Effect of Sample Difficulty on the Time-to-First-Spike of Spiking Neural Networks

Eren Aydoslu
TU Delft

Abstract

Spiking neural networks (SNNs) with Time-to-First-Spike (TTFS) coding promise rapid, sparse, and energy-efficient inference. However, the impact of sample difficulty on TTFS dynamics remains underexplored. We investigate (i) how input hardness influences first-spike timing and (ii) whether training on hard samples expedites inference. By quantifying difficulty via geometric margins and Gaussian-noise perturbations, and modeling leaky integrate-and-fire dynamics as Gaussian random walks, we derive first-hitting-time predictions. We further show that training-time noise, akin to ridge regularization, reduces weight variance and increases expected spike latencies. Empirical results on a synthetic task, MNIST, NMNIST, and CIFAR-10 with spiking MLPs/CNNs confirm that harder inputs slow inference and noise-trained models trade robustness for latency. Our findings align TTFS behavior with drift-diffusion models and provide a framework for balancing speed and robustness in neuromorphic SNNs.

1. Introduction

Spiking neural networks (SNNs), inspired by biological neural systems, encode information using spike timings rather than continuous activation values. Among various neural coding schemes, Time-to-First-Spike (TTFS) coding has garnered attention for its efficiency and biological plausibility. TTFS-based SNNs inherently prioritize rapid, sparse, and energy-efficient computation, making them particularly appealing for low-power and real-time applications [10].

However, the performance and characteristics of TTFS-based SNNs under conditions of varying sample difficulty remain under-explored. Specifically, understanding how the time-to-first-spike behavior of latency-encoded SNNs changes as samples become intrinsically more challenging is crucial for both theoretical insights and practical applications. In classical machine learning, sample difficulty is often associated with ambiguity in the data distribution and proximity to decision boundaries [8, 41]. Inspired by this, our research investigates the following central questions:

1. What happens to the TTFS behavior of latency-encoded SNNs as samples become increasingly difficult?
2. Can we use harder samples during training to make inference faster?

We approach this question by considering sample difficulty through a proxy measure, additive Gaussian noise. Drawing parallels from statistical learning theory and ridge regression equivalence in linear models, we hypothesize that introducing Gaussian noise to training samples reduces the variance of the learned weights. Under this assumption, we model neuronal membrane potentials as Gaussian random walks, enabling analysis through the lens of first-hitting-time stochastic processes. We believe that reduced weight variance, induced by noisy training, leads to higher expected first-spike times, thus potentially decelerating inference.

To validate our hypothesis, we conducted extensive experiments utilizing spiking multi-layer perceptrons (MLPs) and convolutional neural networks (CNNs) using a synthetic dataset and across widely-used datasets, MNIST, NMNIST, and CIFAR-10, examining network responses under various noise strength. Through empirical analyses, we measure and explain how first-spike latency evolves relative to sample difficulty, thereby advancing the understanding of latency coding dynamics in spiking neural networks.

In summary, we make the following contributions:

1. We introduce a principled analytical framework that models TTFS dynamics under varying sample difficulty by treating membrane-potential accumulation as a Gaussian random walk with decay/mean-reversion and applying first-hitting-time theory.
2. We establish a theoretical link between training-time Gaussian noise (akin to parameter regularization) and increased first-spike latencies via reduced synaptic-weight variance.
3. We empirically validate our theory on both a synthetic SepDots task and three standard benchmarks (MNIST,

NMNIST, CIFAR-10) using spiking MLPs and CNNs, showing that harder inputs systematically slow inference and that noise training trades robustness for latency.
4. We demonstrate that TTFS latency under uncertainty aligns with classical drift-diffusion models.

2. Related Work

2.1. Sample Difficulty Metrics

In related literature, many different principal approaches have been proposed to quantify the intrinsic "difficulty" of individual samples. However, for the scope of this research, we are only interested in two: (i) geometric margin measures relative to a decision boundary, and (ii) synthetic hardness induced by Gaussian-noise perturbations. These metrics have been successfully applied in supervised learning, active learning and robustness studies, but have not yet been systematically linked to latency behavior in TTFS-coded SNNs.

2.1.1. Margin-Based Difficulty

A classical measure of sample difficulty is the signed distance (margin) from the sample to the optimal decision boundary. In support vector machines, maximizing the minimum margin yields better generalization, and samples with small margins are those most susceptible to misclassification under slight model perturbations [9, 43]. Varshney et al. [45] showed that margin directly captures both geometric resilience and Bayes error: points near the boundary lie in regions where class-conditional distributions overlap, incurring high irreducible error. More recent work has extended margin concepts to deep networks, demonstrating that margin distributions correlate with per-sample uncertainty and generalization gaps [1, 26]

2.1.2. Gaussian Noise as a Proxy for Difficulty

When the true decision boundary is unknown or intractable, hardness can be induced by adding isotropic Gaussian noise to inputs. The manifold hypothesis posits that high-dimensional data (e.g., images) lie near much lower-dimensional manifolds, each encoding semantic factors such as object identity or pose [12]. In modern vision embedding spaces, whether derived from contrastive models like CLIP or from GAN latent representations, individual semantic concepts (e.g., "cat"-ness, orientation, age) align with specific low-dimensional directions on these manifolds [22, 33]. Because isotropic Gaussian noise perturbs all coordinates equally at random, it is very unlikely to produce a change that aligns with a particular semantic direction (with probability near zero in high dimensions) and thus very unlikely to increase a sample's semantic class alignment or "margin" relative to its true manifold [2, 19]. In other words, to give an example, adding noise to an image of a cat is not very likely to make the image more cat-like. Instead, noise almost invariably pushes samples off their native manifolds toward regions of higher class overlap, increasing conditional uncertainty and Bayes error [13, 20]. Empirical studies confirm that moderate Gaussian perturbations can improve generalization by discouraging overfitting, whereas large noise amplitudes reliably degrade accuracy by elevating sample hardness [6, 21, 31].

2.2. Sample Difficulty and SNNs

Drift Diffusion Models (DDMs) describe the decision process as the accumulation of noisy evidence to one of two decision thresholds (see Figure 1), successfully capturing response-time distributions and choice accuracies across tasks and species [34, 35]. In their simplest form, DDMs introduce a one-dimensional decision variable $X(t)$ that evolves according to

$$dX = v\,dt + \sigma\,dW(t), \tag{1}$$

where v is the drift rate (mean evidence per unit time), σ the diffusion coefficient (noise magnitude), and $W(t)$ a standard Wiener process; choice and response time correspond to the first-passage time of $X(t)$ to one of two absorbing boundaries at $\pm a$ [35]. This framework naturally accounts for both the mean and variability of reaction times as well as speed–accuracy trade-offs: higher v yields faster, more consistent responses, while lower v produces slower, more variable decisions with increased error rates [3].

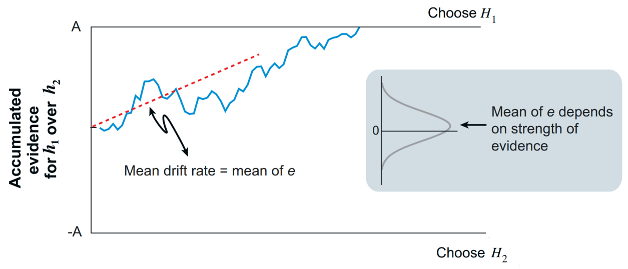

A
Choose H_1
Accumulated evidence for h_1 over h_2
0
Mean of e depends on strength of evidence
Mean drift rate = mean of e
-A
Choose H_2

Figure 1. Schematic of the drift diffusion model. The decision variable accumulates noisy momentary evidence $e(t)$ over time with mean drift rate μ (red dashed line), until it reaches the upper boundary $+A$ (choose H_1) or lower boundary $-A$ (choose H_2). The side-panel plot illustrates the probability density of momentary evidence e, whose mean shifts according to stimulus strength. Emphasizing how strong evidence leads to faster decisions. Figure taken from [18].

Although DDMs provide an elegant account of behavior, their abstract variables lack a clear mapping onto biophysically realistic neurons, motivating neural-level implementations that can measure diffusion dynamics in spiking activity. Analyses have directly linked neuron firing rates to accumulator processes in perceptual decisions, motivating neural-level diffusion analogues of DDM [18, 27, 29, 32].

In computational neuroscience, spiking neural network implementations have long been proposed as neural substrates for evidence accumulation, with early work demonstrating that interconnected pools of excitatory and inhibitory neurons can instantiate drift-like dynamics mapped onto diffusion model parameters [3, 47]. For instance, the biophysically detailed model by Wang (2002) [46] and its extension by Wong and Wang (2006) [47] showed that recurrent spiking circuits could replicate behavioral data by varying input “drift rate” and synaptic parameters corresponding to decision thresholds. Subsequent analyses mapped manipulations of spiking circuit parameters, such as input sensitivity, background excitation, and recurrent connectivity, to drift rate, boundary separation, and non-decision time in the diffusion model, elucidating concrete neural implementations of cognitive-level variables [44]. Additionally, spiking decision-making models with learning rules have been proposed to bridge cognitive models and SNNs, showing that accumulation can emerge from trainable spiking networks [23, 28].

However, these efforts have primarily focused on biological plausibility and parameter-mapping, often overlooking task-performance and latency effects under varying sample difficulties in deep learning oriented TTFS SNN architectures [3, 28, 47]. To date, the impact of sample-specific difficulty metrics, such as margin distance or noise-induced hardness, on first-spike latency within a drift-diffusion framework in TTFS SNNs has not been systematically addressed. This gap underscores the need to integrate sample difficulty measures into evidence-accumulation analyses of TTFS networks to predict and control inference latency under uncertainty.

