A Novel Diffusion-Guided Hybrid Model for High-Accuracy Accident Detection in Intelligent Transportation Systems

Siva Sai*, Saksham Gupta*, Vinay Chamola *Senior Member, IEEE*, Rajkumar Buyya *Fellow, IEEE*

***Abstract*—The integration of diffusion models into Intelligent Transportation Systems (ITS) offers a promising direction for image-based accident detection. We present a diffusion-guided hybrid classification model, where "hybrid" refers to the combination of an EfficientNet guidance classifier with a conditional diffusion classifier. By leveraging fine-tuned EfficientNet architecture outputs as input for our proposed diffusion model and processing image tensors as our conditioning, our approach creates a robust classification framework. Our model consists of multiple conditional modules, which aim to modulate the linear projection of inputs using time embeddings and image feature embeddings, allowing the network to adapt its behavior dynamically throughout the diffusion process. To address the computationally intensive nature of diffusion models, we keep the number of reverse-diffusion steps small, since the sampling cost of the reverse process grows linearly with the number of timesteps. Our strategy overcomes the shortcomings of conventional classification approaches by leveraging diffusion models' inherent capacity to effectively understand complicated data distributions. We investigate important diffusion characteristics, such as timestep schedulers, timestep encoding techniques, timestep count, and architectural design changes, using a thorough ablation study, and have conducted a comprehensive evaluation of the proposed model against the baseline models on two publicly available datasets. The proposed diffusion model achieved 97.32% accuracy on the reported test split.**

***Index Terms*—Diffusion Models, Accident Detection, Intelligent Transportation Systems, Conditional Diffusion.**

I. Introduction

Intelligent Transportation Systems (ITS) have benefited greatly from the rapid growth of artificial intelligence (AI), which has had a huge impact on many other industries [1]. A key component of ITS is accident detection, which is essential to maintaining traffic safety and streamlining traffic control. Rapid emergency responses, fewer secondary events, and less traffic congestion are all made possible by early and precise accident detection [2]. Traffic cameras, sensors installed on roads, and manual reporting systems have all been used in traditional accident detection methods [1]. Although somewhat successful, these approaches frequently have drawbacks in terms of coverage, real-time processing, and the capacity to decipher intricate situations [2]. AI-driven techniques have recently been included to improve accident detection capabilities. For example, machine learning algorithms may evaluate sensor data to discover abnormalities suggestive of collisions, and computer vision techniques can be used to analyze video feeds to identify accidents [1]. The dynamic and unstructured nature of real-world data, however, may be difficult for these models to handle, and they frequently require intensive feature engineering. Furthermore, difficulties in incorporating sophisticated AI models into real-world applications have been noted, including high computational costs and constrained processor power [2].

There are still a number of issues with accident detection models, even with the improvements made by conventional machine learning approaches. These include issues with managing unstructured data, processing in real-time, and adjusting to various and changing traffic situations [1]. Diffusion models have been suggested as a viable way to address these issues [3]. Our objective is to tackle classification problems through a generative modeling framework utilizing diffusion models, leveraging the inherent stochasticity of their outputs to improve the accuracy of an underlying deterministic classifier. In comparison to other generative architectures like GANs and VAEs, diffusion models capture a richer latent space representation owing to the introduction of timesteps. Although GANs generate high-fidelity samples, they do not match the distribution of real data as effectively, whereas VAEs, while good for representation learning and uncertainty estimation, struggle to generate samples as sharp and detailed as diffusion models. Hence, the use case of classification via generative architectures is best suited to diffusion models. Diffusion models can be integrated into ITS to improve planning and decision-making, which will result in more precise and faster accident detection.

In order to provide a comprehensive grasp of traffic conditions and possible accident situations, diffusion models may also be expanded to incorporate data from a variety of sources, including social media, meteorological information, and traffic sensor data. In summary, diffusion models provide a promising direction for image-based accident detection in ITS. By using these models' advantages, ITS can identify accidents more thoroughly, accurately, and

* Equal contribution

This work was supported by CHANAKYA Fellowship Program of TIH Foundation for IoT & IoE (TIH-IoT) received by Dr. Vinay Chamola under Project Grant File CFP/2022/027.

Siva Sai is with the Department of Electrical and Computer Engineering, National University of Singapore, Singapore (e-mail: siva.sai@nus.edu.sg).

