Anisotropic next-nearest-neighbor Ising Model. Observe how steady the transformer distribution is until a sudden flip at high spin counts. Like in earlier figures, the distribution includes all runs at these fixed settings (i.e., all context values are put together).

### 3.1.3 Heisenberg Square Lattice

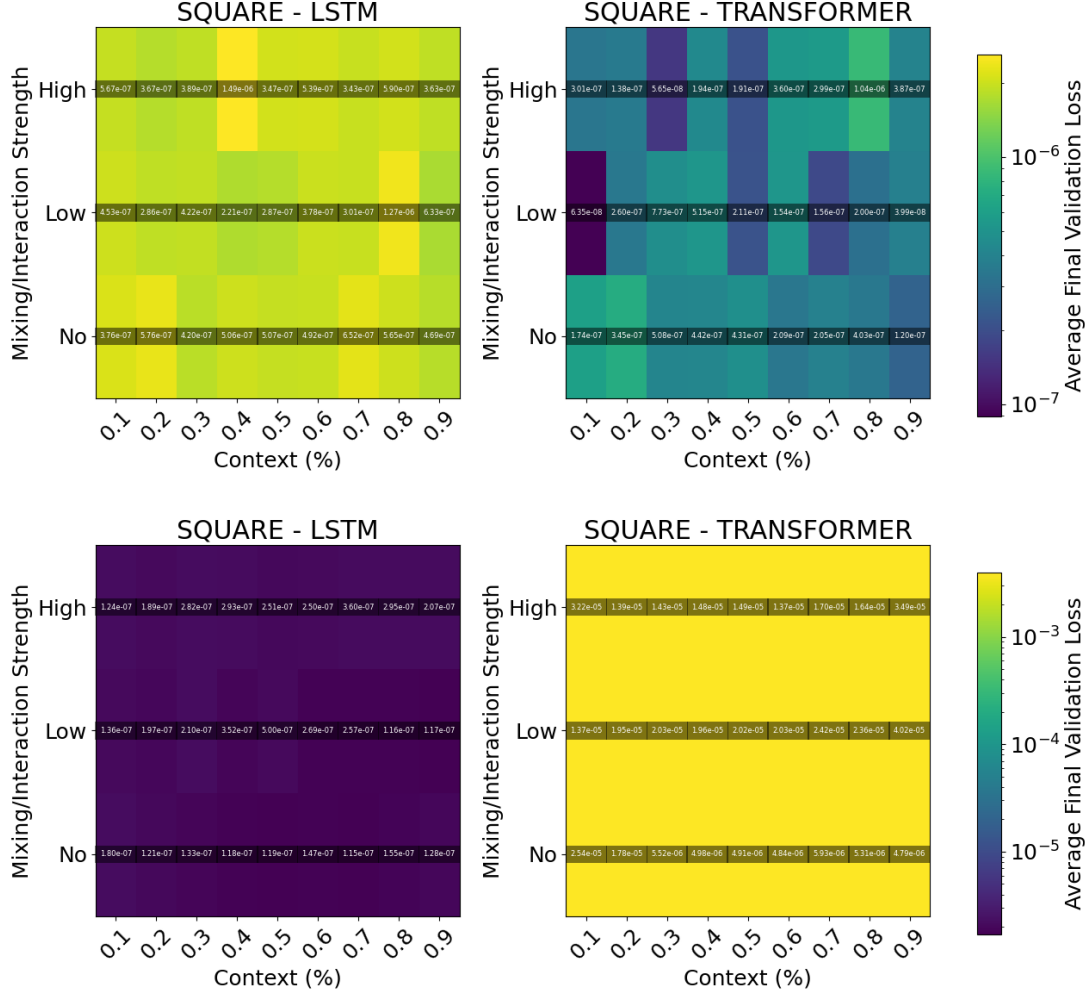


Figure 3.9: *Results are continued on the next page.* This figure setup is identical to prior sections like in figure 3.1. In this case, however, we only present runs for  $2 \times 2$ ,  $3 \times 2$  (note  $2 \times 3$  results were roughly identical), and  $3 \times 3$  grids in that order.  $3 \times 3$  is presented on the next page. These results are strictly for the Heisenberg square lattice grid. Just like in figures 3.1 and 3.5, we see a flip in model performance at 6 total spins. This is true even in a square lattice structure. Due to the computational complexity of the triangular model and the amount of required RAM, as well as our available cluster resources, we present a reduced context sweep for the  $3 \times 3$  Square grid.

Figures 3.9, 3.10, 3.11, and 3.12 cover the effect of the context amount, Frobenius norm, and spin counts on the training distributions across LSTM and transformer models both during training and for the final validation loss. Considering the fixed interaction pattern for the Heisen-

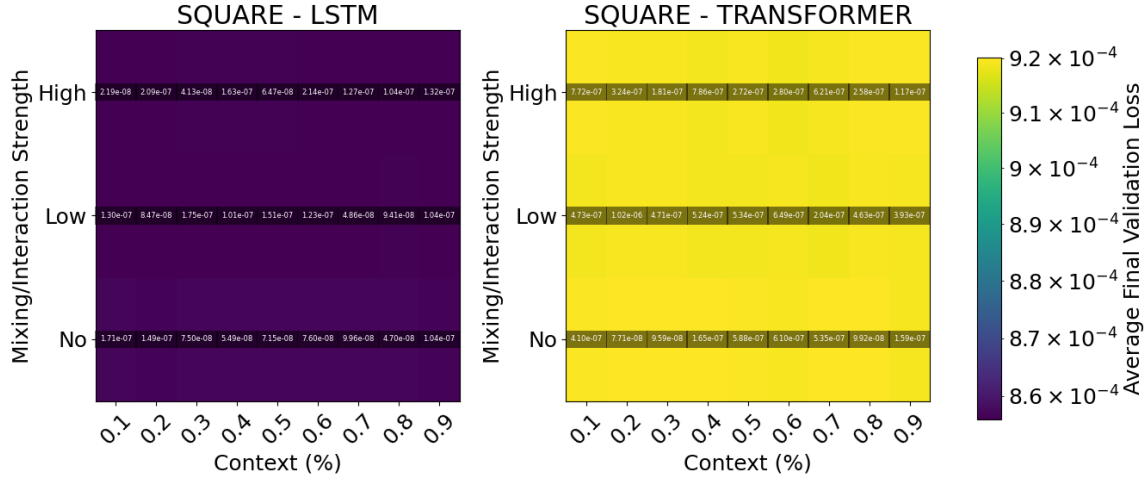


Figure 3.9: Continued results from figure 3.9. Here we show  $3 \times 3$  grid results.

berg model, we note that the Frobenius norm here is a knob by which *allowed* interaction strength can be controlled. This will hold true for the rest of the following models, triangular Heisenberg and Hubbard.

Once again, we see results that are very similar to those in the previous section. Context and Frobenius norm seem to have little effect on the performance of the models. Like in figure 3.4, we observe a 'tightening' of the distributions of training results until the model performances flip and the final loss dramatically increases. Curiously, we observe this flip again at 6-spins. Considering that the interaction model is completely different, including both isotropic spin interaction and a square lattice, the equivalent point of model performance flipping is surprising.

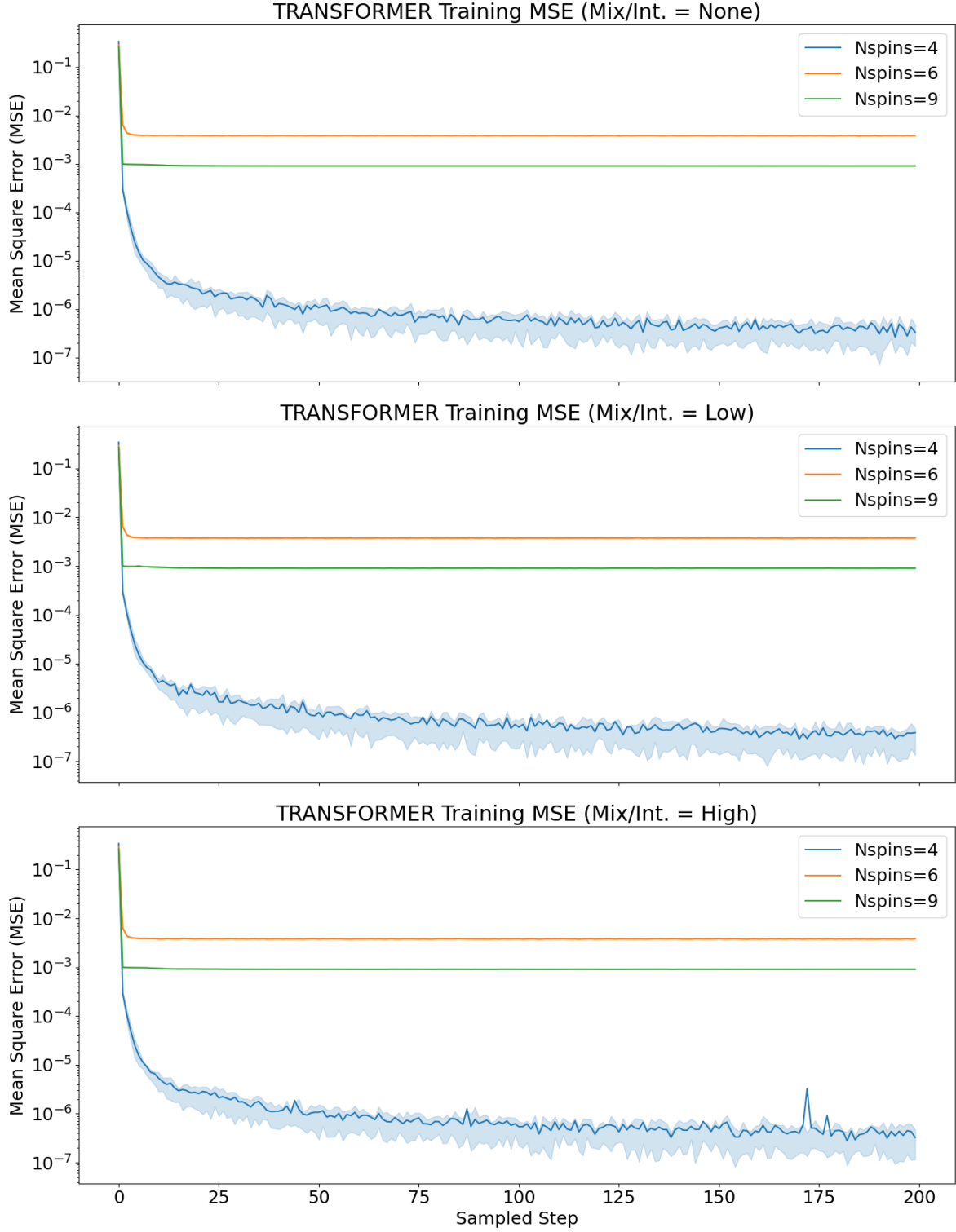


Figure 3.10: This section is identical to the setup in previous figures, yet these are the results for the Heisenberg square lattice transformers alone. We observe a similar drop in performance at 6-spins, though, notably, an increase in performance towards 9-spins. This is unlike what was observed in, say, figure 3.2. These results are very consistent across their training distributions.