In the DDM framework, the impact of sample difficulty on decision times is captured by the drift rate parameter v. Harder samples, such as stimuli with lower discriminability or higher noise, yield smaller drift rates, resulting in longer times for the decision variable to reach a boundary and thus slower and more variable response times (RT) [3, 35]. Across diverse perceptual tasks, formal fittings of the DDM consistently show that difficulty-induced reductions in drift rate account for observed changes in RT distributions under ambiguous or noisy conditions [27, 37]. Classic random-dot motion tasks illustrate this relationship: as motion coherence decreases (i.e., the stimuli becomes harder), drift rate diminishes, producing increased mean reaction times and heavier right tails in RT distributions [30, 34, 37]. Consequently, when mapping sample difficulty metrics to TTFS-coded SNNs, higher difficulty should correspond to lower effective drift rates in a DDM interpretation, predicting systematically later first-spike latencies for harder samples.

Moreover, the DDM also predicts that trials ending in an incorrect choice tend to have longer response times than correct trials. This follows from across-trial variability in drift rate: high drift rates produce fast, accurate boundary crossings, whereas low drift rates both increase error probability and slow the accumulation process, leading to a distribution of missclassified response times that is shifted toward longer latencies [35, 38]. In our TTFS context, this implies that samples with particularly small or negative margins, not only accumulate evidence more slowly on average, but when they do lead to misclassification, will evoke the longest first-spike latencies.

2.3. Spiking Neuron as a Stochastic Process

2.3.1. Membrane Potential as a Gaussian Random Walk

The sub-threshold membrane potential of leaky integrate-and-fire neurons has long been modeled as a stochastic process whose increments converge to a Gaussian distribution under the Central Limit Theorem (CLT) when driven by many weak synaptic inputs [14, 16]. Gerstein and Mandelbrot [17] first employed an inverse-Gaussian framework to describe first-passage problems in neural spiking, demonstrating that membrane dynamics under constant drift and Gaussian noise can be treated as a random walk with drift. Later works formalized the discrete-time analogue, showing that if (i) the number of presynaptic inputs is large, (ii) synaptic weights have finite variance, and (iii) presynaptic spike trains are weakly correlated, then the membrane potential sum approaches Gaussian [4, 25]. These analyses, however, predominantly consider continuous-time or asymptotic regimes and do not fully characterize the influence of finite-step effects or input “difficulty” in Time-to-First-Spike coding schemes.

2.3.2. First Hitting Time Models for Time-to-First-Spike

First-hitting time (or first-passage time) models have been extensively used to predict spike latencies in stochastic neuron models. Classic results by Siegert derived the inverse-Gaussian density for barrier crossing times in diffusion approximations of integrate-and-fire neurons [39, 40]. Chang and Peres [7] later provided rigorous bounds showing that discrete-time Gaussian random walks converge to the inverse-Gaussian limit as the time step vanishes. More recent numerical methods have improved the estimation of first-passage time densities for Ornstein–Uhlenbeck neuron models [5, 42]. Despite these foundational contributions, existing work largely addresses homogeneous synaptic inputs and continuous diffusion models; the specific effects of discrete timesteps, synaptic weight variability, sample and difficulty, remain underexplored in the context of TTFS SNNs architectures.

3. Approach

In this section, we detail the methodologies and key theoretical formulations employed in our research.

$$t_{spike}(x,y,c) = \begin{cases} \text{round}((1 - I(x,y,c)) \times T_{max}), & \text{if } I(x,y,c) > \epsilon \\ \text{no spike}, & \text{otherwise} \end{cases} \quad (2)$$

3.1. Leaky Integrate-and-Fire Neuron Model

We use the discrete-time Leaky Integrate-and-Fire (LIF) neuron model, characterized by the following update equation:

$$V_i[t+1] = \beta V_i[t] + \sum_j W_{ij} S_j[t] - V_{th} S_i[t] \quad (3)$$

Here, $V_i[t]$ denotes the membrane potential of neuron i at discrete timestep t, $\beta \in [0,1]$ is the leak parameter controlling the decay of the potential over time, W_{ij} represents the synaptic weight between neuron j (presynaptic) and neuron i (postsynaptic), $S_j[t]$ indicates whether neuron j emitted a spike at timestep t, and V_{th} is the threshold potential. If $V_i[t]$ surpasses V_{th}, neuron i emits a spike ($S_i[t] = 1$) and its membrane potential resets accordingly.

3.2. Latency Encoding of Inputs

In our implementation, latency encoding converts pixel intensities from input images into spike timings. Each pixel value, initially in the range $[0,1]$, is transformed into a spike timing such that pixels with higher intensities (closer to 1) spike earlier, and pixels with lower intensities spike later. Formally, this is described in Equation 2, where $I(x,y,c)$ represents the intensity of the pixel at location (x,y) in channel c, T_{max} denotes the maximum number of timesteps, and ϵ is a clipping threshold below which pixel intensities do not generate spikes.

3.3. Temporal Mean Squared Error Loss

To effectively train our latency-encoded SNN, we utilize a temporal mean squared error (MSE) loss function defined over spike timings [10]. Consider an output layer of neurons, each producing spike times represented as normalized timings within $[0,1]$. Given example spike times $[t_1, t_2, t_3]$ for three neurons, normalized by dividing each by the total number of timesteps T_{max} (e.g., for spike times $[1,2,3]$ and $T_{max} = 5$, the normalized times are $[0.2, 0.4, 0.6]$), the temporal MSE loss is computed as follows:

$$L = \frac{1}{N} \sum_{i=1}^{N} (t_i - y_i)^2 \quad (4)$$

where t_i is the normalized spike time of neuron i, y_i is the desired normalized target time (0 for the correct class, meaning spike as early as possible, and 1 for incorrect classes, meaning spike as late as possible or ideally never), and N is the total number of output neurons. This approach encourages the network to minimize the latency of correct class spikes while delaying or inhibiting incorrect class spikes.

One significant challenge in training with the temporal MSE loss arises from the fundamental discontinuity in spike timing mechanisms. Since the precise moment a neuron fires depends on a threshold-crossing event, the derivative of spike timing with respect to membrane potential is mathematically undefined. To circumvent this issue, we employ a commonly used custom gradient approximation during backpropagation [10]. Specifically, we set:

$$\frac{\partial t_{spike}}{\partial U} = -1 \quad (5)$$

This sign estimator establishes the directional relationship that increasing membrane potential leads to earlier firing times.

3.4. Surrogate Gradients and Backpropagation Through Time

Training SNNs presents a fundamental challenge: the spike generation mechanism is inherently non-differentiable, creating a discontinuity in the gradient flow. To address this, we employ surrogate gradients that approximate the derivative of the spiking function. For the forward pass, we use the Heaviside step function:

$$S = \begin{cases} 1 & \text{if } U \geq U_{thr} \\ 0 & \text{if } U < U_{thr} \end{cases} \quad (6)$$

For the backward pass, we utilize the gradient of a shifted arc-tangent function as our surrogate:

$$S \approx \frac{1}{\pi} \arctan(\pi U \frac{\alpha}{2}) \quad (7)$$

$$\frac{\partial S}{\partial U} = \frac{1}{\pi} \frac{1}{(1 + (\pi U \frac{\alpha}{2})^2)} \quad (8)$$

where α is a hyperparameter controlling the smoothness of approximation [11]. Additionally, since our network processes information across multiple timesteps, we use Backpropagation Through Time (BPTT), which unfolds the SNN temporally and applies backpropagation across the resulting computational graph. This allows gradients to flow backward through both spatial connections and temporal dynamics, enabling end-to-end training while preserving the temporal characteristics essential to latency-encoded networks.

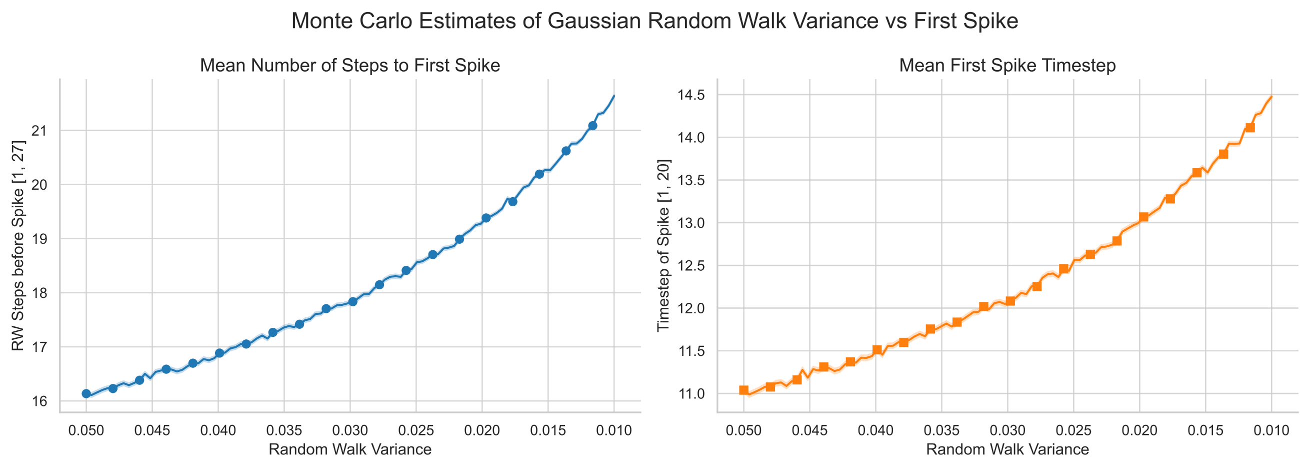


Figure 2. Monte Carlo ($N = 100\,000$) estimates of (a) mean number of steps to first spike and (b) mean first-spike timestep as functions of synaptic variance σ^2. Decreasing σ^2 leads to slower accumulation relative to leak, increasing the expected latency. N.B., the x-axis is inverted.