Saksham Gupta and Vinay Chamola are with the Department of Electrical and Electronics Engineering, BITS-Pilani, Pilani Campus, 333031, India. (e-mails: {f20220758, vinay.chamola}@pilani.bits-pilani.ac.in).

Rajkumar Buyya is with The Quantum Cloud Computing and Distributed Systems (qCLOUDS) Laboratory, School of Computing and Information Systems, The University of Melbourne, Australia (e-mail: rbuyya@unimelb.edu.au).

quickly, which will eventually improve road safety and make transportation systems more effective. Our proposed diffusion-guided classification model combines a guidance classifier with a conditional diffusion classifier. It utilizes softmax outputs from a fine-tuned EfficientNet model as the mean for the diffusion model's final timestep, while image tensors serve as covariates processed through an encoder. The architecture features multiple Conditional Module layers incorporating timestep embeddings, enhanced with softmax activation functions and batch normalization. A key checkpoint multiplies the image encoder's output features with processed labels before proceeding through four Conditional Module blocks to produce the final output. Our proposed model showed better Accuracy, F1-Score, Precision, and Recall values compared to conventional CNN architectures, namely ResNet50, ResNet18, and GoogLeNet. Our model showed the highest accuracy of 97.32%.

The main contributions of the paper are listed as follows:

- We adapt conditional diffusion classification with image-based guidance to the vehicle accident-detection setting.
- We identify the best-performing diffusion configuration among scheduler, timestep encoding, and architecture choices for this accident-detection setting.
- We analyze how the timestep count affects training stability, accuracy, and loss for the attention-based variant, showing the tradeoff between shorter sampling and stronger loss behavior.

The rest of the paper is organized as follows. In Section II, we cover the works related to the proposed model. We discuss the proposed methodology in Section III followed by a discussion on the experimentation results in Section IV. Finally, we conclude the paper in Section V.

II. Related Work

Classification models have been applied extensively to accident detection. Earlier work focused on traditional machine learning pipelines like Random Forest, SVM, and gradient boosting [4] [5]. Detection in connected vehicles has also been approached without images, for example through a trust-based distributed ledger that flags anomalous vehicles [6]. With the advent of deep learning, multilayer perceptrons (MLPs) and, later, parameter-efficient convolutional neural networks (CNNs) were adopted for the task. The work in [7] proposes a CNN framework to recognize congestion in varied scenarios using visual data. The architecture of the proposed model includes parallel residual convolutional units, inspired by the ResNet architecture. The work in [8] tackles real-time accident detection using a combination of YOLO [8] and CNN frameworks which work together to improve scalability by designing the system to be compatible with multiple camera feeds for urban environments. Other notable directions include the introduction of ViTs (Vision Transformers). The work in [9] introduces multi-phase frameworks that combine YOLO, ByteTrack, and Vision Transformers to precisely detect accident events and mitigate false positives in real-world traffic scenes, showing superiority over standard CNN-based models. More recently, Fang et al. [10] extended accident analysis from frame-level detection to ego-view accident video understanding.

Generative adversarial networks were the first generative family applied to this problem, and they supersede the discriminative models above by learning the data distribution rather than a decision boundary alone. The work in [11] leverages GANs for accident detection. Designed for real-time accident detection from CCTV, the approach is suited for smart city surveillance and emergency management integration. The authors tackle the problem of accident detection using three models: standard CNN, fine-tuned CNN (FTCNN), and Vision Transformer (ViT). The authors in [12] developed a Class-conditioned Consistency GAN for anomaly detection that conditions generation on class labels and incorporates consistency regularization. It applies anomaly detection in traffic data, enhancing the quality of synthetic anomaly samples for classifier training. By adopting this framework, the authors supersede vanilla GAN architectures which often run into model collapse.

In recent years, diffusion-based deep generative models have shown remarkable success in capturing complex high-dimensional, multi-modal distributions [13]–[18]. Several recent works employed diffusion models for generative modeling in Intelligent Transportation Systems. Sai et al. [19] applied a diffusion model directly to vision-based accident detection in vehicular networks, using it to synthesize additional accident imagery for training, and a related diffusion-guided semantic augmentation method has been proposed for vehicular IoT data [20]. In both cases the diffusion model acts as a generator that feeds a separate downstream classifier, whereas the present work uses the conditional reverse process as the classification stage itself.