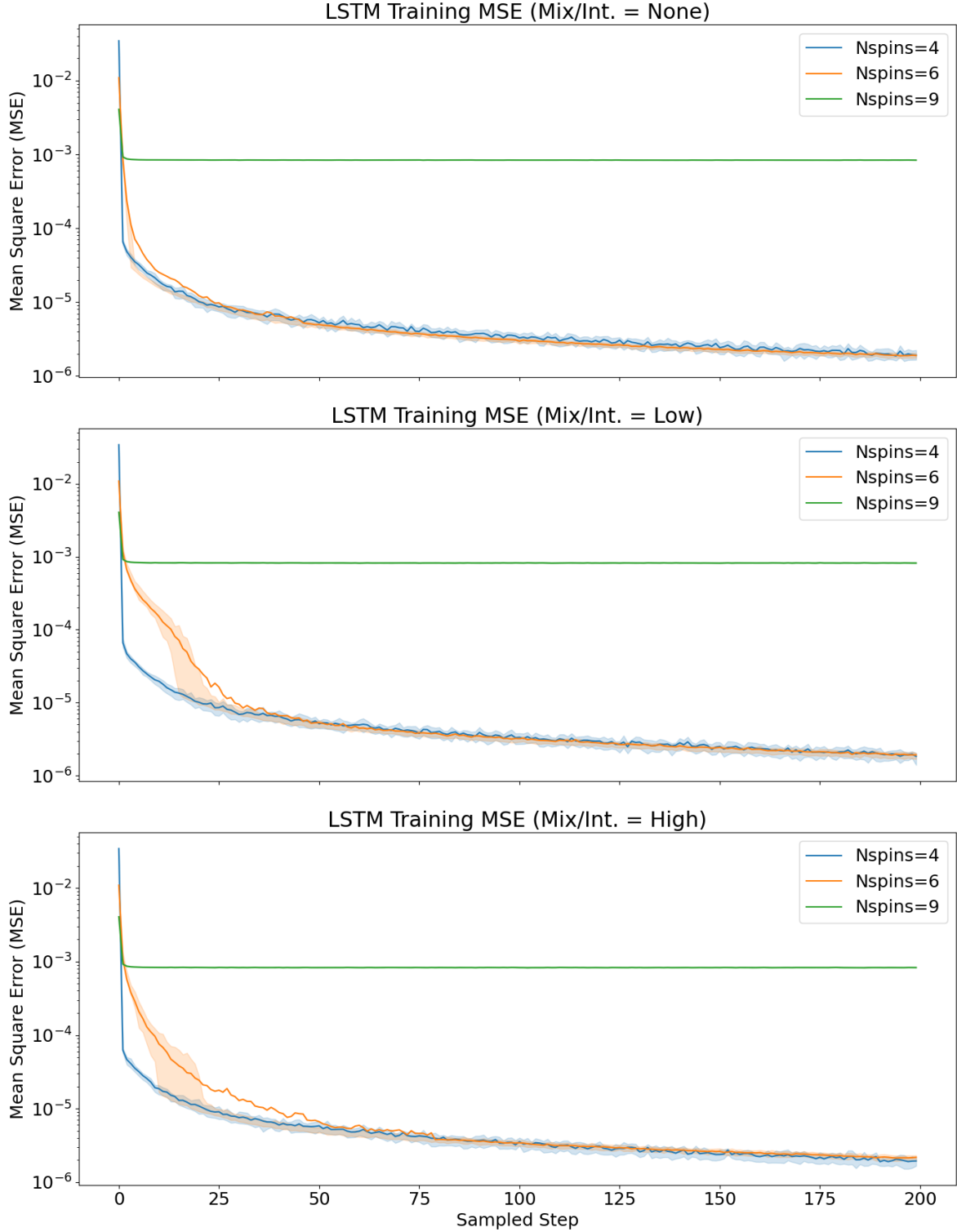


Figure 3.11: This section is identical to the setup in previous figures, yet these are the results for the Heisenberg square lattice LSTM models alone. We observe a similar drop in performance at 6-spins, though, notably, an increase in performance towards 9-spins. This is unlike what was observed in, say, figure 3.3. Like the Transformer results, the 9-spin models perform better here than the 6-spin models.

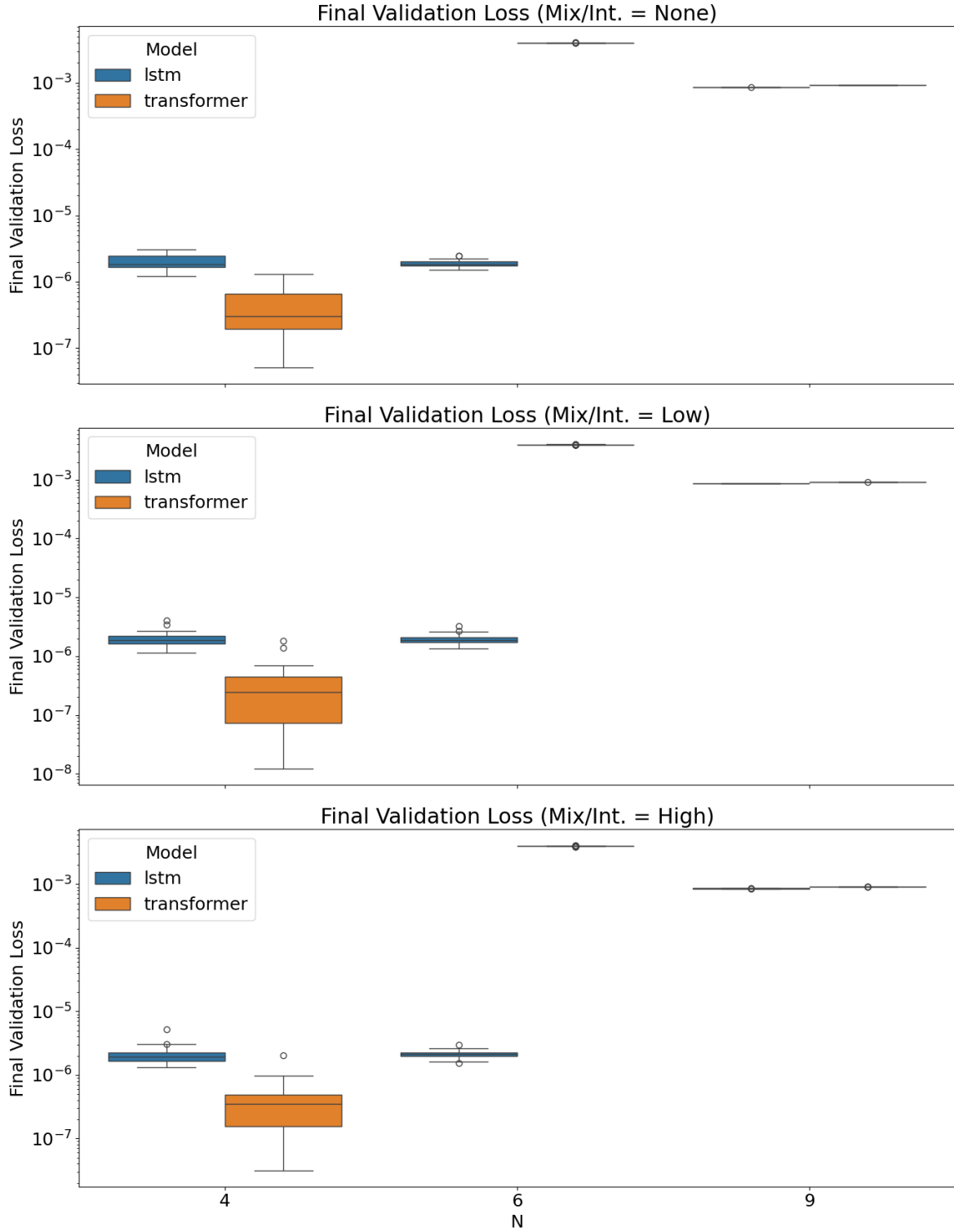


Figure 3.12: This section is identical to the setup in previous figures. Once again, the spin counts correspond to the relevant lattices ( $2 \times 2 = 4$ ,  $3 \times 2 = 6$ , *etc.*). We still observe the squeezing, but with less granularity in our results before  $N = 6$  we are unable to observe squeezing in the transformer's distribution before it flips.

### 3.1.4 Heisenberg Triangular Lattice

Figures 3.13, 3.14, 3.15, and 3.16 cover the effect of the context amount, Frobenius norm, and spin counts on the training distributions across LSTM and transformer models both during training and for the final validation loss. Once again, we see very similar results as in the previous section, despite now testing a *triangular* lattice. We do also observe this curious flip in model distributions at the same 6-spin point as in the square lattice. The similarity of the two implies that the triangular lattice is not significantly different from the square lattice.

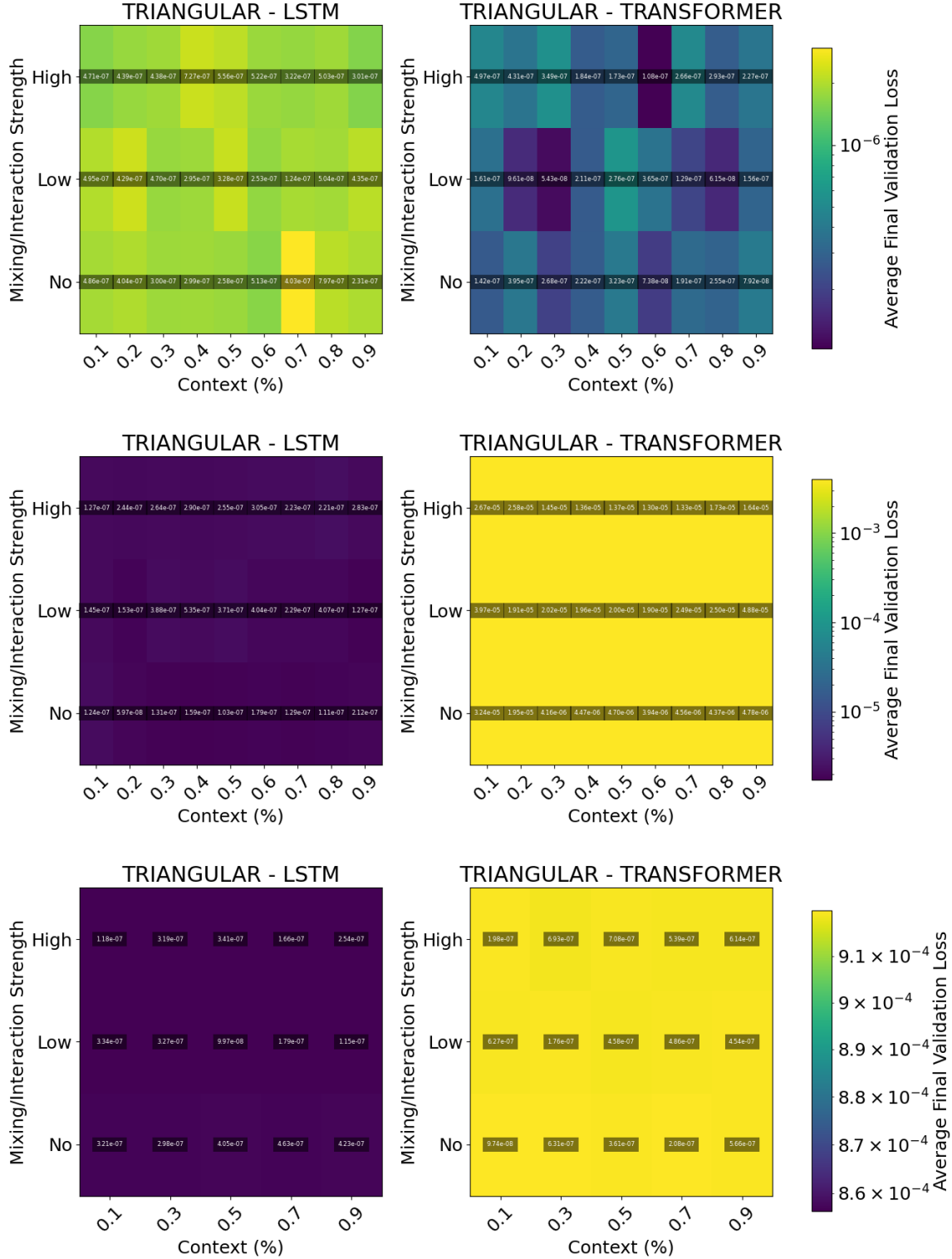


Figure 3.13: This figure setup is identical to prior sections like in figure 3.9. These results are strictly for the Heisenberg triangular lattice grid. Due to the computational complexity of the triangular model and the amount of required RAM, as well as our available cluster resources, we present a reduced context sweep for the  $3 \times 3$  Triangular grid. Here we present all  $2 \times 2$ ,  $3 \times 2$ , and  $3 \times 3$  results together.