3.5. Discrete-Time Spiking Neuron Model and Variance Effects

We model each neuron as a non-leaky integrate-and-fire unit operating in discrete timesteps. Let S_t denote the membrane potential at timestep t. Assuming each presynaptic spike contributes a weight $w_i \in \{0,1\} \cdot \mathcal{N}(\mu, \sigma^2)$, the subthreshold dynamics form a Gaussian random walk:

$$S_t = S_{t-1} + \sum_{i=1}^{n_t} w_i \tag{9}$$

with $S_0 = 0$ and threshold $V_{\text{th}} = 1$. Under the i.i.d. Gaussian weight assumption, the first-hitting time T to reach V_{th} admits a continuous-time approximation via a Wiener process with drift $\mu > 0$ and diffusion σ^2, whose hitting-time density is inverse Gaussian [15, 24]:

$$f_T(t) = \frac{1}{\sqrt{2\pi\sigma^2 t^3}}\ \exp\Big(-\tfrac{(1-\mu t)^2}{2\sigma^2 t}\Big), \tag{10}$$

$$\mathbb{E}[T] = \frac{1}{\mu}, \quad \text{Var}(T) = \frac{\sigma^2}{\mu^3} \tag{11}$$

To capture realistic membrane leakage and finite input spikes, we introduce a decay factor $\beta \in (0, 1]$ and bin inputs into T discrete steps:

$$S_t = \beta\, S_{t-1} + \sum_{i=1}^{n_t} w_i, \qquad \sum_{t=1}^{T} n_t = N_{\max} \le 27. \tag{12}$$

Monte Carlo simulations of this leaky random walk (with $\beta = 0.95$, varying σ^2, up to $N_{\max} = 27$ for a 3×3 RGB receptive field in a convolutional layer) reveal that decreasing synaptic-weight variance shifts the first-spike distribution to the right, i.e., lower σ^2 yields heavier tails and longer mean latencies (see Figure 2).

Quantitatively, both the expected number of steps to threshold and the mean first-spike timestep increase monotonically as σ^2 decreases, confirming that reduced weight variability slows evidence accumulation relative to leak, thereby prolonging time-to-first-spike.

3.6. SepDots Synthetic Classification Task

SepDots (short for separating the dots) is a synthetic binary classification dataset designed to provide analytical control over sample difficulty via tunable class separation, enabling exact margin computation and direct investigation of first-spike latency under varying difficulty.

3.6.1. Problem Definition

$$\mathbf{x} = \begin{cases} \mathcal{N}\big([-\mu, -\mu]^T,\ \sigma^2 I\big), & \text{Class } 1, \\ \mathcal{N}\big([+\mu, +\mu]^T,\ \sigma^2 I\big), & \text{Class } 2, \end{cases} \tag{13}$$

where $\mu > 0$ controls class separation (and thus difficulty) and $\sigma^2 = 0.05$ is fixed. We simulate 20 discrete timesteps and at each timestep we draw $K = 5$ i.i.d. samples from the true class distribution (see Figure 3), where each class is chosen with probability $1/2$. Finally, sampled values are mapped to a binary 35×35 image in a linear fashion, such that [0, 0] corresponds to the center of the image.

3.6.2. Margin Computation

The optimal linear decision boundary is

$$x_1 + x_2 = 0\,. \tag{14}$$

Then, we can calculate the signed margin of a sample $\mathbf{x}$ as

$$m(\mathbf{x}) = \frac{x_1 + x_2}{\sqrt{2}}\,. \tag{15}$$

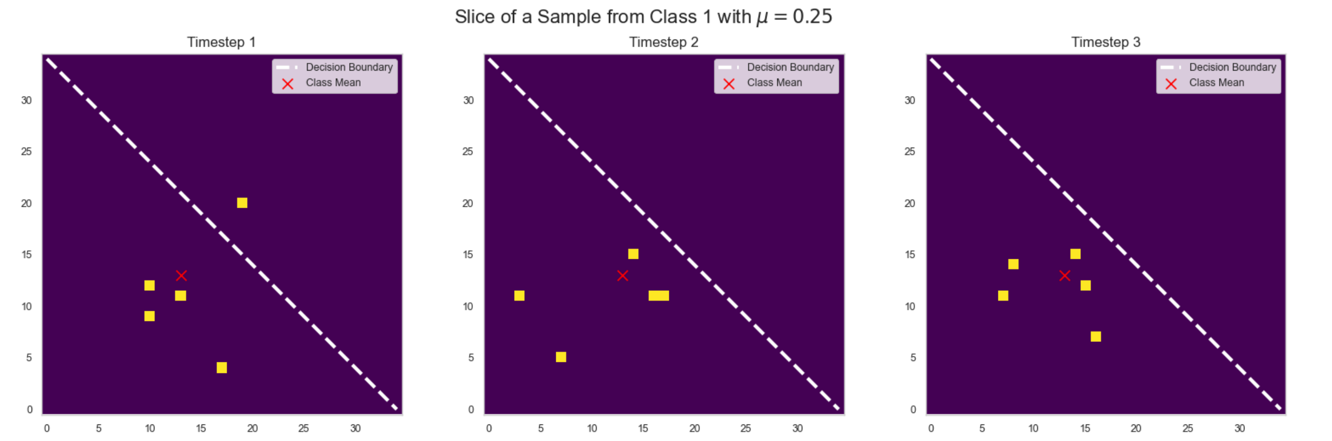


Figure 3. Three-timestep slices of a SepDots sample. Each yellow square is one of the $K = 5$ dots at that timestep and the red cross is the true class mean.

Dataset	Type	Levels
SepDots	Class mean, μ	$\{0.01, 0.025, 0.05, 0.075, 0.1, 0.15, 0.25\}$
MNIST	Gaussian noise, σ	$\{0, 0.025, 0.05, 0.075, 0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4\}$
NMNIST	Pixel-flip, p_{flip}	$\{0, 0.01, 0.02, 0.03, 0.04, 0.05, 0.06, 0.07, 0.08, 0.09, 0.1\}$
CIFAR-10	Gaussian noise, σ	$\{0, 0.025, 0.05, 0.075, 0.1, 0.15, 0.2, 0.25\}$
CIFAR-10	Gaussian blur (33×33 kernel), σ	$\{0, 0.25, 0.5, 0.75, 1, 1.25, 1.5, 1.75, 2\}$

Table 1. Perturbation parameters used to induce sample difficulty.

Let the true label be

$$y = \begin{cases} +1, & \mathbf{x} \text{ from Class } 2, \\ -1, & \mathbf{x} \text{ from Class } 1, \end{cases} \tag{16}$$

so that the class-corrected margin is

$$\tilde{m}(\mathbf{x}) = y\, m(\mathbf{x})\,. \tag{17}$$

3.6.3. Temporal Margin Aggregation

At each discrete timestep t, we draw K i.i.d. samples $\{\mathbf{x}_k^{(t)}\}_{k=1}^K$. We then compute:

$$\tilde{m}^{(t)} = \frac{1}{K}\sum_{k=1}^{K} \tilde{m}(\mathbf{x}_k^{(t)}) \quad \text{(instantaneous margin)}, \tag{18}$$

$$\widetilde{M}^{(t)} = \frac{1}{t}\sum_{\tau=1}^{t} \tilde{m}^{(\tau)} \quad \text{(cumulative margin)}. \tag{19}$$

These exact margin metrics allow us to quantitatively link sample difficulty, via $\tilde{m}$ and $\widetilde{M}$.

The key advantage of SepDots is its tunable difficulty via μ: as μ decreases, class overlap increases and the Bayes error rises, offering a fine-grained control over sample hardness. This synthetic task thus provides a clear testbed for linking evidence-accumulation predictions under the Drift Diffusion Model to actual TTFS latency under varying difficulty levels.

3.7. Experiments

We evaluate our theoretical predictions on four datasets, SepDots, MNIST, NMNIST, and CIFAR-10, by systematically varying both testing and training sample hardness. Furthermore, we repeat each experiment 30 times to increase the reliability of the results. Our two primary hypotheses are:

1. Increasing *testing* hardness will slow inference (longer first-spike latencies) due to reduced effective drift (weaker evidence accumulation).
2. Increasing *training* hardness will also slow inference, because noise-driven reductions in learned synaptic-weight variance will decrease membrane potential threshold-crossing speed.

3.7.1. SepDots

Hardness is controlled by the class-separation parameter μ, which directly modulates Gaussian overlap and margins. We train and test networks under multiple separation settings, observing how reduced μ (smaller margins) affects first-spike latency.

3.7.2. MNIST, NMNIST, CIFAR-10

For MNIST and CIFAR-10, hardness is induced by adding isotropic Gaussian noise or (only for CIFAR-10) Gaussian blur to each image prior to training and/or evaluation. For event-based NMNIST, hardness is induced by flipping each pixel with a fixed probability. However, we don't want to augment the data. Therefore, we sample each noise or blur pattern once before training and use the same perturbed dataset both during training. To keep overall spike-count statistics constant, we histogram-match the noisy inputs to their clean counterparts.

Additionally, for CIFAR-10 we use a small CNN (CNN-S) with $\approx$ 86k parameters and a large CNN (CNN-L) with $\approx$ 20M parameters.

3.7.3. Training and Evaluation Regime

The perturbation parameters we use to induce hardness in each dataset are defined in Table 1.

1. For each dataset and training hardness setting, train a TTFS-coded SNN from scratch.
2. For each trained model, evaluate on each testing hardness setting, recording first-spike latencies across output neurons.
3. Analyze the mean and variance of first-spike times as functions of training and testing hardness.

This process should allow us to separate the effects of *testing* difficulty, slower evidence accumulation, from those of *training* difficulty, changed synaptic weight statistics, on TTFS inference latency.