The work in [21] introduced a Conditional Diffusion Framework with Spatio-Temporal Estimator (CDSTE) to predict dynamic traffic states and evolution with uncertainty modeling and the work in [22] uses observable traffic data as conditional information in the reverse process of diffusion to fill in missing traffic data. The work in [23] introduced a transformer-based diffusion model with a combination of enhanced cross-attention to model detailed car-following dynamics. This improves accuracy in long-term vehicle trajectory forecasting in autonomous driving contexts. The work in [24] introduced knowledge-guided conditioning to diffusion for generating realistic, controllable traffic data useful for simulation and anomaly detection studies. The work in [25] introduced a novel implicit-explicit conditional diffusion model designed for imputing missing entries in incomplete traffic datasets. They do this by using the features in a dual-stream architecture that separately captures the temporal characteristics and frequency domain patterns.

We hypothesize that diffusion models offer a useful conditional-distribution perspective for classification tasks. The work in [26] introduced extensions of diffusion models for categorical data, and [27] proposed diffusion models for discrete data as a generalization of multinomial diffusion models. Diffusion-based classification has also been applied to medical images, for example by DiffMIC [28],

Table I: Comparative analysis of related works and the proposed model, restricted to methodological attributes.

Work	Core Methodology	Generative Modeling	Diffusion
Jiang et al. [7]	Convolutional Neural Network (CNN)	×	×
Nusari et al. [8]	YOLO + CNN	×	×
Jartarghar et al. [9]	YOLO + ByteTrack + Vision Transformer (ViT)	×	×
Xi et al. [11]	Generative Adversarial Network (GAN) + CNN/ViT	✓	×
Wang et al. [12]	Class-conditioned Consistency GAN (C3-GAN)	✓	×
Proposed Model	**Guided Diffusion Model + EfficientNet**	✓	✓

Table II: Notation used in the diffusion-guided classification model.

Symbol	Meaning
$\mathbf{w}$	Input accident-scene image.
z_0, z_t	Clean and noised class-label representation at timestep t.
t, T	Diffusion timestep and total number of timesteps.
δ_t, γ_t	Signal-retention factor and noise variance, $\gamma_t = 1 - \delta_t$.
$\bar{\delta}_t$	Cumulative product $\prod_{i=1}^{t} \delta_i$.
$h_\phi(\mathbf{w})$	EfficientNet guidance classifier.
$g_\psi(\mathbf{w})$	Soft guidance vector from the classifier output.
ϵ, ϵ_θ	Sampled noise and the network's predicted noise.
μ_θ, Σ_θ	Reverse-transition mean and fixed diagonal covariance $\tilde{\gamma}_t I$.
$\tilde{\gamma}_t$	Posterior variance $\frac{1-\bar{\delta}_{t-1}}{1-\bar{\delta}_t}\gamma_t$ of the forward-process posterior.
λ_0, λ_1, λ_2	Weights of z_0, z_t, and $g_\psi(\mathbf{w})$ in the posterior mean.

a dual-guidance diffusion network. Our work builds on CARD-style conditional diffusion rather than presenting it as a new diffusion formulation. Its contribution is the accident-detection adaptation, the EfficientNet-guided image-conditioning path, and the accompanying architecture, schedule, embedding, and timestep analysis. Table I presents a comparative analysis of related works and the proposed model.

III. Proposed Methodology

Our proposed model architecture is illustrated in Figure 1, which provides a visual representation of the entire pipeline. The diagram shows how the guidance classifier's softmax outputs are integrated with the diffusion process, highlights the Conditional Module layers with their respective timestep embeddings, and demonstrates the crucial multiplication checkpoint between encoder features and processed labels. Figure 1 also depicts the four Conditional Module blocks and their interconnections, offering a comprehensive overview of our hybrid classification approach.

In the notation below, $\mathbf{w}$ denotes the input accident-scene image, z_0 denotes the class-label representation, $h_\phi(\mathbf{w})$ denotes the EfficientNet guidance classifier, and $g_\psi(\mathbf{w})$ denotes the soft guidance vector supplied by that classifier to the diffusion model. Table II summarizes the notation used throughout this section.