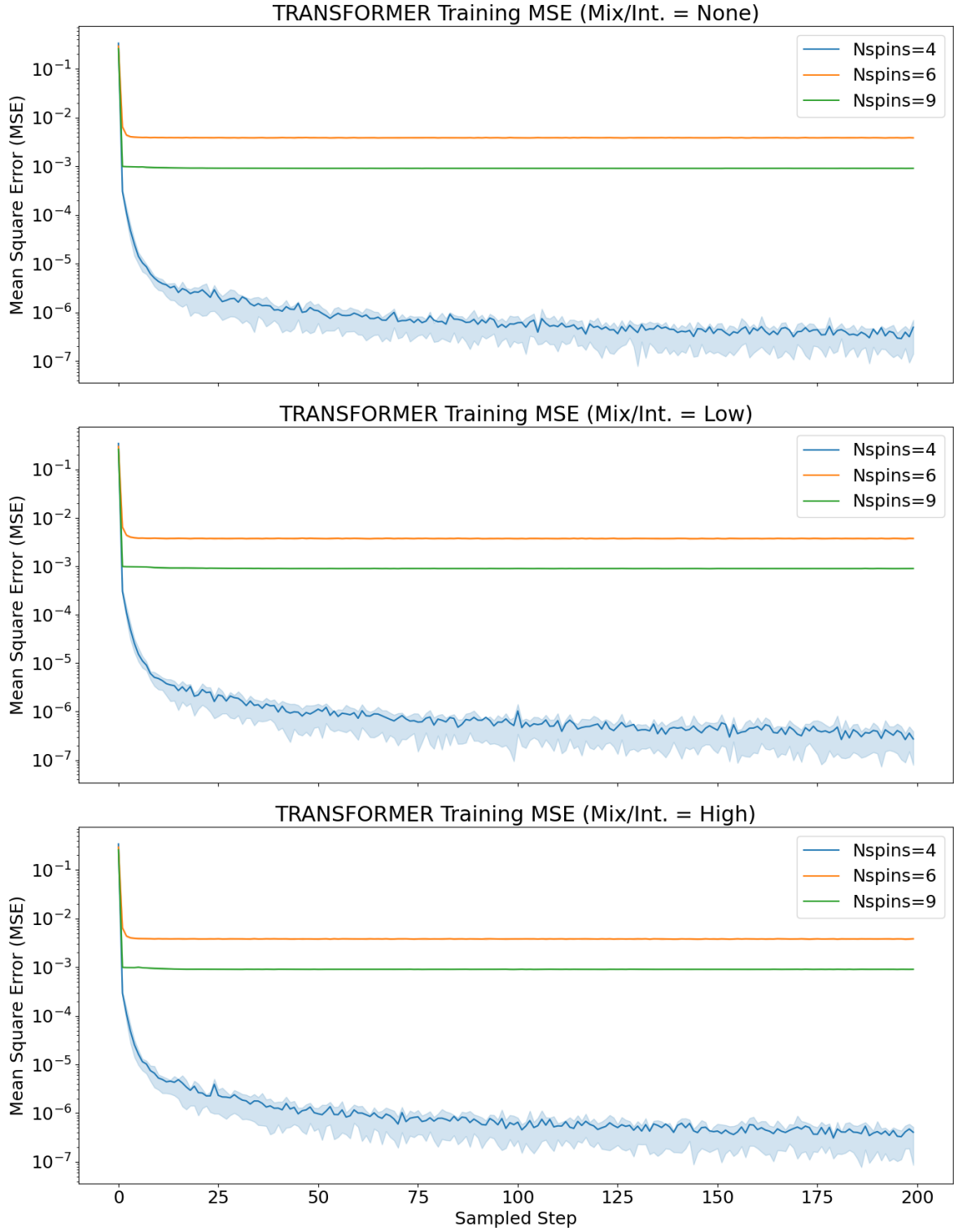


Figure 3.14: This section is identical to the setup in previous figures, yet these are the results for the Heisenberg Triangular lattice transformers alone. We observe a similar drop in performance at 6-spins, exactly like the square lattice results in figure 3.10.

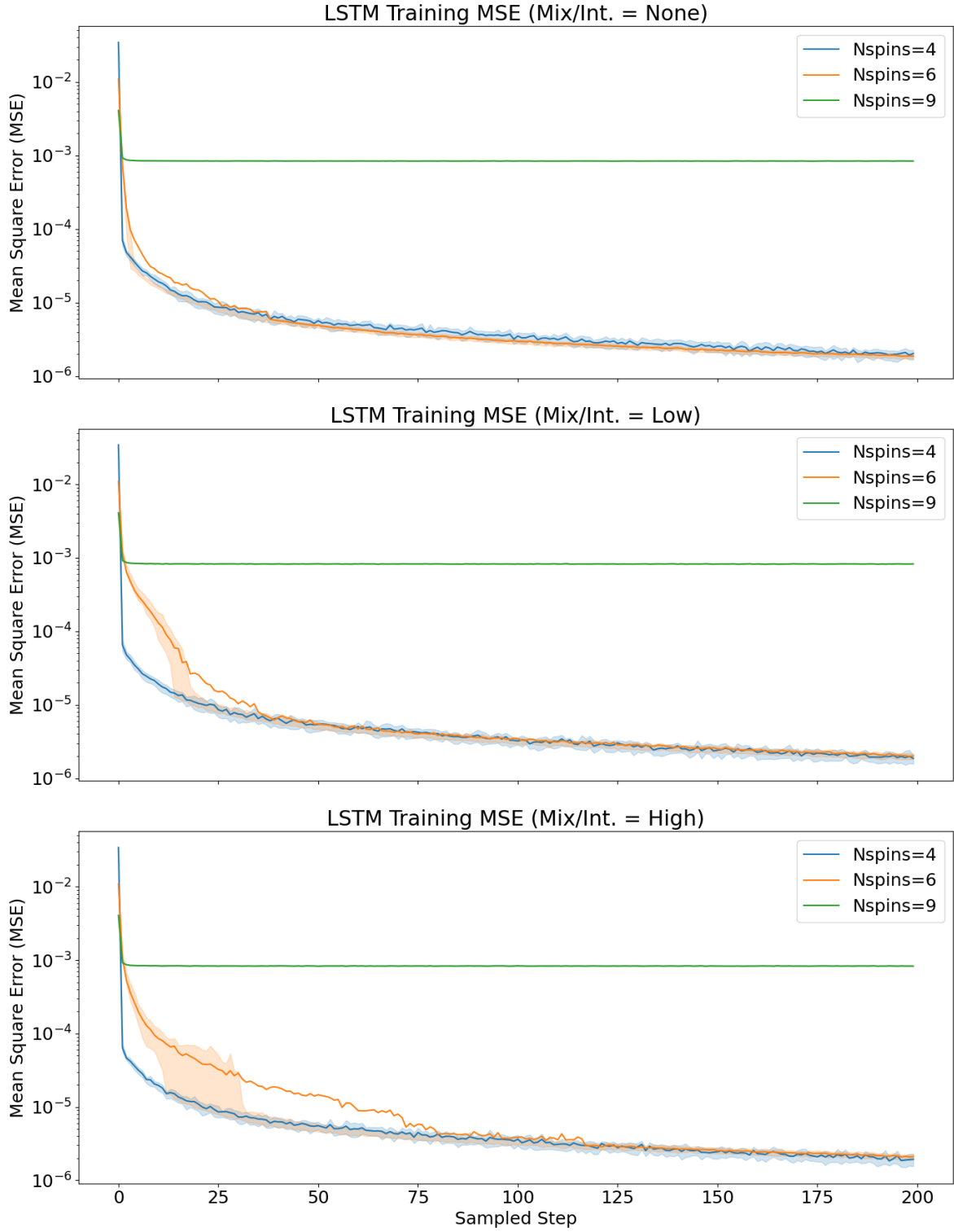


Figure 3.15: This section is identical to the setup in previous figures, yet these are the results for the Heisenberg Triangular lattice LSTM models alone. We observe a similar drop in performance at 6-spins, exactly like the square lattice results in figure 3.11.

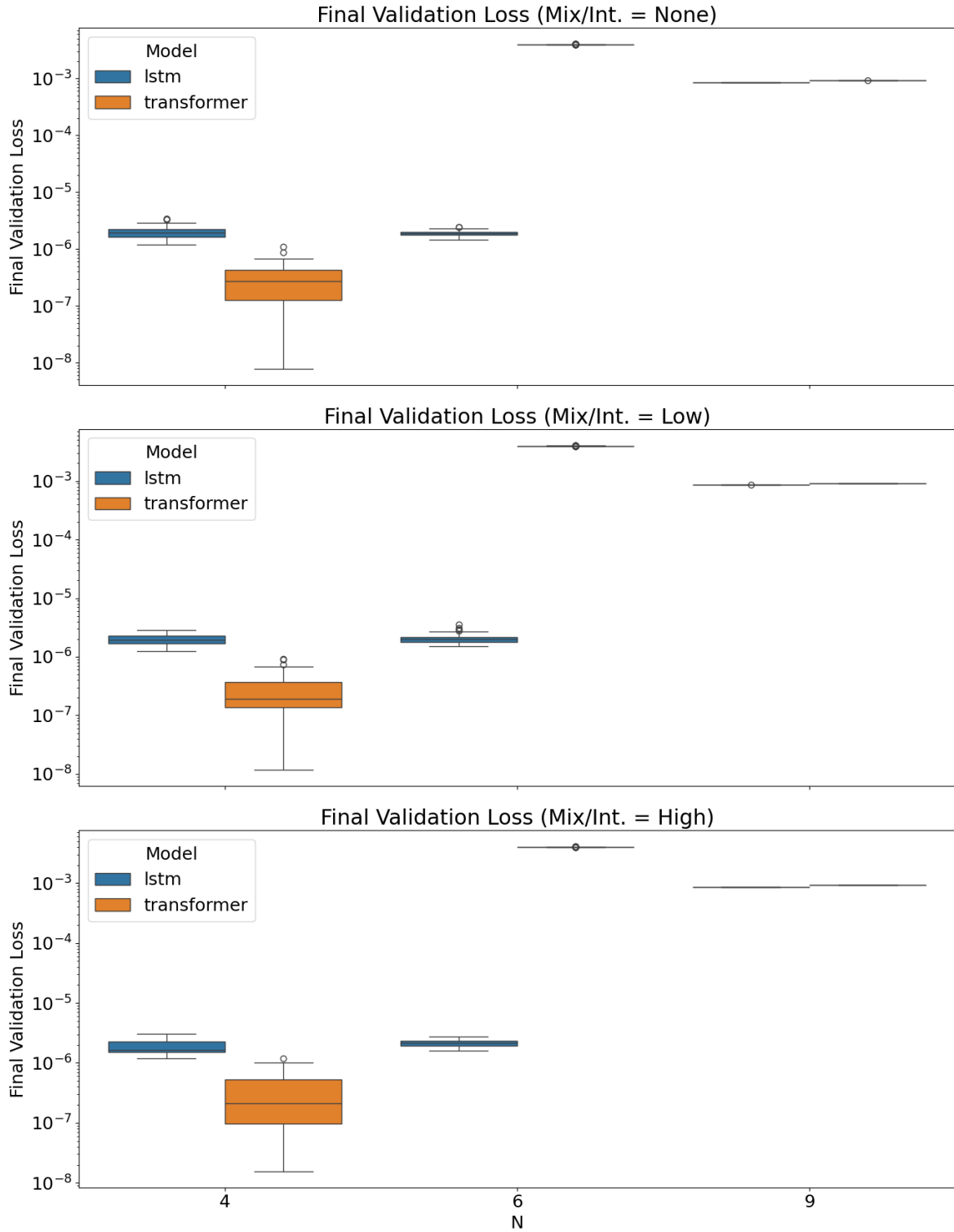


Figure 3.16: This section is identical to the setup in previous figures. As in figure 3.12, we are visualizing the box plot distributions for our three tested lattices in the triangular case. Note that all results are equivalent to what was presented in figure 3.12.