4. Results

Before we go into classic datasets, let's first analyze the behavior of the first spike times in our synthetic classification task SepDots.

4.1. Time-to-First-Spike Behavior

4.1.1. SepDots

Figure 4 reports the classification accuracy matrix as a function of the training-distribution mean μ_{train} (x–axis) and the testing-distribution mean μ_{test} (y–axis). Accuracy remains near 100% when $\mu_{\text{test}} \geq 0.1$ regardless of training hardness, but degrades sharply when testing separation falls below $\mu_{\text{test}} \approx 0.05$.

To probe the latency effects, we first examine the marginal relationship between cumulative margin at spike time and latency (Figure 5). Latency is longest for near-zero cumulative margins, where evidence is weakest, and decreases monotonically as $\widetilde{M}$ grows in magnitude. This brings prominence to the drift-diffusion prediction that higher net evidence yields faster first-spike times.

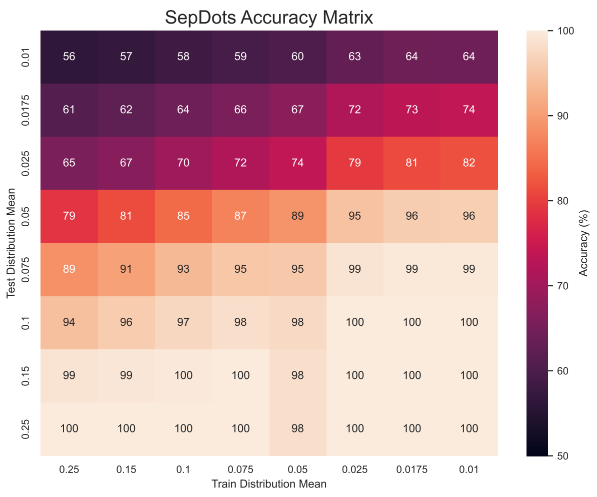


Figure 4. SepDots: Classification accuracy for varying train and test distribution means μ_{train} (x-axis) and μ_{test} (y-axis). High test separation ($\mu_{\text{test}} \geq 0.2$) yields near-perfect accuracy, while low separation degrades performance, especially when networks are trained on very hard (small-μ_{train}) data. From left-to-right harder training; from bottom-to-top harder testing.

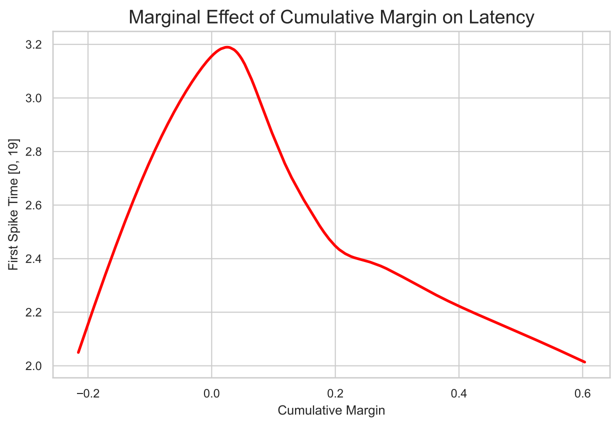
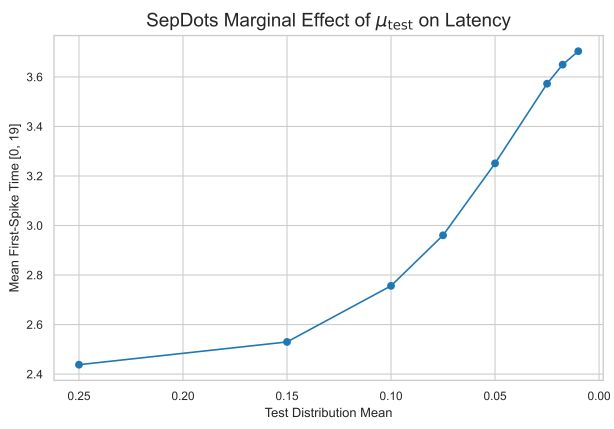
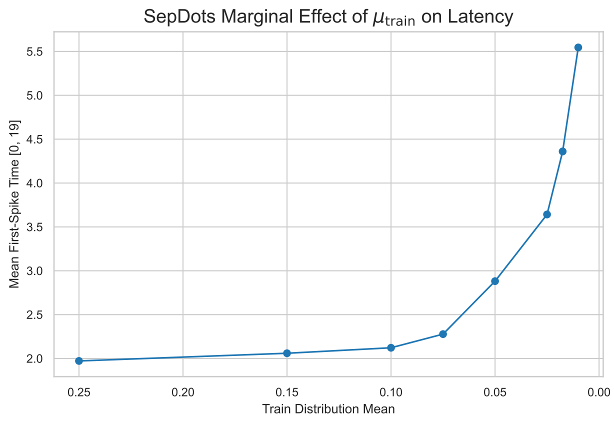


Figure 5. SepDots: Marginal effect of cumulative margin $\widetilde{M}$ at first-spike time on latency, estimated via LOWESS regression. Latency peaks near zero margin (weak evidence) and decreases as the absolute cumulative margin grows, confirming that stronger net evidence speeds first spikes.

Next, Figure 6a and Figure 6b show how mean first-spike time varies with μ_{test} and μ_{train}, respectively. Averaging out training effects, increasing μ_{test} reduces latency from $\approx$ 3.7 timesteps at $\mu_{\text{test}} = 0.01$ down to $\approx$ 2.4 at $\mu_{\text{test}} = 0.25$. Conversely, when averaging out testing hardness, raising μ_{train} speeds inference even more dramatically, from over 5.5 timesteps at $\mu_{\text{train}} = 0.01$ to just under 2.0 at $\mu_{\text{train}} = 0.25$. Thus both testing and training hardness could be independently controlling spike-latency via



(a) SepDots: Mean first-spike time versus test distribution mean μ_{test} (averaging over μ_{train}). Decreasing μ_{test} yields slower spikes. Networks tested on easier data exhibit faster inference. N.B. the x-axis is inverted to show the effect of going from easy to hard.



(b) SepDots: Mean first-spike time versus train distribution mean μ_{train} (averaging over μ_{test}). Networks trained on harder data exhibit slower inference. N.B. the x-axis is inverted to show the effect of going from easy to hard

Figure 6. Marginal effects of test-set and train-set distribution means on TTFS first-spike latency.

evidence-accumulation dynamics.

Finally, the combined heatmap in Fig. Figure 7 visualizes mean first-spike time across the full (train, test) grid. The lower-left corner (small $\mu_{\text{train}}, \mu_{\text{test}}$) exhibits the largest latencies, while the upper-right corner (large $\mu_{\text{train}}, \mu_{\text{test}}$) compresses latencies. The smooth gradient along both axes seems to provide evidence for our two hypotheses.

4.1.2. Qualitative Analysis

To motivate our quantitative analysis (which will come later in subsection 4.2) and better understand its implications, we visualize the results obtained on the NMNIST dataset. While our earlier qualitative study focused on the SepDots

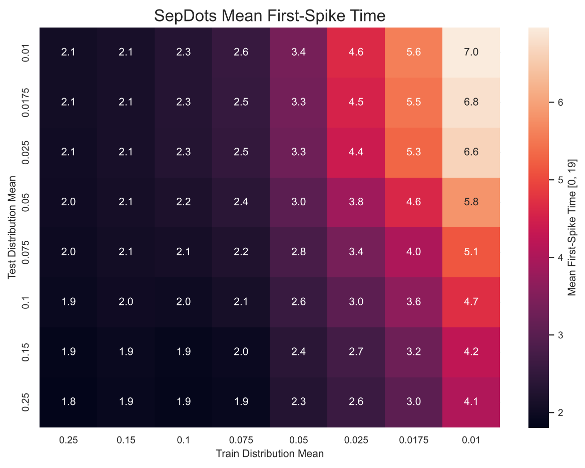


Figure 7. SepDots: Heatmap of mean first-spike time across the grid of (μ_{train}, μ_{test}) values. Latency increases smoothly from easy–easy settings in the bottom left to from hard–hard settings in the top-right, confirming both training and testing hardness effects.

dataset, we now extend this investigation to the classic NMNIST benchmark. In particular, we examine:

1. **Accuracy under noise:** how varying levels of additive input noise affect overall classification accuracy.
2. **Spiking behavior:** the temporal spike patterns produced during correctly and incorrectly classified samples as noise increases.
3. **Latency dynamics:** the impact of noise injected during training on the time-to-first-spike inference latency of the network.

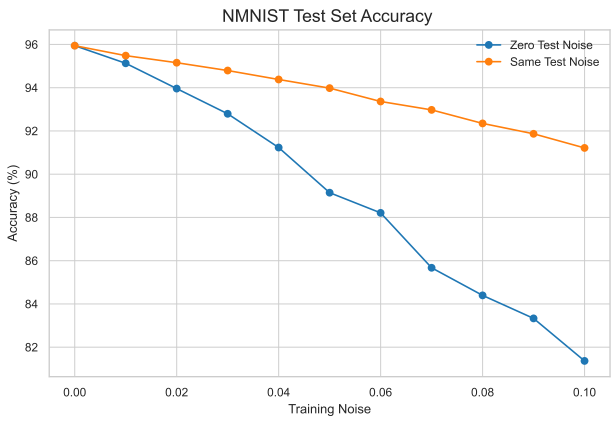


Figure 8. NMNIST test-set accuracy as a function of the training noise level. The blue curve shows performance when evaluated with zero test noise, while the orange curve shows performance when evaluated with the same noise level used during training.

Figure 8 illustrates how increasing the level of additive noise during training impacts classification accuracy on the

NMNIST test set. When evaluated without any test noise (blue line), accuracy degrades sharply as training noise increases, dropping from about 96% at zero noise to around 81% at a noise level of 0.1. By contrast, evaluating with matching test noise level with training noise level yields a more shallow decline, demonstrating that training with noise produces robustness to the similar perturbations at inference time.