The model is composed of a guidance classifier, which is subjected to classification via a diffusion process. The guidance classifier consists of a modified architecture of a pre-trained EfficientNet model where the initial convolutional layer and the final fully connected layer are fine-tuned. The fine-tuned layers of the model allow it to adapt to the dataset. Fine-tuning the model contributes to improved feature extraction and generalization. These layers efficiently extract spatial features using depthwise separable convolutions, as evidenced by consistent performance across training and validation metrics, hence improving classification performance (Section IV). The pre-trained model weights of the guidance classifier are used in the diffusion classification.

Our model consists of multiple Conditional Modules. This component modulates and injects the linear projection of inputs using time-dependent embeddings, allowing the network to adapt its behavior dynamically throughout the diffusion process. We consider two timestep embedding choices.

- Learnable (Linear) Embeddings: implemented via an embedding layer, these embeddings are learned during training and provide flexibility in representing the diffusion timestep.
- Fixed (Sinusoidal) Embeddings: fixed sinusoidal embeddings, inspired by the positional encodings in Transformer architectures, enable the model to distinguish between different noise levels during the denoising process.

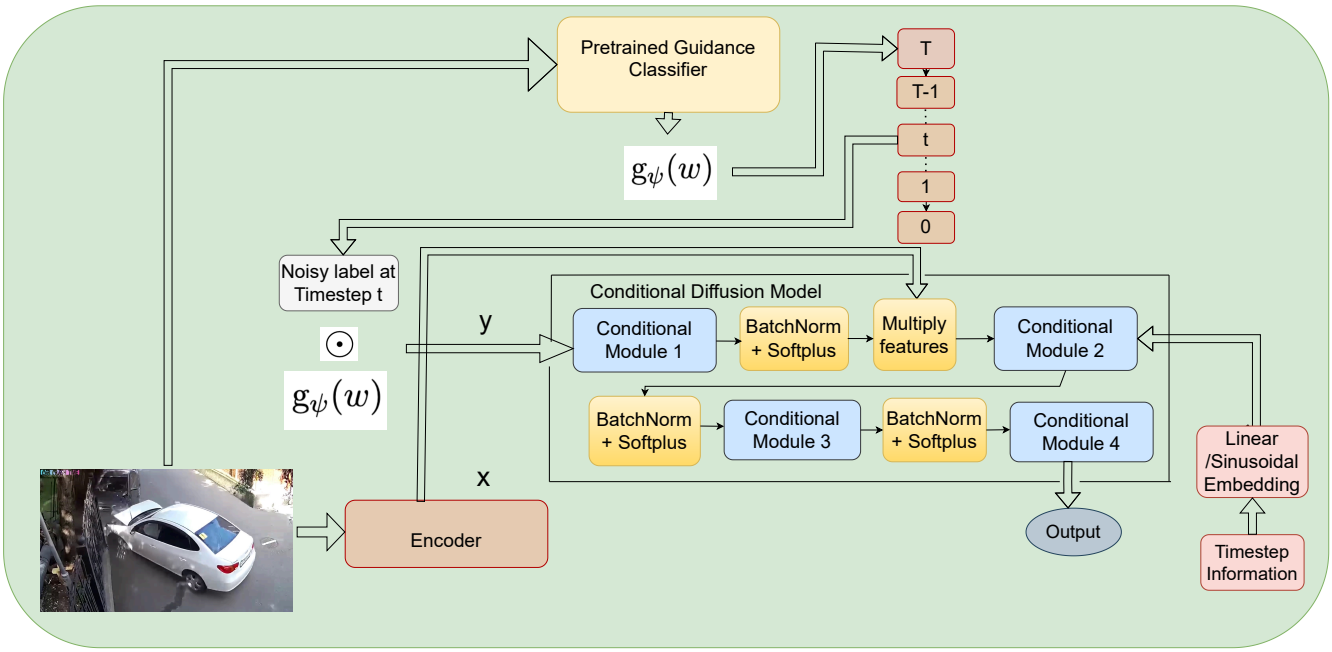


Figure 1: Architecture of the proposed diffusion-guided classification model.