### 3.1.5 1D Hubbard Model

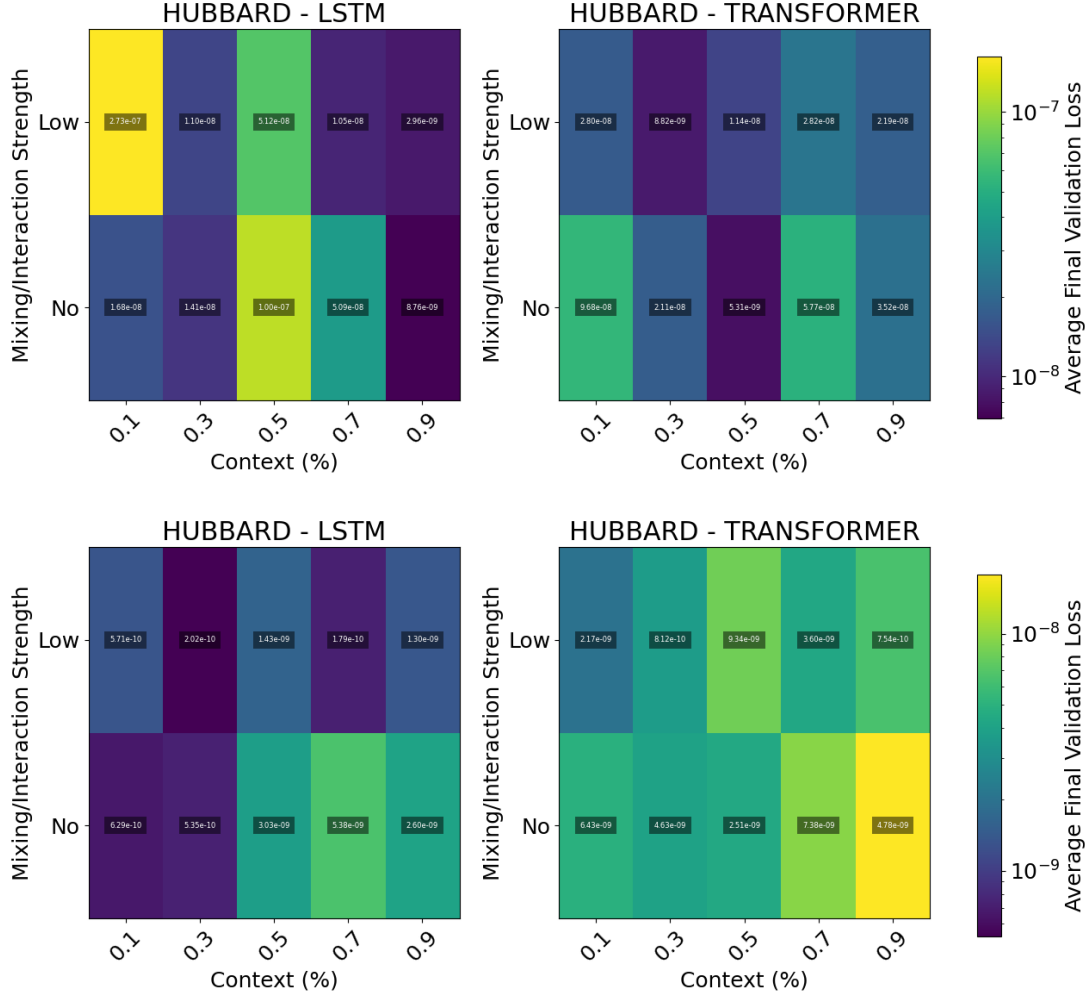


Figure 3.17: This plot is exactly alike in structure to figure 3.1. We do present a reduced context sweep due to the computational complexity of the Hubbard model - scaling by  $4^n$ . Further, we note that the standard deviation in the boxes shown indicates the variation in colors, which is likely due to random statistical fluctuations in training - not true dependence on context or Frobenius norm. We are limited to showing results for 2 and 4 spins, respectively (top to bottom), in the Hubbard model due to this scaling. Curiously, we do not observe a flipping of model performance despite the number of values to predict far exceeding that for 6-spins  $4^4 > 2^6$ . We also observe lower loss for the higher spin count system.

The Hubbard model results shown in figures 3.17, 3.18, 3.19, 3.20 are exciting. We were limited from testing 6-spins due to computational complexity, but we observe here on the 4-spin

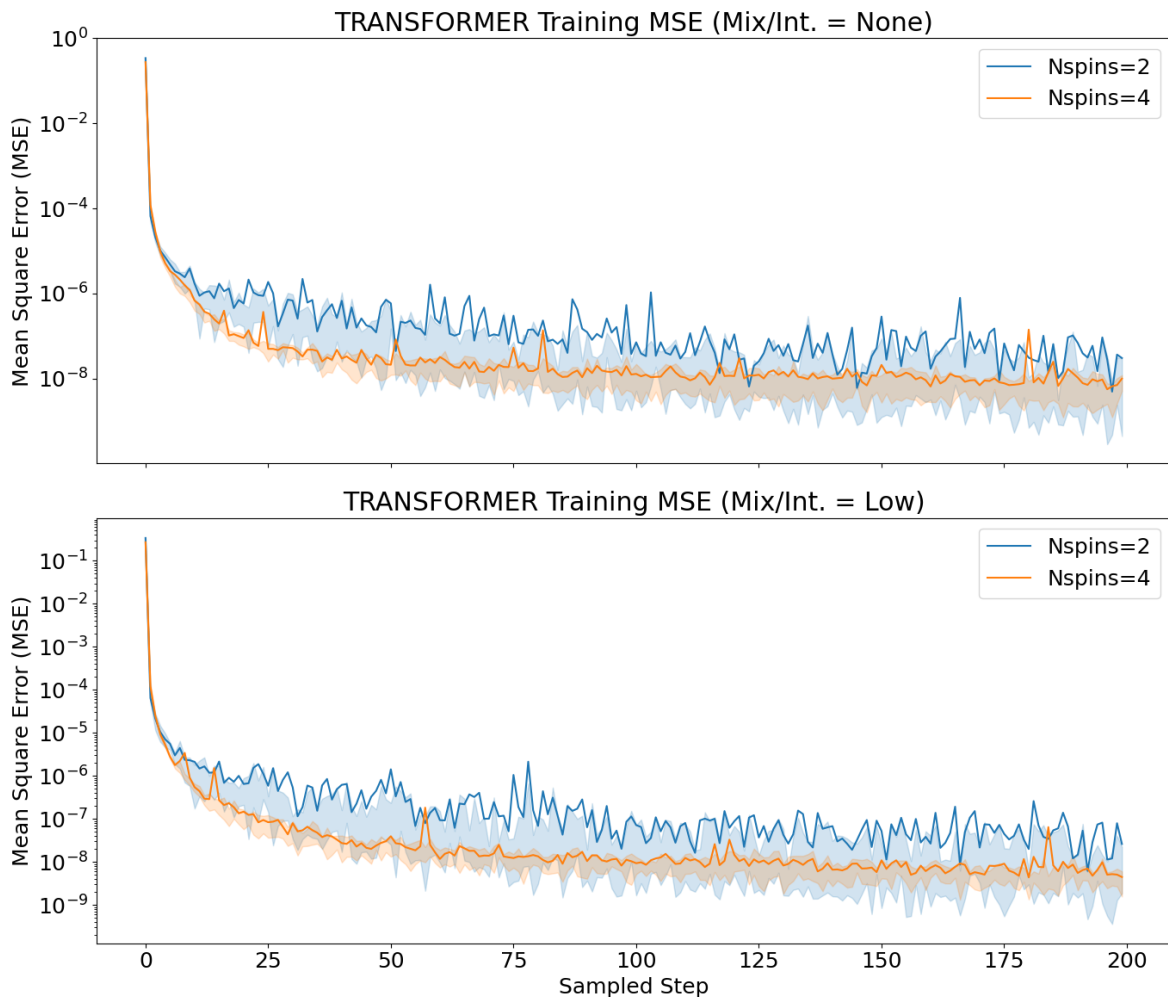


Figure 3.18: This plot is exactly alike in structure to figure 3.2, though this shows only transformer results for the Hubbard model. Both models perform well and are roughly consistent. Curiously, the higher spin model is more stable in its distribution than the lower spin model.

system an *increase* in performance. Observe especially in figure 3.18, 3.19 that the distribution of model performance becomes significantly tighter with increasing spins, and *lower*. This is to the point that for the LSTM, the 4 spin distribution is entirely below the majority of the transformer training distribution (see figure 3.20) by the end of training.

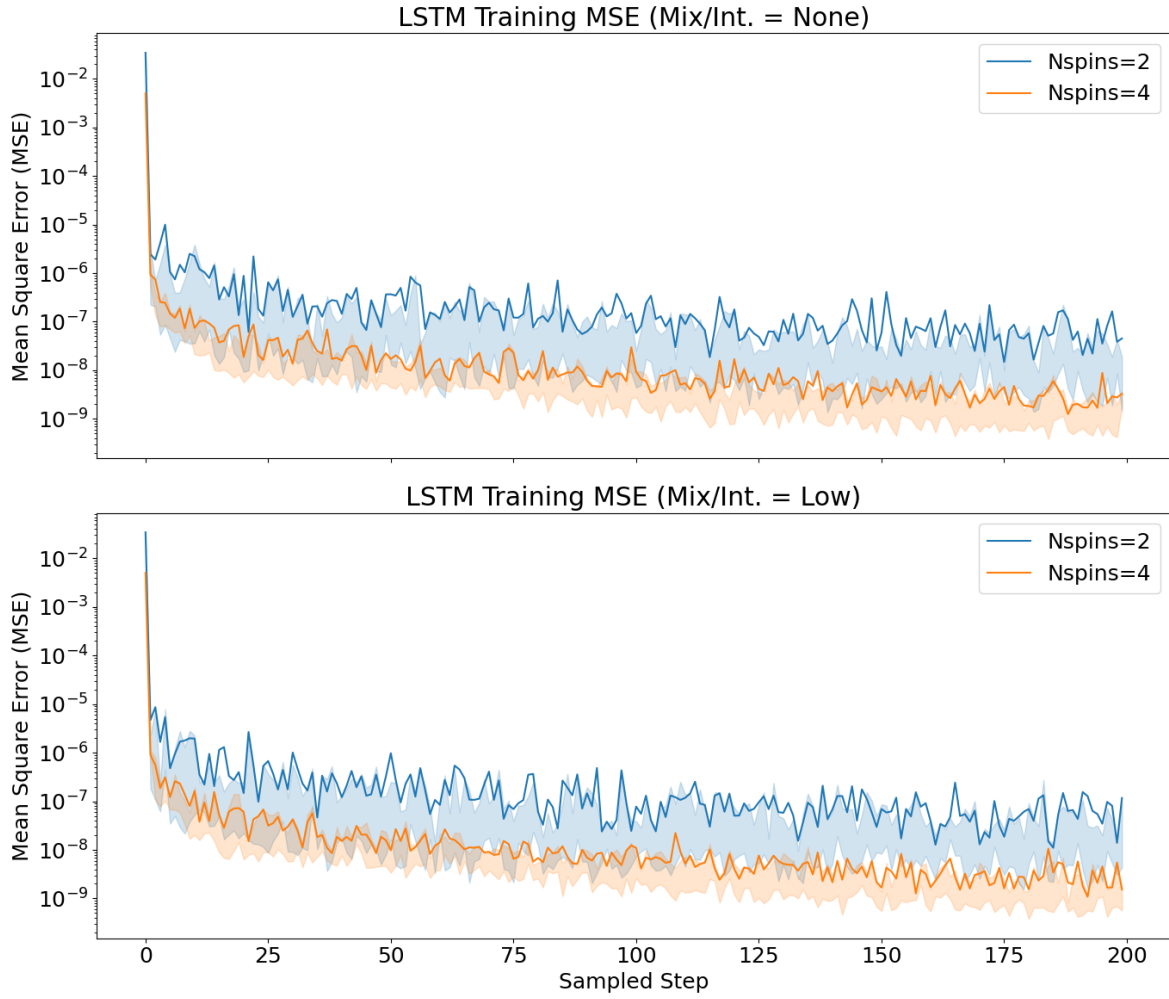


Figure 3.19: This plot is exactly alike in structure to figure 3.3, though this shows only LSTM model results for the Hubbard model. Both models perform well and are roughly consistent, as shown in figure 3.18. The distributions here visually appear to be tighter and lower than in figure 3.18, though the overlap is still large.