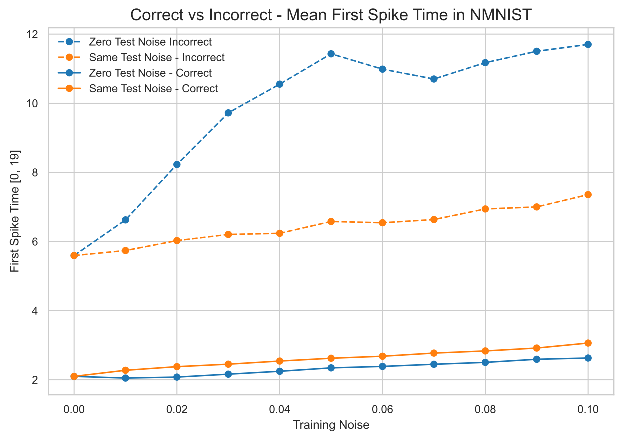


Figure 9. Mean time-to-first-spike for correctly and incorrectly classified NMNIST samples as a function of training noise. Solid lines denote correctly classified samples (blue: zero test noise; orange: matching test noise with training noise), while dashed lines denote incorrectly classified samples.

Figure 9 reveals a key insight: incorrectly classified samples (dashed lines) exhibit substantially longer TTFS compared to correctly classified ones (solid curves), indicating that misclassifications are associated with delayed spiking responses.

Figure 10 focuses on the TTFS of the correctly classified samples. In the figure, the TTFS under matching test noise (orange) consistently lies above that under zero test noise (blue), demonstrating that higher inference noise reliably increases latency. Moreover, for both test-noise settings, the mean TTFS increases monotonically with the training noise level, showing that increased training hardness prolongs output latency in all these cases.

4.2. Quantitative Analysis of Results

Prior to detailed latency analysis, we first examined how first-spike times varies between correct versus incorrect classifications (see Table 2). Since classification accuracy itself varied substantially with both training and testing noise, pooling all TTFS values would merge the effects correct vs incorrect spike times with sample difficulty. If we used all, the estimated parameters might be dominated by the change in accuracy rather than observing the real effect of training and testing noise. To isolate changes in latency

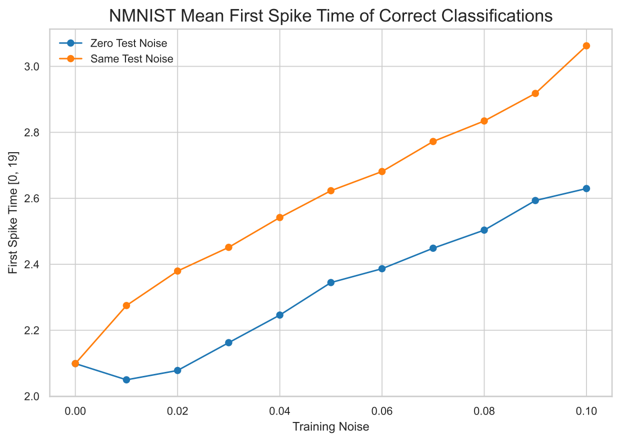


Figure 10. Mean time-to-first-spike of correctly classified NMNIST samples under zero test noise (blue) and matching test noise (orange) as a function of training noise. This figure is equivalent to Figure 9 except it's zoomed in on correct classifications

Dataset	Hardness	Model	Correct TTFS	Incorrect TTFS
SepDots	Distribution Overlap	CNN	3.04	3.47
MNIST	GWN	MLP	0.91	6.82
MNIST	GWN	CNN	2.61	8.24
NMNIST	Pixel Flip	CNN	2.75	8.39
CIFAR-10	GWN	CNN-S	3.85	4.62
CIFAR-10	GWN	CNN-L	3.88	4.58
CIFAR-10	Gaussian Blur	CNN-S	3.91	4.84

Table 2. Mean first spike times for correct and incorrect predictions across different datasets and models. The table highlights the significant difference between the first spike times of correctly classified samples and incorrectly classified samples. GWN = Gaussian White Noise

from accuracy, we do the following analysis exclusively on correctly classified samples.

To estimate quantify the effects of testing and training hardness, we fit a ordinary least squares (OLS) regression to predict TTFS as a function of training noise, testing noise, and their interaction. Formally, for test sample i:

$$\begin{aligned}\mathrm{TTFS}_i = \beta_0 + \beta_1\, \sigma_i^{\mathrm{train}} + \beta_2\, \sigma_i^{\mathrm{test}} \\ + \beta_3 \left(\sigma_i^{\mathrm{train}} \times \sigma_i^{\mathrm{test}}\right) + \varepsilon_i\end{aligned} \tag{20}$$

where $\sigma_i^{\mathrm{train}}$ and σ_i^{test} are the noise levels during training and testing, respectively, and ε_i is the residual error.

Although our primary focus is on the marginal effects

Dataset	Hardness	Model Type	Intercept β_0 ($\pm$SE)	Training Noise β_1 ($\pm$SE)	Test Noise β_2 ($\pm$SE)	Interaction β_3 ($\pm$SE)
SepDots	Distribution Overlap	CNN	4.83 (< 0.01)	$-15.5^{\dagger}$ (0.01)	$-9.17^{\dagger}$ (0.01)	45.1 (0.06)
MNIST	Gaussian White Noise	MLP	0.66 (< 0.01)	-0.30 (< 0.01)	2.23 (0.01)	-2.30 (0.02)
MNIST	Gaussian White Noise	CNN	2.34 (< 0.01)	0.37 (0.01)	1.10 (0.01)	0.62 (0.03)
NMNIST	Pixel Flip	CNN	1.94 (< 0.01)	4.13 (0.03)	23.7 (0.03)	$-212.$ (0.44)
CIFAR-10	Gaussian White Noise	CNN-S	3.78 (< 0.01)	1.21 (0.01)	-1.07 (0.01)	4.53 (0.08)
CIFAR-10	Gaussian White Noise	CNN-L	3.81 (< 0.01)	0.14 (< 0.01)	0.14 (< 0.01)	3.76 (0.06)
CIFAR-10	Gaussian Blur	CNN-S	3.70 (< 0.01)	0.11 (< 0.01)	0.12 (< 0.01)	-0.02 (< 0.01)

† For **SepDots** we *expect* β_1 and β_2 to be *negative* due to decreasing hardness with increasing distribution μ. For all other datasets, we expect $\beta_1, \beta_2 > 0$.

Table 3. OLS estimates for predicting TTFS from training noise, test noise, and their interaction (correctly classified samples only). Standard errors in parentheses. N.B., because each dataset's noise levels are defined over a different scale and exhibits a distinct functional relationship with TTFS, the magnitudes of $\beta_1, \beta_2, \beta_3$ estimates aren't directly comparable across datasets.

of training and test noise on TTFS, we additionally include the interaction term $\sigma^{\text{train}} \times \sigma^{\text{test}}$ to guard against potential moderation effects. In this context, a significant interaction would indicate that the impact of increasing test-time noise on TTFS depends on the noise regime under which the network was trained, for instance, a model trained with high noise might exhibit smaller latency shifts when exposed to further noise at test time compared to a model trained on clean data. By modelling this non-additivity, we ensure that the estimates of the main effects β_1 and β_2 remain unbiased and that the overall model fit is improved.

To evaluate how training-noise (hardness) acts as a regularizer on the learned synaptic weights, we fit the following two OLS regression models. Let σ^{train} denote the noise level used during training, and let

$$y_i^{(\text{std})} = \operatorname{StdDev}\big(\mathbf{w}_i^{\text{learn}}\big), \quad y_i^{(\text{mean})} = \operatorname{Mean}\big(\mathbf{w}_i^{\text{learn}}\big)$$

be, respectively, the standard deviation and mean of the synaptic weights measured across the 30 repeats. We then estimate

$$y_i^{(\text{std})} = \beta_4 + \beta_5\,\sigma^{\text{train}} + \varepsilon_i, \tag{21}$$

$$y_i^{(\text{mean})} = \beta_6 + \beta_7\,\sigma^{\text{train}} + \varepsilon_i. \tag{22}$$

Table 4 reports the estimates of β_4 and β_5 for (21), while Table 5 reports the estimates of β_6 and β_7 for (22).

As summarized in Table 3, we fit a OLS to predict the time-to-first-spike based on training noise intensity, test noise intensity, and their interaction. The intercept term, β_0, captures the baseline TTFS for each dataset under zero noise. For the SepDots dataset, both the training-hardness coefficient $\beta_1 = -15.5$ and the testing-hardness coefficient $\beta_2 = -9.17$ are significantly negative[1], providing evidence for our hypothesis that increasing sample difficulty decelerates inference latency. In contrast, for MNIST, NMNIST, and CIFAR-10, the noise coefficients are mostly positive, indicating that higher noise levels tend to delay spike emission, in agreement with our prior research. Finally, only SepDots exhibits a strong positive interaction ($\beta_3 = 45.1$), suggesting a compounding effect when both training and test noises are elevated.

[1]For SepDots we a priori expected $\beta_1, \beta_2 < 0$ due to decreasing hardness (and sample margins) with increasing distribution separation; for all other datasets we hypothesized $\beta_1, \beta_2 > 0$.