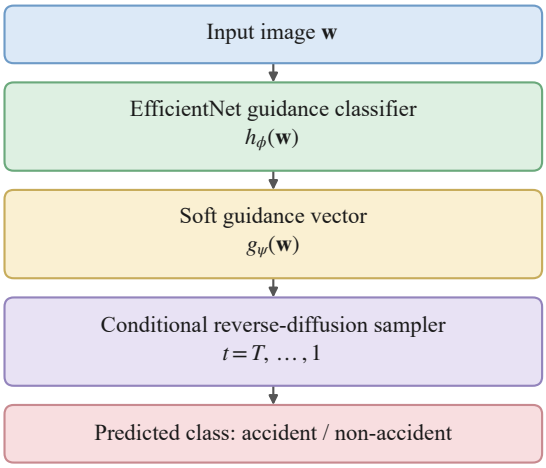


Figure 2: Prediction flow for one input image.

A. *Classification Through Diffusion Models*

Training diffusion models involves optimizing the evidence lower bound (ELBO), which can be decomposed into a series of KL divergences between the forward and reverse processes. This is done at each timestep and hence allows the model to train stably across noise levels. Classification through diffusion models involves conditioning on covariates w and on the predictions of the guidance classifier. We extend the concept of Denoising Diffusion Models [13] to Diffusion Models that have the ability to classify categories by integrating classification objectives into the diffusion process [29]. Intuitively, the ELBO below trains the reverse process to undo the artificial noise added to the label representation while remaining conditioned on the image and the guidance classifier output. A model is trained such that the log-likelihood is maximized by optimizing the following ELBO:

$$\log p_\theta(z_0 \mid w) = \log \int p_\theta(z_{0:T} \mid w)\, dz_{1:T} \tag{1}$$

$$\geq \mathbb{E}_{q(z_{1:T}|z_0,w)}\left[\log \frac{p_\theta(z_{0:T} \mid w)}{q(z_{1:T} \mid z_0, w)}\right] \tag{2}$$

where $q(z_{1:T}|z_0, w)$ is called the forward process or diffusion process in diffusion models. We denote $D_{KL}(q \parallel p)$ as the Kullback-Leibler (KL) divergence from distribution p to distribution q. The objective can be rewritten as:

$$L_{ELBO}(z_0, w) := L_0(z_0, w) + \sum_{t=2}^{T} L_{t-1}(z_0, w) + L_T(z_0, w) \tag{3}$$

$$L_0(z_0, w) := E_q\left[-\log p_\theta(z_0 \mid z_1, w)\right] \tag{4}$$

$$L_{t-1}(z_0, w) := E_q\left[D_{KL}\left(q(z_{t-1} \mid z_t, z_0, w) \parallel p_\theta(z_{t-1} \mid z_t, w)\right)\right] \tag{5}$$

$$L_T(z_0, w) := E_q\left[D_{KL}\left(q(z_T \mid z_0, w) \parallel p(z_T \mid w)\right)\right] \tag{6}$$

We follow the convention to assume L_T does not depend on any parameter and will be close to zero by carefully diffusing the observed response variable z_0 towards a pre-assumed distribution $p(z_T|w)$. The remaining terms will make the model $p_\theta(z_{t-1}|z_t, w)$ approximate the corresponding tractable ground-truth denoising transition step $q(z_{t-1}|z_t, z_0, w)$ for all timesteps. Unlike vanilla diffusion models, we assume the endpoint of our diffusion process to be:

$$p(z_T|w) = N(g_\psi(w), I) \tag{7}$$

where $g_\psi(w)$ represents prior knowledge of the relation

between w and z_0, i.e., the pre-trained guidance classifier. This endpoint means that the diffusion trajectory is not pushed toward an unconditional standard Gaussian only; instead, it is centered around the EfficientNet guidance prediction, so the denoising model starts from a classifier-informed prior.

Algorithm 1 Training (Classification)

Require: Pre-train EfficientNet guidance classifier $h_\phi(\mathbf{w})$ and use its soft output $g_\psi(\mathbf{w})$ as the diffusion guidance vector

repeat
  Draw $\mathbf{z}_0 \sim q(\mathbf{z}_0 \mid \mathbf{w})$
  Draw $t \sim \text{Uniform}(\{1, \ldots, T\})$
  Draw $\epsilon \sim \mathcal{N}(0, I)$
  Compute noise estimation loss:

$$\mathcal{J}_\epsilon = \Big\| \epsilon - \epsilon_\theta \Big(\mathbf{w},\ \sqrt{\bar{\delta}_t}\, \mathbf{z}_0 + \sqrt{1-\bar{\delta}_t}\, \epsilon + \big(1 - \sqrt{\bar{\delta}_t}\big)\, g_\psi(\mathbf{w}),\ g_\psi(\mathbf{w}),\ t \Big) \Big\|^2$$