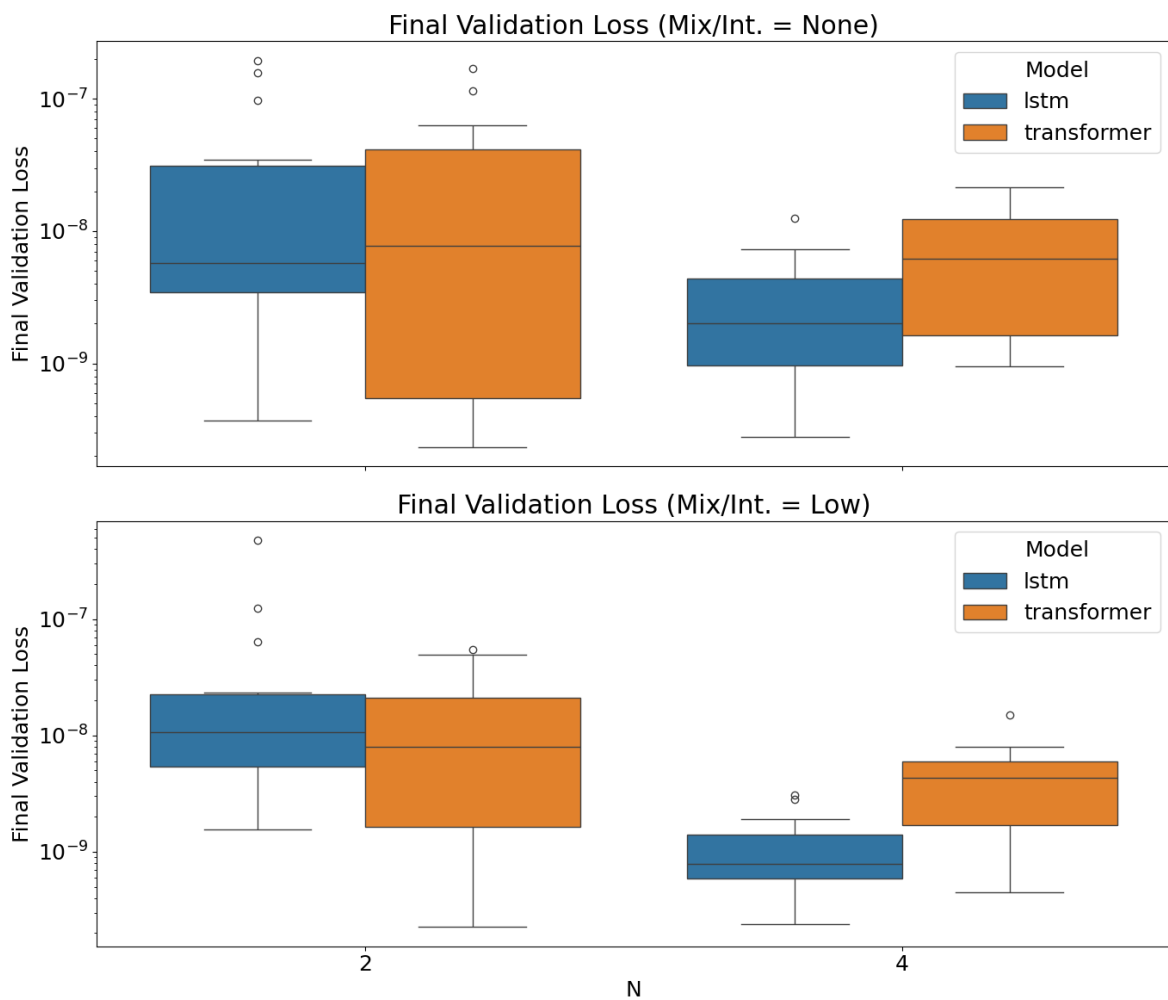


Figure 3.20: Box plot distributions for the Hubbard model results, exactly alike in structure to figure 3.4. Curiously, we not only note that the distributes squeeze as spins increase, but also *improve*.

## Chapter 4: Conclusions

### 4.1 Anisotropic Transverse Ising Model

Our results in Chapter 3 show that it is possible to simulate the time evolution of transverse Ising model spin systems for two and four spins to degrees of accuracy on the order of  $10^{-5}$  and  $10^{-6}$ , using the LSTM and transformer frameworks we developed respectively [see figures 3.2, 3.3]. For both models, training MSE serves as a reliable predictor of final validation performance. The training MSE for spins 2 and 4 for both models follow similar trajectories and reach training MSE on the order of  $10^{-5}$  and  $10^{-6}$  for the LSTM and transformer models, respectively. 6 spins with the LSTM model also follow this pattern. These results highlight the consistency between the models' training performance and final validation accuracy. As we increase the number of spins, and thus the complexity of the physical model being predicted, the ML models see orders of magnitude increases in training MSE and final validation loss, reaching between  $10^{-3}$  and  $10^{-2}$ . While accuracy is a crucial metric, understanding the limitations and behavior of these models under increasing complexity is equally essential. In the following discussion, we detail common trends we observe and possible avenues to enhance model predictions.

As mentioned before, the LSTM model maintains a consistent average final validation loss on the order of  $10^{-5}$  for spins two, four, and six. Interestingly, the transformer average final validation loss changes from being on the order of  $10^{-6}$  for two and four spins to being on the

order of  $10^{-2}$  for six spins. These results were expected as previous literature has shown that LSTMs can perform better than transformers for time-series problems with complex dynamics [25, 26]. One consideration we have is that the LSTM and Transformer models are relatively small frameworks. By increasing the size of both models, we could improve their robustness and observe better performance. To do so, we could increase the size of the latent space or number of hidden layers, compare the performance of the larger models against their smaller counterparts, and see if the Transformer remains inferior to the LSTM when simulating more complex systems. Though, due to later results, it is an open question for us whether that would improve the relative performance between the LSTM and Transformer at the breakdown point where performance flips.

Another compelling result for the strength of the ML models is the apparent lack of impact of context percentage and Frobenius norm on the average final validation loss. We show that we can feed our ML models anywhere between 10% and 90% of a single  $J^{-1}$  and produce average final validation losses on the same order of magnitude. Both ML models can also accurately predict the transverse Ising model with isotropic interaction strengths and varying degrees of anisotropic interaction strengths to similar accuracy. We acknowledge there is room to lower context percentages and higher Frobenius norms to determine the limits at which our models begin to perform worse. However, with these factors negligibly affecting the accuracy of our models, we show that our models are solely affected by scaling the amount of spins in the system being modeled.

We observe that the standard deviations of the final validation loss for both the LSTM and transformer models decrease as the spin number increases. Intuitively, one might expect the opposite since the wavefunction dynamics of higher-complexity spin systems should be more

challenging to model. However, this trend is likely due to the LSTM and Transformer models not producing consistently strong models as the spin number grows. At lower spin numbers, a greater number of models achieve high performance, leading to greater variability in final validation loss. This number of higher performance models relative to the average model decreases as the predictions become more complex.

The trends in the data we observe in the transverse Ising model system persist for higher complexity systems, like in figures [3.5](#), [3.9](#), etc., making this model a valuable starting point for conveying the success of the ML models we developed in predicting quantum systems.

## 4.2 Anisotropic Ising Next-Nearest Neighbor Model

Interestingly, introducing next-nearest neighbor interactions does not appear to impact the training or validation of the ML models. The same trends we observed for the Ising model without next-nearest neighbor interactions persist. We still observe no meaningful impact of changing the degree of anisotropy of the interaction strength of context percentage we input to the ML models. Given the lack of impact of varying the interaction strength, we also evaluate increasing and decreasing the total interaction strength to conclusively determine that our ML models can successfully predict the Ising model regardless of interaction strength, Figure [4.1](#).

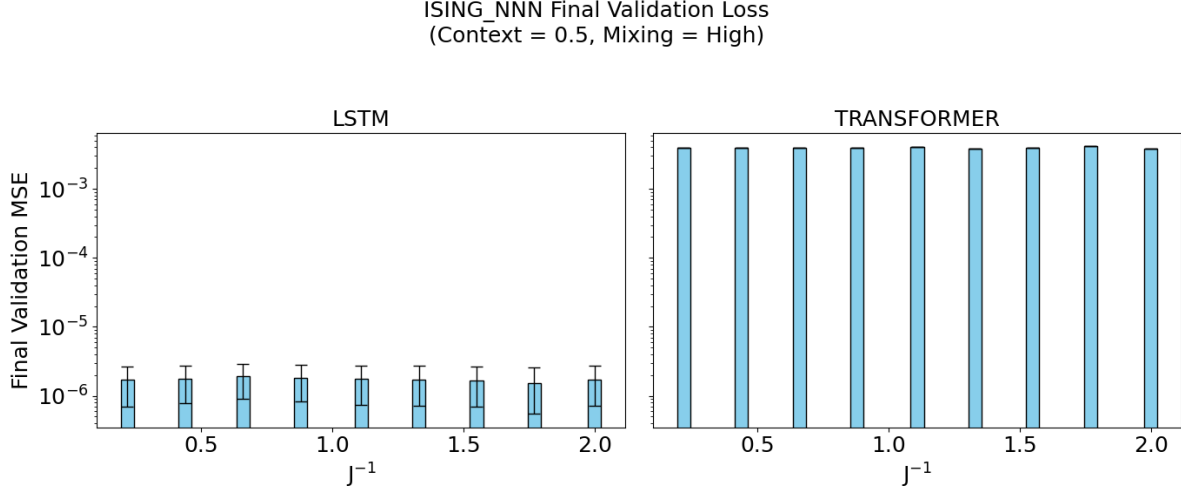


Figure 4.1: Final validation MSE for Ising NNN model with varying interaction strengths (JINV) with both ML models. Final validation MSE is not affected by increasing or decreasing JINV for the LSTM and transformer ML models for the next-nearest Ising model. Note that this figure serves as a small subset of runs for visualization purposes and the data trends apply uniformly across the other simulated physical models. Error bars are estimated through the standard deviations in Figure 3.5.

With both the LSTM and transformer models, we show that changing the interaction strength does not have any effect on either model’s final validation loss, even at the highest degree of anisotropy that we evaluated at. Aside from a few challenging predictions, the absolute error of both models evaluating with the same conditions in Figure 4.1 seems to cluster around lower values, Figure 4.2. This indicates that despite potential complexities introduced by varying interaction strengths, both models generalize well and maintain stable performance across different scenarios. To further evaluate another potential factor that could effect the prediction accuracy of our ML models, we evaluated whether our error accumulated over time, which was also found to be negligible, Figure 4.2.

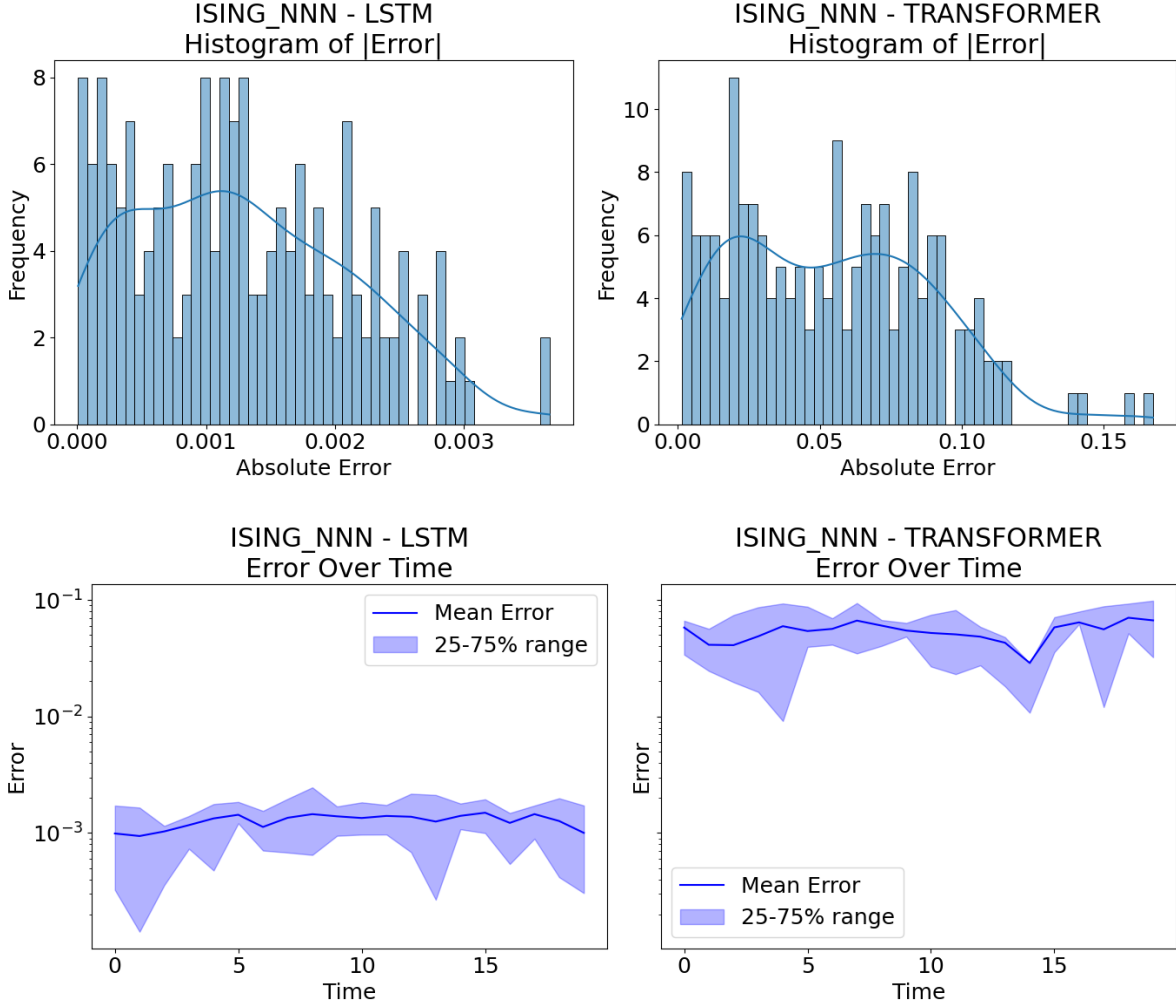


Figure 4.2: Top: histogram plots for JINV runs shown in Figure 4.1 demonstrate that the models have very low absolute error, aside from a few challenging predictions. Bottom: Error over time (steps) for the same JINV runs. The ML model predictions do not accumulate error over time. Note that this figure serves as a small subset of runs for visualization purposes and the data trends apply uniformly across the other simulated physical models.