4.3. Effect of Test-Time Difficulty on TTFS

The estimates of the test-noise coefficient β_2 across most datasets are positive, indicating that as samples become more difficult at test time (higher σ^{test}), the mean time-to-first-spike increases. This provides direct evidence for our first research question *"What happens to the TTFS behavior of latency-encoded SNNs as samples become increasingly difficult?"*. Furthermore, the results are consistent with drift-diffusion predictions that lower drift rates yield longer response times as harder inputs slow down inference latency [3, 35]. An exception arises for CIFAR-10 with Gaussian white noise, where $\beta_2 = -1.07$ (SE = 0.01), suggesting a slight decrease in latency under added test noise. One plausible explanation is that the CNN never learned a robust feature representation on CIFAR-10 (peak accuracy $\approx 60\%$ across all training-testing combinations), so moderate noise may inadvertently regularize activations in a way that triggers earlier spikes. To an extent, this is supported by the near–zero $\beta_2 = 0.12$ in the Gaussian-blur condition, indicating very minimal sensitivity.

4.4. Effect of Training-Time Difficulty on TTFS

The training-noise coefficient β_1 quantifies whether exposing the network to harder samples during training speeds up or slows down inference, our second research question, *"Can we use harder samples during training to make inference faster?"* Except for SepDots (where hardness is inversely correlated with μ), almost all β_1 estimates are pos-

Dataset	Hardness	Model Type	Intercept β_4 ($\pm$SE)	Training Noise β_5 ($\pm$SE)
SepDots	Distribution Overlap	CNN	0.099 (0.001)	$0.163^{\dagger}$ (0.006)
MNIST	Gaussian White Noise	MLP	0.046 ($<$ 0.001)	-0.008 ($<$ 0.001)
MNIST	Gaussian White Noise	CNN	0.114 ($<$ 0.001)	0.009 (0.001)
NMNIST	Pixel-Flip	CNN	0.056 ($<$ 0.001)	-0.024 ($<$ 0.001)
CIFAR-10	Gaussian White Noise	CNN-S	0.075 ($<$ 0.001)	-0.018 ($<$ 0.001)
CIFAR-10	Gaussian White Noise	CNN-L	0.017 ($<$ 0.001)	-0.003 ($<$ 0.001)
CIFAR-10	Gaussian Blur	CNN-S	0.075 ($<$ 0.001)	-0.002 ($<$ 0.001)

† For **SepDots** we *expect* β_5 to be *positive* due to decreasing hardness with increasing distribution μ, i.e., increased standard deviation in weights with increased training distribution μ. For all other datasets, we expect $\beta_5 < 0$.

Table 4. OLS estimates for the effect of training noise on the model's **synaptic weight standard deviation**. In all cases but MNIST-CNN, the standard deviation of synaptic weights decreases with increased training hardness. Giving us a reasonable indication that increased latency during inference due to increased training noise might be stemming from decreased weight variance.

Dataset	Hardness	Model Type	Intercept β_6 ($\pm$SE)	Training Noise β_7 ($\pm$SE)
SepDots	Distribution Overlap	CNN	0.028 ($<$ 0.001)	0.065 (0.004)
MNIST	Gaussian White Noise	MLP	-0.003 ($<$ 0.001)	0.014 ($<$ 0.001)
MNIST	Gaussian White Noise	CNN	0.012 ($<$ 0.001)	0.009 (0.001)
NMNIST	Pixel-Flip	CNN	-0.003 ($<$ 0.001)	-0.005 (0.001)
CIFAR-10	Gaussian White Noise	CNN-S	0.001 ($<$ 0.001)	0.003 ($<$ 0.001)
CIFAR-10	Gaussian White Noise	CNN-L	-0.002 ($<$ 0.001)	$<$ 0.001 ($<$ 0.001)
CIFAR-10	Gaussian Blur	CNN-S	0.001 ($<$ 0.001)	$<$ 0.001 ($<$ 0.001)

Table 5. OLS estimates for the effect of training noise on model's learned **synaptic weights**. Overall, the coefficient β_7 values are near zero, except for SepDots. The effect of training hardness seems to be minimal on mean weight.

itive, indicating that noisier training regimes systematically increase TTFS and thus *slow* inference. Hence, on standard vision benchmarks, harder training does not accelerate first spikes but rather delays them, reflecting a trade-off between robustness and latency. Furthermore, in SepDots, where increasing μ makes samples easier, $\beta_1 = -15.5$ (SE = 0.01) also confirms that training on "easier" distributions quickens inference, consistent with our first-hitting-time model of spiking neuron predictions.

An interesting exception also appears for the MNIST MLP, which exhibits a slightly negative $\beta_1 = -0.30$ (SE < 0.01), suggesting that moderate Gaussian-noise training speeds up TTFS. We hypothesize this arises from an unexpected increase in the mean of synaptic weights under noise (see β_7 for MNIST-MLP in Table 5): whereas statistical learning theory predicts that adding input noise should regularize the network, reducing both the mean and variance of learned weights, the MLP instead showed an upward trend in weight means. Within our stochastic integrate-and-fire framework, an increased mean input current corresponds to a higher effective drift rate, thereby reducing first-hitting times and yielding faster spikes.

As shown in Table Table 4, the coefficient β_5 indicates that increased training hardness reduces parameter variance in all cases except the MNIST-CNN. This aligns with the concept of noise as a regularizer and provides reasonable evidence as to why inference slows down with increased training hardness. In contrast, Table Table 5 shows that β_7 remains mostly near zero, demonstrating that noise has minimal effect on the mean synaptic weight.

Finally, Figure 5 demonstrates that cumulative decision margin $\widetilde{M}$ at the spike time exhibits the inverted-U relationship with latency, very closely mimicking what drift diffusion models predict [18, 35, 36, 38]: TTFS peaks near zero margin and decreases as $|\widetilde{M}|$ grows. This suggests that in SepDots, margin could be an effective proxy for instantaneous drift rate, and that TTFS SNNs embody classic evidence-accumulation dynamics when sample difficulty is measured geometrically.

5. Conclusion

In this work, we have shown that the classical drift-diffusion model, long used to describe decision times in biological neurons and detailed spiking-network simulators, also provides a principled framework for understanding latency-encoded TTFS SNNs in machine-learning settings [44, 46, 47]. Specifically, we hypothesized that (i) harder samples, whether defined by geometric margin or synthetic Gaussian perturbations, would slow down inference by reducing the effective drift rate, and (ii) misclassified trials would exhibit longer first-spike times, mirroring DDM predictions about error-driven RT elongation. Our empirical results confirm both implications: test-noise coefficients (β_2) are overwhelmingly positive (indicating longer TTFS for harder inputs) and mean TTFS for incorrect trials is substantially higher than for correct ones across all datasets. Furthermore, the relationship between cumulative margin $\widetilde{M}$ and latency (Figure 5) exhibits the characteristic inverted-U shape of DDM first-passage times, peaking near zero evidence and decaying as $|\widetilde{M}|$ grows, thus validating geometric margin as a drift-rate proxy in TTFS SNNs.

We built a model based on the classical first-hitting-time theory for static thresholds, most notably the inverse-Gaussian model, to discrete-time leaky integrate-and-fire neurons under sample-dependent inputs. Statistical learning theory predicts that adding Gaussian noise during training is mathematically equivalent to a ridge penalty, thus reducing the variance of the learned synaptic weights [2]. Our Monte Carlo simulations then show that a reduction in weight variance alone systematically increases first-hitting times, and hence TTFS, across discrete timesteps. Moreover, the empirical training-noise coefficients β_1 from our OLS analyses provide strong evidence for this mechanism, as increased training-time hardness led to longer inference latencies in nearly all dataset–model combinations, yielding a clear accuracy-latency trade-off: test-time corruption degraded accuracy sharply unless matched by comparable training-time noise, but this matching incured slower inference. These findings bridge a gap between mathematical neuroscience and ML-oriented TTFS SNNs, demonstrating that (i) first-hitting-time analyses are directly applicable to predict SNN latency under discrete timesteps [39], and (ii) controlling synaptic-weight variance via noise-equivalent regularization provides a tunable mechanism for trading off robustness against speed in neuromorphic classifiers.

References

[1] Aurélien Bellet, Amaury Habrard, and Marc Sebban. A survey on metric learning for feature vectors and structured data. 2013. 2

[2] Christopher M Bishop. *Neural networks for pattern recognition*. 1995. 2, 12

[3] Rafal Bogacz, Eric Brown, Jeff Moehlis, Philip Holmes, and Jonathan D. Cohen. The physics of optimal decision making: a formal analysis of models of performance in two-alternative forced-choice tasks. *Psychological review*, 113: 700–765, 2006. 2, 3, 10

[4] Nicolas Brunel. Dynamics of sparsely connected networks of excitatory and inhibitory spiking neurons. *Journal of computational neuroscience*, 8:183–208, 2000. 3

[5] A. N. Burkitt. A review of the integrate-and-fire neuron model: I. homogeneous synaptic input. *Biological Cybernetics*, 95:1–19, 2006. 3

[6] Alexander Camuto, Xiaoyu Wang, Lingjiong Zhu, Chris Holmes, Mert Gürbüzbalaban, and Umut Şimşekli. Asymmetric heavy tails and implicit bias in gaussian noise injections, 2021. 2

[7] Joseph T. Chang and Yuval Peres. Ladder heights, gaussian random walks and the riemann zeta function. *Annals of Probability*, 25:787–802, 1997. 3

[8] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995. 1

[9] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995. 2

[10] Jason K. Eshraghian, Max Ward, Emre O. Neftci, Xinxin Wang, Gregor Lenz, Girish Dwivedi, Mohammed Bennamoun, Doo Seok Jeong, and Wei D. Lu. Training spiking neural networks using lessons from deep learning. *Proceedings of the IEEE*, 111(9):1016–1054, 2023. 1, 4

[11] Wei Fang, Zhaofei Yu, Yanqi Chen, Timothee Masquelier, Tiejun Huang, and Yonghong Tian. Incorporating learnable membrane time constant to enhance learning of spiking neural networks. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2641–2651, 2021. 4

[12] Charles Fefferman, Sanjoy Mitter, and Hariharan Narayanan. Testing the manifold hypothesis. *Journal of the American Mathematical Society*, 29(4):983–1049, 2013. 2