  Take numerical optimization step on: $\nabla_\theta \mathcal{J}_\epsilon$
until Convergence

Algorithm 2 Conditional Reverse-Diffusion Inference

Require: Input image $\mathbf{w}$, trained noise predictor ϵ_θ, and EfficientNet guidance classifier h_ϕ

Compute the soft guidance vector $g_\psi(\mathbf{w}) = h_\phi(\mathbf{w})$
Draw $\mathbf{z}_T \sim \mathcal{N}(g_\psi(\mathbf{w}), I)$
for $t = T, \ldots, 1$ **do**
  Predict $\hat{\epsilon} = \epsilon_\theta(\mathbf{w}, \mathbf{z}_t, g_\psi(\mathbf{w}), t)$
  Estimate $\hat{\mathbf{z}}_0$ from $\mathbf{z}_t$, $\hat{\epsilon}$, and $g_\psi(\mathbf{w})$
  Form the conditional posterior mean $\tilde{\mu}(\mathbf{z}_t, \hat{\mathbf{z}}_0, g_\psi(\mathbf{w}))$
  if $t > 1$ **then**
    Draw $\mathbf{z}_{t-1}$ from the fixed-diagonal posterior covariance $\tilde{\gamma}_t I$
  else
    Set $\mathbf{z}_0 = \hat{\mathbf{z}}_0$
  end if
end for
return the class whose one-hot prototype is closest to $\mathbf{z}_0$

The objective detailed in (3) is optimized through an iterative training process, which is summarized in Algorithm 1. The model learns to estimate the noise prediction ϵ_θ by leveraging the input image $\mathbf{w}$ and the guidance classifier's prediction $h_\phi(\mathbf{w})$. Algorithm 1 details the training procedure for our proposed model. The process begins with a pre-training step for the EfficientNet guidance classifier $h_\phi(\mathbf{w})$; its softmax output is denoted by $g_\psi(\mathbf{w})$ when used inside the diffusion equations. The core training loop draws the clean label representation $\mathbf{z}_0$, samples a timestep t, adds Gaussian noise, and optimizes the conditional diffusion model to predict that noise. Algorithm 2 gives the corresponding conditional reverse-diffusion sampler used to obtain the final class decision. In vanilla diffusion models, in the forward process, noise is incrementally added to the data over a series of time steps, effectively transforming structured data into noise. This process is typically modeled as a Markov chain, where each step adds Gaussian noise to the data:

$$q(z_t \mid z_{t-1}) = N\big(z_t; \sqrt{\delta_t}\, z_{t-1},\ (1-\delta_t) I\big) \tag{8}$$

The above can be extended to classification diffusion models. With a diffusion schedule $\{\gamma_t\}_{t=1:T} \in (0,1)^T$, we specify the forward process conditional distributions similarly for all timesteps including $t = 1$:

$$q(z_t \mid z_{t-1}, g_\psi(w)) = \mathcal{N}\Big(z_t;\ \sqrt{\delta_t}\, z_{t-1} + \Big(1 - \sqrt{\delta_t}\Big)\, g_\psi(w),\ \gamma_t I\Big) \tag{9}$$

where,

$$\gamma_t = 1 - \delta_t \tag{10}$$

Here, z_t represents the data at time step t, γ_t is a variance schedule controlling the noise level at each step, and N denotes a Gaussian distribution. As t increases, z_t becomes progressively noisier, approaching a standard normal distribution as t approaches T. For classification, this process can be read as gradually corrupting the class-label representation while retaining information from the image-conditioned guidance vector. The cumulative factor $\bar{\delta}_t$ below summarizes how much of the original label signal remains after t noising steps.