### 4.3 Heisenberg Models

With the Heisenberg model, we expand our analysis to more physically relevant systems. Once again, we showcase the ML models' capability to accurately predict time evolution with errors on the order of  $10^{-5}$  and  $10^{-6}$ , using the LSTM and transformer models, respectively,

for  $2 \times 2$  and larger rectangular and triangular lattices [as in figures 3.9, 3.13]. Notably, we observe that the LSTM model starts to outperform the transformer model between spin sizes 4 and 6. While the results align in significance and trends with the previous two models, the key advancement demonstrated with the Heisenberg model is our ability to simulate systems of contemporary interest in quantum physics with high accuracy. Isotropic 2D lattice structures remain fundamental tools in modern physics for studying phase transitions, entanglement estimation, and long-range interactions in quantum systems [27–29]. Additionally, one of the most significant problems of interest is frustration in quantum spin systems. Frustration arises when competing interactions prevent the system from reaching a simple ordered ground state, leading to highly degenerate or exotic quantum phases, especially in triangular lattices [22]. Our ML models could enhance the ability to analyze these frustrated quantum spin systems by efficiently predicting their time evolution. Additionally, with our data generation framework, it would also be trivial to implement other lattice structures that are currently of interest or will be of interest in the future, such as Kagome or honeycomb lattices [22], which we believe would not impact the performance of our models.

#### 4.4 1D Hubbard Model

Here, we demonstrate our ability to expand into predicting magnetism-based quantum systems. Contrary to what we expected, our ML models predicted the Hubbard model with the highest degree of accuracy, having losses on the order of  $10^{-7}$ - $10^{-8}$  for 2 and 4 spins with both the LSTM and transformer. This is especially curious as the physical amount of information that must be predicted in the 4-spin case is actually much larger than the ‘flip’ point we observed in

the anisotropic Ising model and Heisenberg systems.  $2^6$  spins is a total of 64 values to predict per transient timestep, but the Hubbard models must predict 256 values per transient timestep. Additionally, the average final validation loss and training MSE did not follow the same trends as the Ising or Heisenberg models. For both the ML models, the training MSEs for spins 2 and 4 did not follow the same trajectories or reach the final MSE on the same order of magnitude. Also, for both ML models, the training MSE for 2 spins was higher than for 4 spins. The final validation loss follows the same trend, where the average final validation losses for 2 spins for the LSTM and transformer models are higher than the average for 4 spins, respectively. The transition for the LSTM model outperforming the transformer model also occurs with lower spin systems, occurring between spins 2 and 4 as opposed to 4 and 6 with the other models. This is extremely interesting as it suggests some form of *anti*-scaling as spin count increases - models perform *better* in more complex systems.

Unfortunately, the data generation for the Hubbard model requires amounts of RAM that are currently not feasible to handle with our available machines. The computational demands of simulating large Hubbard model systems often exceed the memory capacity of standard hardware, limiting our ability to perform large-scale simulations. A 6 spin model would be forced to predict 4096 values *per* transient timestep, limiting our training batch size as well based on available computational VRAM. However, with extensive computational resources, we could feasibly and accurately predict Hubbard model systems with more spins, allowing us to study this aforementioned suggested anti-scaling. Existence of systems that become easier to predict in the limit of high spins might enable us to study extremely large quantum systems to a high degree of computational accuracy that is not achievable in experiment. However, we would require either significant speedups in the simulation methods for the Hubbard model, or a theoretical back-

ground on which we could stitch 1D Hubbard results into a much larger chain than simulated.

Additionally, we could study even more complex systems, such as the Tavis–Cummings model. This model is particularly significant because it describes the interaction between a quantum spin system and a single-mode electromagnetic field, making it a valuable tool for understanding quantum optics, cavity QED, and quantum information processing [30,31].

## 4.5 Summary

Across the transverse Ising, Ising with next-nearest neighbor interaction, and Heisenberg models, we observe an interesting phenomenon where the LSTM model performs better than the transformer model between four and six spins. We *speculate* that a reason for this is that the way a transformer encodes time dependence is with a predefined function known as a positional encoder. We used the positional encoding presented in the original paper, which was initially developed to perform for language models. We believe that this positional encoder may not be optimized for non-dissipative quantum systems. LSTMs, on the other hand, "learn" time dependence, so with larger and more complex systems, the LSTM might learn the form of time dependence in quantum systems. To further evaluate this phenomenon, we could develop different positional encoding functions explicitly optimized to generate the individual systems with which we work to generate a more robust transformer model for higher-spin systems. At the very least, we could evaluate the impact of using different positional encoding on long-term prediction accuracy. The same reasoning can be used to support the development of an alternate positional encoder optimized for the Hubbard model to potentially augment the prediction capabilities of the transformer. More generally, there is space in which to work to evaluate the difference between

LSTM and transformer learning when time dependence is not well structured, specifically for quantum systems. This highlights potential further research that has implications in both general ML studies - understanding situations in which LSTMs might outperform transformers - as well as in studying quantum systems.

Despite there being further questions we wish to explore, we have demonstrated a significant speedup in accurate data generation with our models. For example, using the transformer model, with the current settings of a 64-dimensional latent space, four encoder heads, and two hidden layers, we can generate 20,000 transients of 6 spin data with 10% context (20-step input to generate 180 steps) in 0.21 seconds. Our data generation using the same machine to generate 20,000 transients takes 5.2 minutes on average. This demonstrates a significant decrease in time and resources required to generate full simulations of quantum systems. We can even quickly generate results on systems that our current setup on Zaratan cannot, such as results for 10 spin or more systems. Figure 4.3 demonstrates that even with higher numbers of spins, the inference time of the models remains under a second to generate 20,000 transients. We believe that this may dramatically enhance access to simulations by generating data for larger numbers of spins in fast amounts of time that are otherwise not feasible with smaller machines. We believe there is also potential in using our system to generate evolutions of large spin chains in bulk, which can be bootstrapped to even larger spin chains (see Lieb-Robinson bounds discussion). From this, it would be possible to generate an extremely large amount of data for spin systems that cannot be simulated with our current resources - potentially allowing us to bootstrap model training to even higher spin counts. We believe this process could be continued to generate data for even larger spin systems, limited primarily by available VRAM and the necessity for some amount of 'context' data to be inserted into the ML model.

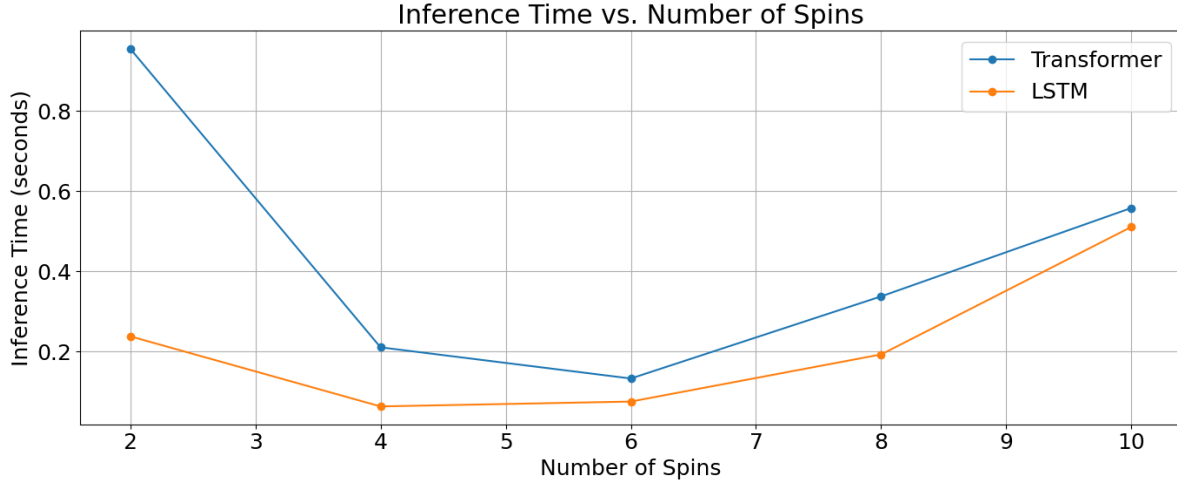


Figure 4.3: Inference time: the ML models take to generate 20,000 transients of the next-nearest neighbor Ising model takes under a second for up to 10 spins. Execution time will vary by machine, but this gives a feeling of the possible speed with commercially available hardware. This was evaluated on an Apple Mac M4 Max with PyTorch MPS (Metal Performance Shader) acceleration.

Another consideration is that our models do not account for dissipation and drive, meaning that our simulations only consider the quantum behavior of the models. We believe that this makes our results more convincing because, with dissipation, the data would approach a steady state over time. This steady state would no longer have any dynamics, making the prediction easily done. Additionally, the presence of a quantum drive - external forces or fields that manipulate or control a quantum system - would also make the dynamics easier to predict as they could dominate the progression of system dynamics. However, one interesting direction we could take is to utilize an external force that flips the state of a qubit or drives different transitions in energy levels at a certain time and determine whether our model can predict the behavior of the system before and after the external factor is applied. External factors such as magnetic fields are commonly used in many physical systems, so our work could augment the prediction time of those altered systems.

Our work could be extended to simulate realistic spin systems. Let  $n \gg 1 \in \mathbb{N}$ ,  $m \ll n \in \mathbb{N}$ ,  $g = \lceil \frac{n}{m} \rceil$ . A lattice of  $n \times n \times n$  spins could be divided into  $g$  non-overlapping groups of  $m \times m \times m$  spins. Then, these  $g$  groups of spins could be simulated in parallel using  $g$  ML models for short periods of time. Taking advantage of the Lieb-Robinson bound, these  $g$  simulations could be stitched together in order to effectively simulate the entire large system, in following with the earlier mention of Lieb-Robinson bounds. This approach offers a promising avenue for simulating large-scale spin systems more efficiently and could pave the way for future advancements in quantum simulation.

## 4.6 Broader Societal Impacts

This project can potentially significantly improve access to advanced quantum simulation tools. Traditional large-scale quantum simulations often require expensive super-computing resources and specialized hardware. Our methods offer a building block toward more accessible simulation platforms. As our system uniquely predicts the full quantum state, it also allows quantum theory groups to simulate time-evolution of any number of desired observables quickly. This increased accessibility is valuable in emerging fields where measurement protocols for physical observables have not yet been developed, and for theoretical situations where the state includes too many interactions to be tractable in a small amount of time.

Quantum systems are inherently complex due to superposition, entanglement, and non-local interactions, making simulations computationally intensive. Despite these challenges, accurate simulations of quantum systems are essential for advancing numerous applications. For example, for quantum biochemistry, simulating quantum dynamics can help rule out different

degenerate protein folding states. A complex time-varying energy landscape governs protein folding, and small errors in simulation can lead to vastly different predicted structures due to the significant number of nearly-degenerate states [32]. Recent advancements in resource-efficient quantum algorithms have demonstrated their ability to model protein folding processes with greater accuracy, reducing the computational cost associated with classical methods [32]. By utilizing efficient machine learning models, such as those used in our work to simulate quantum dynamics, we could more accurately predict the stable configurations of proteins. Discussing broad impacts, improved predictive capability may accelerate drug discovery for personalized medication programs.

Quantum-inspired optimization and simulation techniques hold promise for advancing a range of complex applications, particularly in medicine and environmental monitoring. One specific area for advancement is in utilizing quantum simulations to simulate complex outcomes related to medical treatment plans, as in the case with radiation therapy for cancer [33]. Accurate predictive models could potentially assist in developing algorithms that can simulate better treatment plans than the current classical systems [33].