[13] Charles Fefferman, Sergei Ivanov, Matti Lassas, and Hariharan Narayanan. Fitting a manifold of large reach to noisy data. *Journal of Topology and Analysis*, 2019. 2

[14] William Feller. *An Introduction to Probability Theory and Its Applications*. John Wiley & Sons, 1968. 3

[15] J. L. Folks and R. S. Chhikara. The inverse gaussian distribution and its statistical application—a review. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 40:263–275, 1978. 5

[16] Crispin W. Gardiner. *Handbook of Stochastic Methods for Physics, Chemistry and the Natural Sciences*. Springer, 1985. 3

[17] George L. Gerstein and Benoit Mandelbrot. Random walk models for the spike activity of a single neuron. *Biophysical journal*, 4:41–68, 1964. 3

[18] Joshua I. Gold and Michael N. Shadlen. The neural basis of decision making. *Annual Review of Neuroscience*, 30(1): 535–574, 2007. 2, 11

[19] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org. 2

[20] Matthias Hein and Markus Maier. Manifold denoising. *NIPS 2006: Proceedings of the 19th International Conference on Neural Information Processing Systems*, pages 561–568, 2007. 2

[21] Dan Hendrycks and Thomas G. Dietterich. Benchmarking neural network robustness to common corruptions and surface variations, 2018. 2

[22] Irina Higgins, Le Chang, Victoria Langston, Demis Hassabis, Christopher Summerfield, Doris Tsao, and Matthew

Botvinick. Unsupervised deep learning identifies semantic disentanglement in single inferotemporal face patch neurons. *Nature Communications 2021 12:1*, 12:1–14, 2021. 2

[23] Sophie Jaffard, Giulia Mezzadri, Patricia Reynaud-Bouret, and Etienne Tanré. Spiking neural models for decision-making tasks with learning. 2025. 3

[24] Ioannis Karatzas and Steven E. Shreve. *Brownian Motion and Stochastic Calculus*. Springer, 1991. 5

[25] Bruce W. Knight. Dynamics of encoding in a population of neurons. *The Journal of general physiology*, 59:734–766, 1972. 3

[26] M. Mohri, A. Rostamizadeh, and A. Talwalkar. *Foundations of Machine Learning*. MIT Press, 2nd edition, 2018. 2

[27] M. J. Mulder, L. van Maanen, and B. U. Forstmann. Perceptual decision neurosciences - a model-based review, 2014. 2, 3

[28] Bernhard Nessler, Michael Pfeiffer, Lars Buesing, and Wolfgang Maass. Bayesian computation emerges in generic cortical microcircuits through spike-timing-dependent plasticity. *PLOS Computational Biology*, 9:e1003037, 2013. 3

[29] Redmond G. O'Connell, Paul M. Dockree, and Simon P. Kelly. A supramodal accumulation-to-bound signal that determines perceptual decisions in humans. *Nature Neuroscience*, 15, 2012. 2

[30] John Palmer, Alexander C. Huk, and Michael N. Shadlen. The effect of stimulus strength on the speed and accuracy of a perceptual decision. *Journal of Vision*, 5, 2005. 3

[31] Da Costa Gabriel B. Paranhos, Welinton A. Contato, Tiago S. Nazare, João E. S. Batista Neto, and Moacir Ponti. An empirical study on the effects of different types of noise in image classification tasks, 2016. 2

[32] Marios G. Philiastides, Hauke R. Heekeren, and Paul Sajda. Human scalp potentials reflect a mixture of decision- related signals during perceptual choices. *Journal of Neuroscience*, 34, 2014. 2

[33] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. *Proceedings of Machine Learning Research*, 139:8748–8763, 2021. 2

[34] Roger Ratcliff. A theory of memory retrieval. *Psychological Review*, 85:59–108, 1978. 2, 3

[35] Roger Ratcliff and Gail McKoon. The diffusion decision model: theory and data for two-choice decision tasks. *Neural computation*, 20:873–922, 2008. 2, 3, 10, 11

[36] Roger Ratcliff and Philip L. Smith. A comparison of sequential sampling models for two-choice reaction time. *Psychological Review*, 111:333–367, 2004. 11

[37] Roger Ratcliff and Francis Tuerlinckx. Estimating parameters of the diffusion model: Approaches to dealing with contaminant reaction times and parameter variability. *Psychonomic Bulletin and Review*, 9, 2002. 3

[38] Roger Ratcliff, Philip L. Smith, and Gail McKoon. Modeling regularities in response time and accuracy data with the diffusion model. *Current directions in psychological science*, 24:458, 2015. 3, 11

[39] Arnold J.F. Siegert. On the first passage time probability problem. *Physical Review*, 81:617, 1951. 3, 12

[40] David Siegmund. *Sequential Analysis*. Springer New York, 1985. 3

[41] Michael Reed Smith, Tony Martinez, and Cristophe Giraud-Carrier. An empirical study of instance hardness, 2010. 1

[42] Jonathan Touboul and Olivier Faugeras. A characterization of the first hitting time of double integral processes to curved boundaries. *Advances in Applied Probability*, 40:501–528, 2008. 3

[43] K. Tumer and J. Ghosh. Estimating the bayes error rate through classifier combining. In *Proceedings of 13th International Conference on Pattern Recognition*, pages 695–699 vol.2, 1996. 2

[44] Akash Umakantha, Braden A. Purcell, and Thomas J. Palmeri. Relating a spiking neural network model and the diffusion model of decision-making. *Computational brain & behavior*, 5:279, 2022. 3, 12

[45] Kush R Varshney, Alan S Wilskv, and Edwin Sibley Webster. Frugal hypothesis testing and classification. 2010. 2

[46] Xiao Jing Wang. Probabilistic decision making by slow reverberation in cortical circuits. *Neuron*, 36:955–968, 2002. 3, 12

[47] Kong Fatt Wong and Xiao Jing Wang. A recurrent network mechanism of time integration in perceptual decisions. *The Journal of neuroscience : the official journal of the Society for Neuroscience*, 26:1314–1328, 2006. 3, 12

A. Performance of Trainings

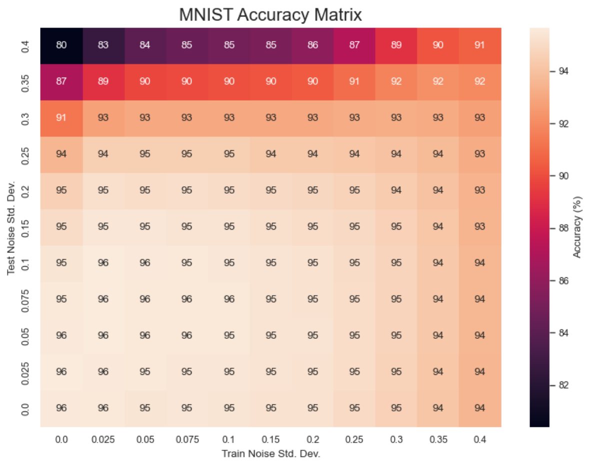


Figure 11. Accuracy matrix for MNIST-MLP

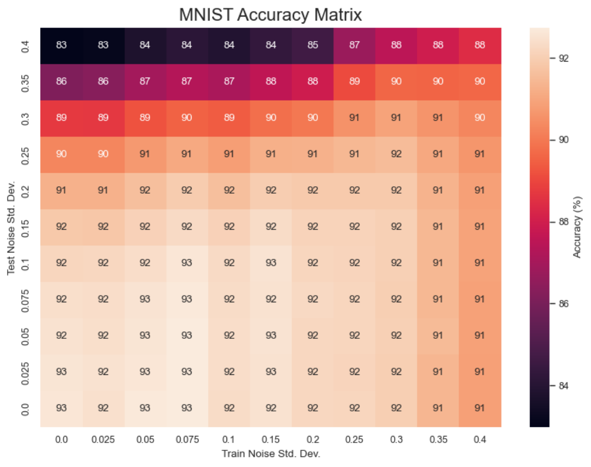


Figure 12. Accuracy matrix for MNIST-CNN

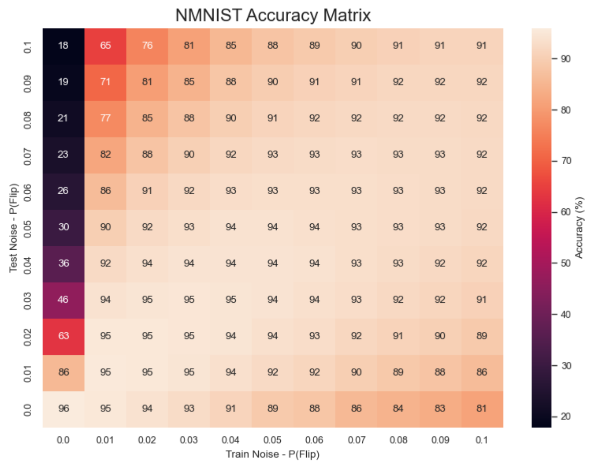


Figure 13. Accuracy matrix for NMNIST-CNN

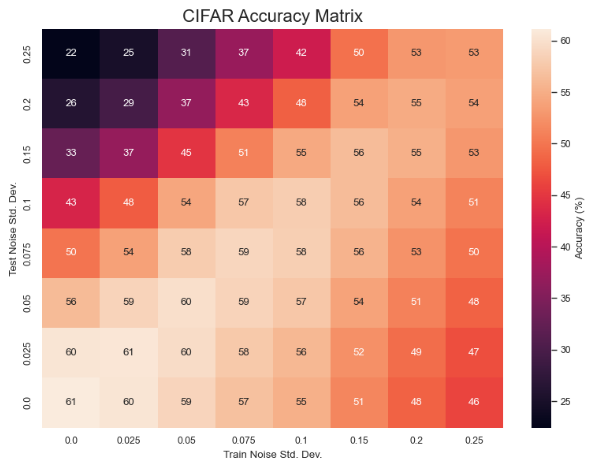


Figure 14. Accuracy matrix for CIFAR-CNN

Figure 11 up to Figure 15 present heatmaps of test accuracy (%) as a function of training-time noise (x-axis) and test-time noise (y-axis) for each dataset–model combination. Several patterns are apparent:

- **MNIST-MLP (Fig. 11).** High accuracy ($> 90\%$) is maintained across low to moderate noise levels; performance only degrades when both training and test noise exceed approximately $\sigma = 0.3$. Models trained with nonzero noise exhibit greater robustness to test-time corruption than those trained on clean images.
- **MNIST-CNN (Fig. 12).** The convolutional network achieves peak accuracies near 93% under clean conditions and shows a more gradual decline under increasing test noise compared to the MLP, sustaining $\geq 90\%$ accuracy up to $\sigma \approx 0.3$ even when trained without noise.
- **NMNIST-CNN (Fig. 13).** Baseline accuracy on event-based inputs is around 95%. Test-time pixel-flip rates above 0.02 causes very steep drops in accuracy in the clean training case, whereas injecting 5–10% flips during training shifts the robust region upward, preserving 80–90% accuracy under moderate corruption.
- **CIFAR-CNN with Gaussian Noise (Fig. 14).** Clean accuracy peaks near 60%. Small amounts of training noise ($\sigma \leq 0.05$) modestly improve clean and noisy inference, but heavy noise degrades performance across the board.
- **CIFAR-CNN with Gaussian Blur (Fig. 15).** Test-time blur with $\sigma \geq 1.0$ reduces accuracy by 10–15%. Light blur augmentation during training ($\sigma = 0.25$–0.5) partially mitigates this drop, whereas strong blur training harms both clean and blurred inference.

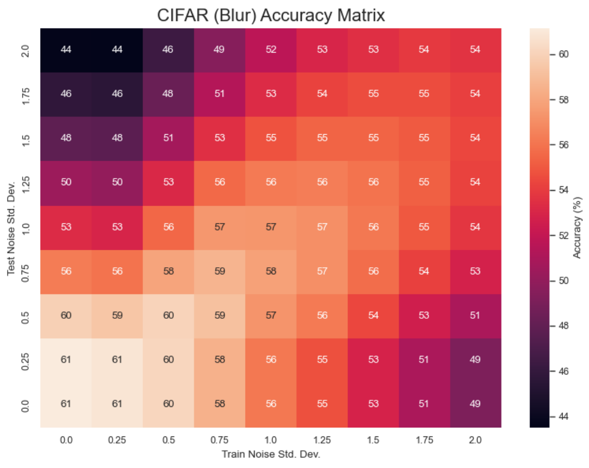


Figure 15. Accuracy matrix for CIFAR-CNN with blur

B. Noise and Synaptic Weights

The line plots in Figure Figure 16 summarize the overall shift in the first-layer weight distribution as a function of training noise standard deviation. The left plot shows the mean deviation from the noiseless baseline, and the right plot shows the variance. As σ increases further, both metrics decrease monotonically, reflecting a global *smoothing* or regularization effect as the network learns to ignore high-frequency fluctuations and focus on robust features.

However, these aggregate statistics obscure important *spatial* patterns. The heatmaps in Figure 17 reveal that at low noise levels ($\sigma = 0.0125$–0.05), the model develops large negative weights on the border pixels (bright red regions), effectively "subtracting" noisy edges instead of just ignoring them with zero-mean edges. This pixel-wise overfitting is invisible in the line plots, which simply report an average increase in variance. As noise grows beyond $\sigma \approx 0.1$, the border artifacts fade and the receptive field becomes smoother and more centrally focused on the digit shape—precisely the smoothing trend hinted at by the declining variance in Figure 13. Thus, while the line plots capture the *magnitude* of regularization, the heatmaps expose the *location* and *nature* of the weight adjustments induced by noise training.

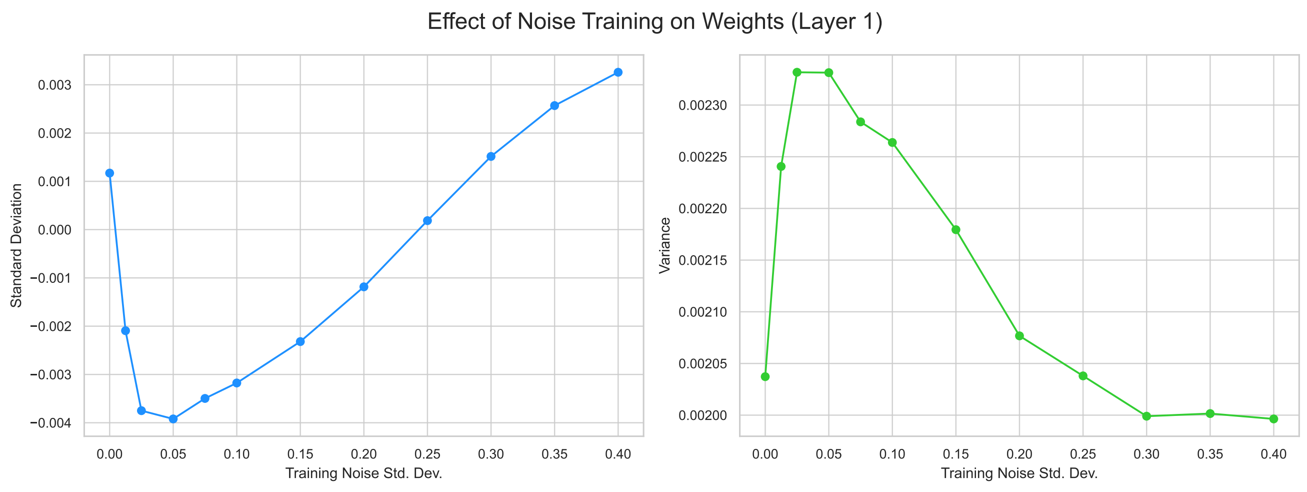


Figure 16. Statistics of first-layer weights in MNIST-MLP as a function of training noise standard deviation.

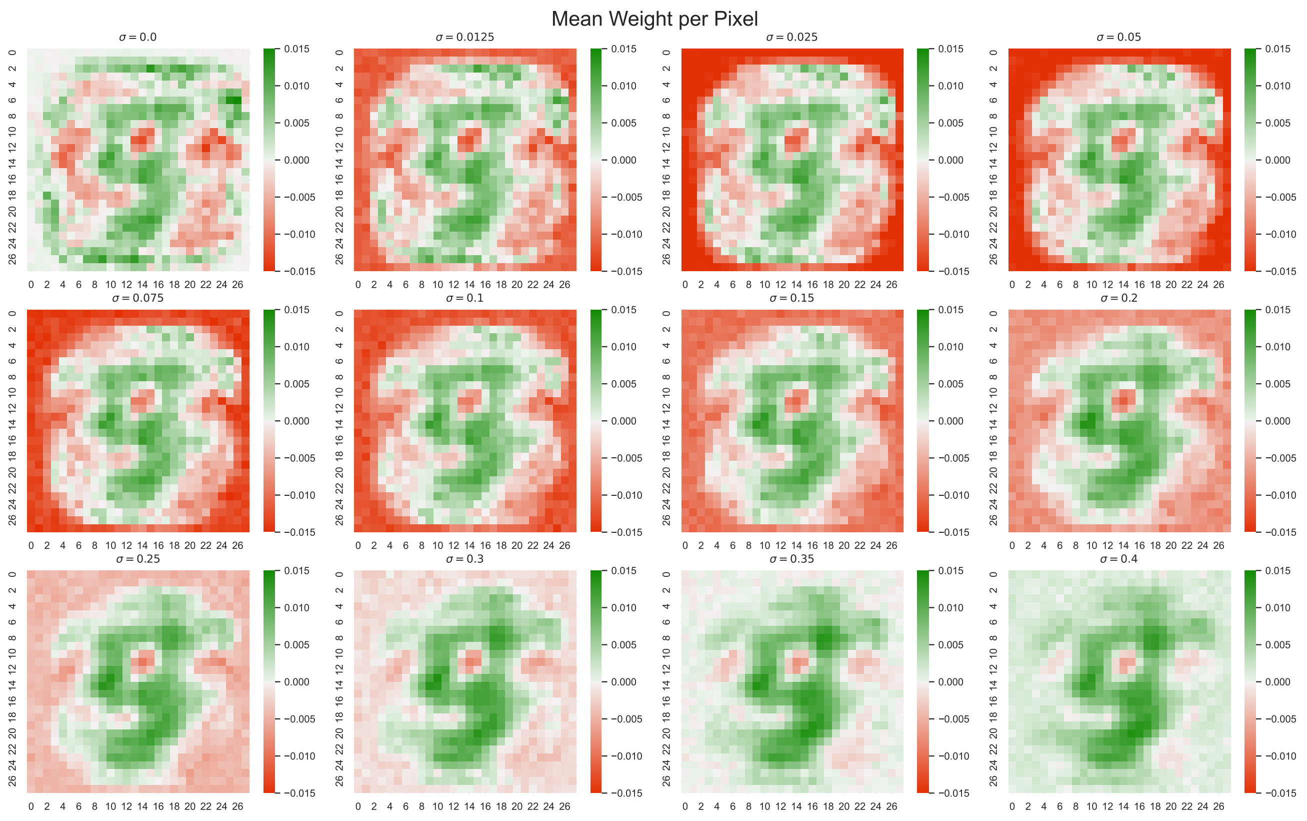


Figure 17. Pixel-wise mean edge weights in MNIST-MLP for various training noise levels.