Let $z_0 \in \mathbb{R}^d$ be the original data. The forward diffusion process is defined as a Markov chain:

$$q(z_{1:T} \mid z_0) = \prod_{t=1}^{T} q(z_t \mid z_{t-1}) \tag{11}$$

Defining $\bar{\delta}_t = \prod_{i=1}^{t} \delta_i$, one can show by the properties of Gaussian distributions that the distribution of the noisy image at any timestep can be defined based on the original equation as follows

$$q(z_t \mid z_0) = N\big(z_t; \sqrt{\bar{\delta}_t}\, z_0,\ (1-\bar{\delta}_t) I\big) \tag{12}$$

Similarly, for the conditional model, the forward process admits a closed-form sampling distribution at an arbitrary timestep t:

$$q(z_t | z_0, g_\psi(w)) = N(z_t; \sqrt{\bar{\delta}_t} z_0 + (1 - \sqrt{\bar{\delta}_t}) g_\psi(w), (1-\bar{\delta}_t) I) \tag{13}$$

The forward process sampling distribution for any arbitrary timestep is obtained via the following process:

Following Pandey et al. [30], the expectation term follows from the forward process equation. From earlier, we have that for all $t = 1, \ldots, T$,

$$z_t = \sqrt{1-\gamma_t} z_{t-1} + (1 - \sqrt{1-\gamma_t}) g_\psi(w) + \sqrt{\gamma_t}\epsilon, \quad \text{where } \epsilon \sim N(0, I) \tag{14}$$

Taking the expectations of both sides, we have

$$E(z_t) = \sqrt{1-\gamma_t}\, E(z_{t-1}) + \left(1 - \sqrt{1-\gamma_t}\right) g_\psi(w) \quad (15)$$

$$= \sqrt{1-\gamma_t}\left[\sqrt{1-\gamma_{t-1}}\, E(z_{t-2}) + \left(1 - \sqrt{1-\gamma_{t-1}}\right) g_\psi(w)\right] + \left(1 - \sqrt{1-\gamma_t}\right) g_\psi(w) \quad (16)$$

$$= \sqrt{(1-\gamma_t)(1-\gamma_{t-1})}\, E(z_{t-2}) + \left(1 - \sqrt{(1-\gamma_t)(1-\gamma_{t-1})}\right) g_\psi(w) \quad (17)$$

Continuing this process recursively:

$$= \sqrt{\prod_{i=2}^{t}(1-\gamma_i)}E(z_1) + \left(1 - \sqrt{\prod_{i=2}^{t}(1-\gamma_i)}\right) g_\psi(w) \quad (18)$$

$$= \sqrt{\prod_{i=2}^{t}(1-\gamma_i)}\left[\sqrt{1-\gamma_1}z_0 + (1-\sqrt{1-\gamma_1})g_\psi(w)\right] + \left(1 - \sqrt{\prod_{i=2}^{t}(1-\gamma_i)}\right) g_\psi(w) \quad (19)$$

$$= \sqrt{\prod_{i=1}^{t}(1-\gamma_i)}z_0 + \left(1 - \sqrt{\prod_{i=1}^{t}(1-\gamma_i)}\right) g_\psi(w) \quad (20)$$

$$= \sqrt{\bar{\delta}_t}z_0 + (1-\sqrt{\bar{\delta}_t})g_\psi(w) \quad (21)$$

The mean term can be seen as an interpolation between true data z_0 and the predicted conditional expectation $g_\psi(w)$, which gradually shifts from the former to the latter throughout the forward process.

Further, in the reverse diffusion process, the goal is to learn the reverse of the diffusion process to generate new data samples from noise. The forward process is as follows:

$$q(z_t|z_0) = N(z_t; \sqrt{\bar{\delta}_t}z_0, (1-\bar{\delta}_t)I) \quad (22)$$

The corresponding posterior $q(z_{t-1}|z_t, z_0)$ is also Gaussian:

$$q(z_{t-1}|z_t, z_0) = N(z_{t-1}; \tilde{\mu}_t(z_t, z_0), \tilde{\gamma}_t I) \quad (23)$$

where

$$\tilde{\mu}_t(z_t, z_0) = \frac{\sqrt{\bar{\delta}_{t-1}}(1-\delta_t)}{1-\bar{\delta}_t}z_0 + \frac{\sqrt{\delta_t}(1-\bar{\delta}_{t-1})}{1-\bar{\delta}_t}z_t \quad (24)$$

and

$$\tilde{\gamma}_t = \frac{1-\bar{\delta}_{t-1}}{1-\bar{\delta}_t}\gamma_t, \quad (25)$$