Additionally, precise quantum simulations could enhance environmental monitoring and medical imaging. In weather prediction, quantum sensors developed through advanced simulation techniques could offer improved sensitivity and selectivity, enabling more accurate climate monitoring. Similarly, enhanced quantum sensors can be incorporated into advanced magnetic resonance imaging (MRI) systems, where refined quantum simulations can further optimize sensor design. This could lead to higher-resolution images, enabling earlier detection and more accurate diagnostics of various diseases.

While quantum-inspired techniques are promising, their integration into practical applica-

tions requires significant further research and validation. Nevertheless, ongoing advancements in quantum optimization and simulation continue to highlight the potential for transformative improvements across diverse fields.

Although our work offers substantial benefits, it is important to acknowledge some limitations. Our ML models scale well in timing with respect to possible data generation, but they do require generated data to learn. While physics-informed neural network approaches are interesting to pursue in the future, the broader networks we investigated need many simulated transients to predict realistic results. Furthermore, with unconstrained outputs, there is no guarantee that the provided wavefunction is physical. This means the predicted wave function may not actually realistically represent a probability distribution. This would raise small problems in computationally determining expectation values of unknown observables, but it can be addressed. Moreover, the uncertainties of these models remain a challenging problem. Therefore, it is critical to continue the research to ensure robust predictions in practical applications.

Overall, by reducing the resource barrier and offering a more accessible pathway for simulating larger-scale quantum systems, our work contributes a critical building block to the quantum simulation field. This supports the development of transformative technologies, from advanced sensors and more accurate weather forecasts to improved healthcare diagnostics, underscoring the broad societal implications of our research.

## 4.7 Future Steps

To expand upon this research, one good course of action would be to expand upon the architectures that are tested with different Hamiltonians. One architecture that we looked at was

an Equavariant transformer, which is more optimized for symmetrical data and could help with computational efficiency [34]. Another architecture that has interesting potential is the Fourier neural operator. This architecture uses fourier transforms to reduce the complexity of the data they're training on and neural operators that only need to be trained once, which can reduce computation costs. This allows for a computationally efficient model that works well for partial differentiation problems [35]. Another step to improve our efficiency is to look at how we can make our data symmetrical to help our models train. Overall, by expanding the scope of models we train on these Hamiltonians and others in the future, the more we can understand this space and how to model it.

## Appendix A: Previous Experiments

### A.1 Overview

In the earlier phases of our research, we explored another approach to predict the time evolution of full wavefunctions under the Transverse Ising model. This section briefly details our experience with the Echo State Network model (ESN) and provides insight into why we abandoned this approach in favor of focusing on the LSTM and Transformer models.

### A.2 Echo State Network

The ESN, like the LSTM model, is a type of recurrent neural network designed to process sequential data; distinct from the LSTM, the ESN is a type of reservoir computing framework [36]. The ESN forces input through a hidden layer of sparsely connected neurons (the "reservoir") [36]. The weights of the reservoir are randomly initialized and remain static; training an ESN involves only adjusting the weights of the output of the reservoir.

Intuitively, we expect that the reservoir's response depends on the input sequence over time, with the effects of initial conditions gradually disappearing. We employed the ESN in hopes of demonstrating its potential for accurate predictions of wavefunction dynamics with reduced computational time and cost compared to fully trainable models like the LSTM and Transformer.

The ESN was not able to capture long-term time dependencies in the problem. We believe that this may be caused by the ESN relying heavily on the randomly initialized properties of the reservoir.

The Transverse Ising (Sec. 1.3.2) model Hamiltonian includes interactions between neighboring spins and an external magnetic field. A wavefunction evolving under this Hamiltonian depends heavily on the effects of prior states. Compared to other models, the ESN struggles to retain information from earlier time steps over long periods.

We performed a sweep of two major hyperparameters of the ESN: spectral radius and the reservoir’s sparsity. Spectral radius refers to the maximum absolute eigenvalue of the reservoir’s weight matrix [36]. Sparsity refers to the proportion of zeros in this matrix, indicating the amount of connections between neurons in the reservoir [36]. Despite extensive tuning, all combinations of these hyperparameters resulted in losses too high for accurate use on the Transverse Ising model, even on our initial pure Ising model simple problem. The sweep in figure A.1 was done on data generated using the pure Ising model - which is isotropic and includes a magnetic field term - with a low spin count. However, it was still unable to capture the time evolution of our total quantum state.

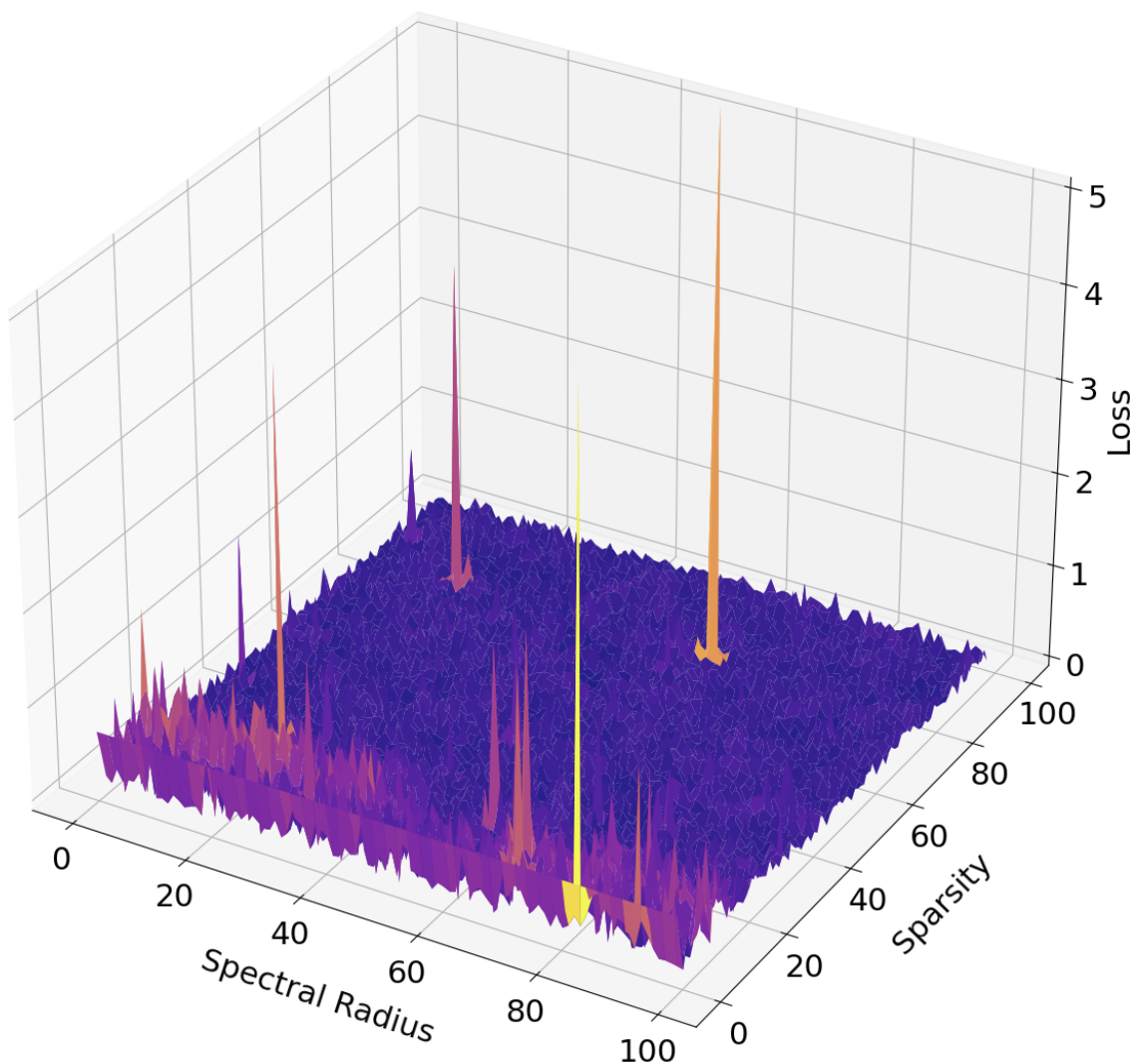


Figure A.1: ESN loss as a function of Spectral Radius and Sparsity hyperparameters

We see that the loss spikes for a small number of combinations of spectral radius and sparsity. Optimal combinations however, yielded mean squared errors on the order of magnitude of  $10^{-2}$ . Evidently, the ESN underperformed the LSTM and Transformer under the Transverse Ising model. Since this architecture struggled in comparison to others for a relatively simple model, we did not pursue it in our studies of models of increased complexity.

## Appendix B: QuTiP and Lindblad Master Equation

All information in this appendix about the Lindblad master equation is from [37].

Recall earlier we stated that quantum systems may live in a superposition of possible states. These possible states are known as 'pure' states, and thus, any system in a superposition of states is in a mixed state. This mixed state can be represented with a matrix known as a 'density' matrix  $\rho$ .

$$\rho = \sum_i p_i |\psi_i\rangle \langle \psi_i| \quad (\text{B.1})$$

where  $p_i$  represents the probability of the  $i$ th state, or  $c_i^* c_i$  based on our definition of the coefficients in the wavefunction's vector.

Time evolution of this density matrix follows this expression

$$\frac{d\rho}{dt} = -i[H, \rho] + \sum_{i,j=1}^{d^2-1} g_{i,j} (F_i \rho F_j^\dagger - \frac{1}{2} \{F_j^\dagger F_i, \rho\}) \quad (\text{B.2})$$

which is known as the Lindblad master equation. This generalized equation allows for the time-evolution of general quantum systems, and is employed by QuTiP's mesolver (master equation solver) that we used in this project.  $H$  represents the system Hamiltonian,  $[\cdot, \cdot]$ ,  $\{\cdot, \cdot\}$  represent the commutator and anti-commutator operations as usual, and  $d$  represents the dimensionality of

the Hilbert space for  $H$ . The summation represents all 'jump' operations - or all operations that are dissipative in some fashion - constructed to include the whole Hilbert space basis. Therefore, the  $F$  operators are said aforementioned jump operations, representing operators that as a whole cover an entire basis of  $d \times d$  operators. The Kronecker product of all the Pauli spin operators is an excellent example of such a set of  $F$ .  $g$  then represents the mixing between different elements of  $F$  to set the strength of which operations are more or less dissipative.

## Appendix C: Sample Visualizations

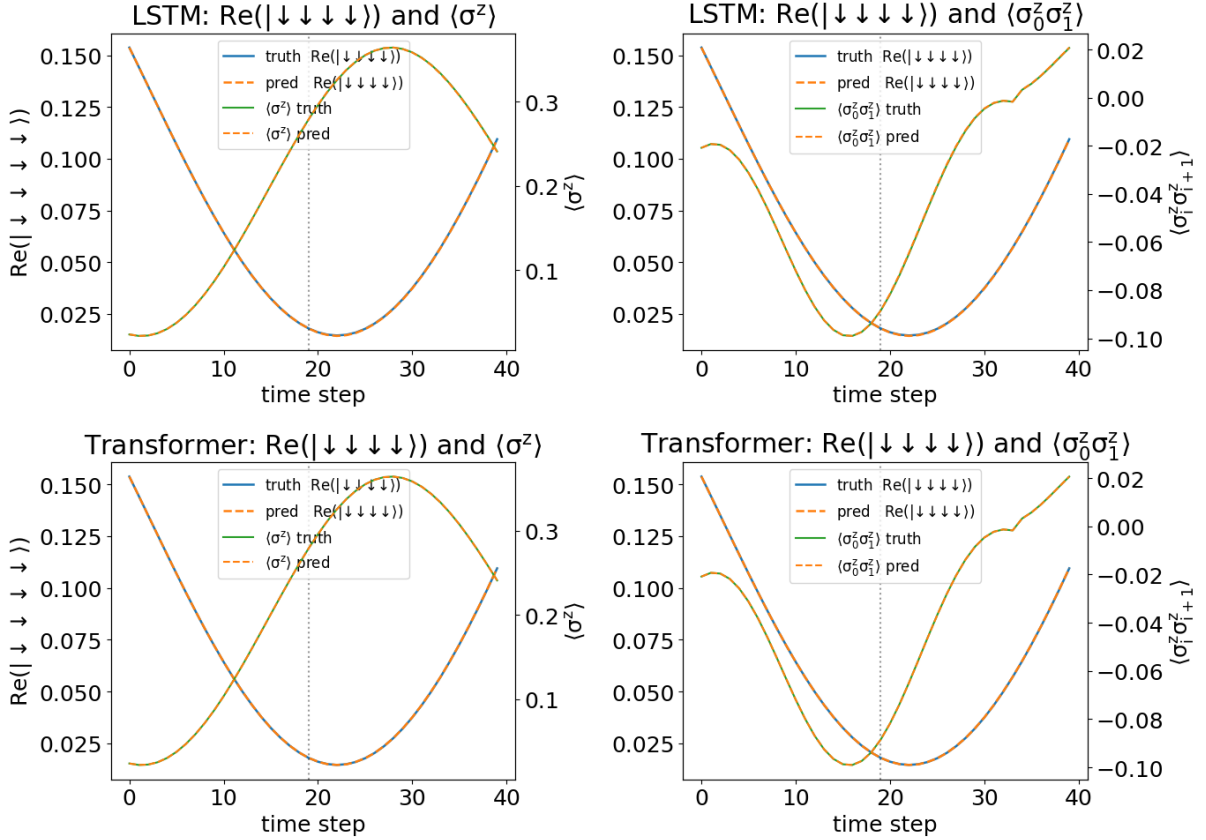


Figure C.1: We train LSTM and Transformer models on the 4 spin Ising model explained earlier in the thesis. We present the real part of the first component of the wave function, and our model’s predictions. The predictions here are exact up until the point marked by the gray line; this separates the context provided to the model and the part that is actually predicted. From there, we observe near pixel-perfect predictions from our model through this first batch. This small-scale example, on a small chunk of our data - first 40 timesteps - generalizes well to other spin site results, other components (imaginary components for example), and models, at a similar  $10^{-6}$  MSE level. We additionally show the value of and prediction of 2 different operators to demonstrate how our model can be used to predict the evolution of theoretical and real observables over time.

To demonstrate briefly the performance of our model by looking at real outputs on some sample validation data, we’ve provided Figure C.1. In Figure C.1, we train our LSTM and Transformer models on 4 spins of the simple Ising model presented in Figures 3.1, 3.3, 3.2, etc. We observe near pixel perfect matching between our predicted results and the actual results after the context divider. We also plot the true calculated and predicted values for 2 observables, and demonstrate no major deviation in observable value and predicted value on validation data. While the shown results are only for the real component of the very first spin in, in this case, the Ising model, our results demonstrate how the  $10^{-6}$  order of magnitude MSE we obtained with the LSTM and Transformer behaves. We show, in this case, the results only for 1 batch up to time step 40, but the patterns shown generalize to other models with similar orders of magnitude loss values.

## Bibliography

- [1] Daniel Loss and David P. DiVincenzo. Quantum computation with quantum dots. *Physical Review A*, 57(1):120–126, January 1998. Publisher: American Physical Society.
- [2] Eunmi Chae, Joonhee Choi, and Junki Kim. An elementary review on basic principles and developments of qubits for quantum computing. *Nano Convergence*, 11:11, March 2024.
- [3] Manuel Erhard, Mario Krenn, and Anton Zeilinger. Advances in high-dimensional quantum entanglement. *Nature Reviews Physics*, 2(7):365–381, July 2020. Publisher: Nature Publishing Group.
- [4] Luca Chirolli and Guido Burkard. Decoherence in Solid State Qubits. *Advances in Physics*, 57(3):225–285, May 2008. arXiv:0809.4716 [cond-mat].
- [5] Masoud Mohseni, Artur Scherer, K. Grace Johnson, Oded Wertheim, Matthew Otten, Navid Anjum Aadit, Yuri Alexeev, Kirk M. Bresniker, Kerem Y. Camsari, Barbara Chapman, Soumitra Chatterjee, Gebremedhin A. Dagnew, Aniello Esposito, Farah Fahim, Marco Fiorentino, Archit Gajjar, Abdullah Khalid, Xiangzhou Kong, Bohdan Kulchytsky, Elica Kyoseva, Ruoyu Li, P. Aaron Lott, Igor L. Markov, Robert F. McDermott, Giacomo Pedretti, Pooja Rao, Eleanor Rieffel, Allyson Silva, John Sorebo, Panagiotis Spentzouris, Ziv Steiner, Boyan Torosov, Davide Venturelli, Robert J. Visser, Zak Webb, Xin Zhan, Yonatan Cohen, Pooya Ronagh, Alan Ho, Raymond G. Beausoleil, and John M. Martinis. How to Build a Quantum Supercomputer: Scaling from Hundreds to Millions of Qubits, January 2025. arXiv:2411.10406 [quant-ph].
- [6] E. Knill. Quantum computing with realistically noisy devices. *Nature*, 434(7029):39–44, March 2005. Publisher: Nature Publishing Group.
- [7] Marco Fellous-Asiani, Jing Hao Chai, Yvain Thonnart, Hui Khoon Ng, Robert S. Whitney, and Alexia Auffèves. Optimizing Resource Efficiencies for Scalable Full-Stack Quantum Computers. *PRX Quantum*, 4(4):040319, October 2023. Publisher: American Physical Society.
- [8] Haoran Liao, Derek S. Wang, Iskandar Sitdikov, Ciro Salcedo, Alireza Seif, and Zlatko K. Minev. Machine learning for practical quantum error mitigation. *Nature Machine Intelligence*, November 2024.
- [9] Ian Convy, Haoran Liao, Song Zhang, Sahil Patel, William P. Livingston, Ho Nam Nguyen, Irfan Siddiqi, and K. Birgitta Whaley. Machine Learning for Continuous Quantum Error

- Correction on Superconducting Qubits. *New Journal of Physics*, 24(6):063019, June 2022. arXiv:2110.10378 [quant-ph].
- [10] Wenjian Hu, Rajiv R. P. Singh, and Richard T. Scalettar. Discovering phases, phase transitions, and crossovers through unsupervised machine learning: A critical examination. *Physical Review E*, 95(6):062122, June 2017. Publisher: American Physical Society.
  - [11] Guido Cossu, Luigi Del Debbio, Tommaso Giani, Ava Khamseh, and Michael Wilson. Machine learning determination of dynamical parameters: The Ising model case. *Physical Review B*, 100(6):064304, August 2019. arXiv:1810.11503 [cond-mat, physics:hep-lat, physics:physics].
  - [12] Francesco D’Angelo and Lucas Böttcher. Learning the Ising Model with Generative Neural Networks, January 2020.
  - [13] Kunni Lin, Jiawei Peng, Chao Xu, Feng Long Gu, and Zhenggang Lan. Automatic Evolution of Machine-Learning-Based Quantum Dynamics with Uncertainty Analysis. *Journal of Chemical Theory and Computation*, 18(10):5837–5855, October 2022.
  - [14] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention Is All You Need, August 2023. arXiv:1706.03762 [cs].
  - [15] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale, June 2021. arXiv:2010.11929 [cs].
  - [16] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, Alex Bridgland, Clemens Meyer, Simon A. A. Kohl, Andrew J. Ballard, Andrew Cowie, Bernardino Romera-Paredes, Stanislav Nikolov, Rishub Jain, Jonas Adler, Trevor Back, Stig Petersen, David Reiman, Ellen Clancy, Michal Zielinski, Martin Steinegger, Michalina Pacholska, Tamas Berghammer, Sebastian Bodenstein, David Silver, Oriol Vinyals, Andrew W. Senior, Koray Kavukcuoglu, Pushmeet Kohli, and Demis Hassabis. Highly accurate protein structure prediction with AlphaFold. *Nature*, 596(7873):583–589, August 2021. Publisher: Nature Publishing Group.
  - [17] Yuan-Hang Zhang and Massimiliano Di Ventura. Transformer Quantum State: A Multi-Purpose Model for Quantum Many-Body Problems. *Physical Review B*, 107(7):075147, February 2023. arXiv:2208.01758 [quant-ph].
  - [18] El Amine Cherrat, Iordanis Kerenidis, Natansh Mathur, Jonas Landman, Martin Strahm, and Yun Yvonna Li. Quantum Vision Transformers. *Quantum*, 8:1265, February 2024. arXiv:2209.08167 [quant-ph].
  - [19] Leo Feng, Frederick Tung, Mohamed Osama Ahmed, Yoshua Bengio, and Hossein Hajimirsadegh. Were RNNs All We Needed?, October 2024. arXiv:2410.01201 [cs] version: 1.

- [20] Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, November 1997.
- [21] Ying He-ping and Ji Da-ren. A Coherent Updating Algorithm for the XYZ Model. *Chinese Physics Letters*, 13(3):219, March 1996.
- [22] J. Khatua, B. Sana, A. Zorko, M. Gomilšek, K. Sethupathi, M. S. Ramachandra Rao, M. Baenitz, B. Schmidt, and P. Khuntia. Experimental signatures of quantum and topological states in frustrated magnetism. *Physics Reports*, 1041:1–60, November 2023.
- [23] Alexander Altland and Ben Simons. *Condensed Matter Field Theory*. Cambridge University Press, Cambridge, 3 edition, August 2023.
- [24] Antoine Georges, Gabriel Kotliar, Werner Krauth, and Marcelo J. Rozenberg. Dynamical mean-field theory of strongly correlated fermion systems and the limit of infinite dimensions. *Reviews of Modern Physics*, 68(1):13–125, January 1996. Publisher: American Physical Society.
- [25] Anna Pölz, Alfred Paul Blaschke, Jürgen Komma, Andreas H. Farnleitner, and Julia Derx. Transformer Versus LSTM: A Comparison of Deep Learning Models for Karst Spring Discharge Forecasting. *Water Resources Research*, 60(4):e2022WR032602, 2024. eprint: <https://onlinelibrary.wiley.com/doi/pdf/10.1029/2022WR032602>.
- [26] Albert Zeyer, Parnia Bahar, Kazuki Irie, Ralf Schlüter, and Hermann Ney. A Comparison of Transformer and LSTM Encoder Decoder Models for ASR. In *2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 8–15, December 2019.
- [27] Hisato Komatsu, Yoshihiko Nonomura, and Masamichi Nishino. Phase diagram of the two-dimensional dipolar Heisenberg model with Dzyaloshinskii-Moriya interaction and Ising anisotropy. *Physical Review B*, 103(21):214404, June 2021. Publisher: American Physical Society.
- [28] Chandrima B. Pushpan, Harikrishnan K. J, Prithvi Narayan, and Amit Kumar Pal. Estimating entanglement in 2D Heisenberg model in the strong rung-coupling limit. *Physical Review A*, 110(3):032408, September 2024. arXiv:2301.04615 [quant-ph].
- [29] Jiarui Zhao, Menghan Song, Yang Qi, Junchen Rong, and Zi Yang Meng. Finite-temperature critical behaviors in 2D long-range quantum Heisenberg model. *npj Quantum Materials*, 8(1):1–6, October 2023. Publisher: Nature Publishing Group.
- [30] Martin Blaha, Aisling Johnson, Arno Rauschenbeutel, and Jürgen Volz. Beyond the Tavis-Cummings model: Revisiting cavity QED with ensembles of quantum emitters. *Physical Review A*, 105(1):013719, January 2022. Publisher: American Physical Society.
- [31] S. Agarwal, S. M. Hashemi Rafsanjani, and J. H. Eberly. Tavis-Cummings model beyond the rotating wave approximation: Quasi-degenerate qubits. *Physical Review A*, 85(4):043815, April 2012. arXiv:1201.2928 [quant-ph].

- [32] Anton Robert, Panagiotis KI Barkoutsos, Stefan Woerner, and Ivano Tavernelli. Resource-efficient quantum algorithm for protein folding. *npj Quantum Information*, 7(1):1–5, February 2021. Publisher: Nature Publishing Group.
- [33] Julia M. Pakela, Huan-Hsin Tseng, Martha M. Matuszak, Randall K. Ten Haken, Daniel L. McShan, and Issam El Naqa. Quantum-inspired algorithm for radiotherapy planning optimization. *Medical Physics*, 47(1):5–18, 2020. \_eprint: <https://onlinelibrary.wiley.com/doi/pdf/10.1002/mp.13840>.
- [34] Akio Tomiya and Yuki Nagai. Equivariant Transformer is all you need. October 2023. arXiv:2310.13222 [hep-lat].
- [35] Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier Neural Operator for Parametric Partial Differential Equations, May 2021. arXiv:2010.08895 [cs].
- [36] Herbert Jaeger. The “echo state” approach to analysing and training recurrent neural networks – with an Erratum note. page 47, January 2010.
- [37] Daniel Manzano. A short introduction to the Lindblad Master Equation. *AIP Advances*, 10(2):025106, February 2020. arXiv:1906.04478 [quant-ph].