For the conditional distribution, the reverse process involves learning a parameterized model $p_\theta(z_{t-1} \mid z_t)$ that approximates the reverse transitions. The parameterized model takes the following form:

$$p_\theta(z_{t-1} \mid z_t) = \mathcal{N}\big(z_{t-1};\, \mu_\theta(z_t, t),\, \Sigma_\theta(z_t, t)\big),$$

In our implementation, the neural network predicts the noise term used to form the reverse mean, while the covariance is not learned. We use the fixed diagonal covariance induced by the diffusion schedule, i.e., $\Sigma_\theta(z_t, t) = \tilde{\gamma}_t I$. By sampling from this learned reverse process starting from the classifier-informed endpoint $z_T \sim \mathcal{N}(g_\psi(\mathbf{w}), I)$, one can generate data resembling the original training data.

The reverse process is parameterized as:

$$p_\theta(z_{0:T}) = p(z_T)\prod_{t=1}^{T} p_\theta(z_{t-1} \mid z_t).$$

A common parameterization for the mean is given by:

$$\mu_\theta(z_t, t) = \frac{1}{\sqrt{\delta_t}}\left(z_t - \frac{1-\delta_t}{\sqrt{1-\bar{\delta}_t}}\,\epsilon_\theta(z_t, t)\right),$$

where $\epsilon_\theta(z_t, t)$ is a neural network predicting the noise component. Thus, the trainable part of the reverse transition is the denoising/noise-prediction network, not a separate covariance predictor. Similarly for the classification diffusion models, the corresponding tractable forward process posterior is

$$q(z_{t-1}|z_t, z_0, w) = q(z_{t-1}|z_t, z_0, g_\psi(w)) = \quad (26)$$

$$\mathcal{N}(z_{t-1}; \tilde{\mu}(z_t, z_0, g_\psi(w)), \tilde{\gamma}_t I) \quad (27)$$

$$\tilde{\mu} := \frac{\gamma_t\sqrt{\bar{\delta}_{t-1}}}{1-\bar{\delta}_t}z_0 + \frac{\sqrt{\delta_t}(1-\bar{\delta}_{t-1})}{1-\bar{\delta}_t}z_t + \left(1 + \frac{(\sqrt{\bar{\delta}_t}-1)\left(\sqrt{\delta_t}+\sqrt{\bar{\delta}_{t-1}}\right)}{1-\bar{\delta}_t}\right) g_\psi(w) \quad (28)$$

$$\tilde{\gamma}_t := \frac{1-\bar{\delta}_{t-1}}{1-\bar{\delta}_t}\gamma_t \quad (29)$$

We derive the mean and variance of the forward process posteriors $q(z_{t-1}|z_t, z_0, w)$, which are tractable when conditioned on z_0:

$$q(z_{t-1}|z_t, z_0, w) \propto q(z_t|z_{t-1}, g_\psi(w))q(z_{t-1}|z_0, g_\psi(w)) \propto \exp\left(-\frac{1}{2}\left(\frac{(z_t - (1-\sqrt{\delta_t})g_\psi(w) - \sqrt{\delta_t}z_{t-1})^2}{\gamma_t} + \frac{(z_{t-1} - \sqrt{\bar{\delta}_{t-1}}z_0 - (1-\sqrt{\bar{\delta}_{t-1}})g_\psi(w))^2}{1-\bar{\delta}_{t-1}}\right)\right) \quad (30)$$

This expands to:

$$\propto \exp\left(-\frac{1}{2}\left(\frac{\delta_t z_{t-1}^2 - 2\sqrt{\delta_t}z_t z_{t-1} + 2\sqrt{\delta_t}(1-\sqrt{\delta_t})g_\psi(w)z_{t-1}}{\gamma_t} + \frac{z_{t-1}^2 - 2\sqrt{\bar{\delta}_{t-1}}z_0 z_{t-1} - 2(1-\sqrt{\bar{\delta}_{t-1}})g_\psi(w)z_{t-1}}{1-\bar{\delta}_{t-1}}\right)\right) \quad (31)$$

This simplifies to:

$$\exp\left(-\frac{1}{2}\left(\left(\frac{\delta_t}{\gamma_t} + \frac{1}{1-\bar{\delta}_{t-1}}\right)z_{t-1}^2 - 2\left(\frac{\sqrt{\bar{\delta}_{t-1}}}{1-\bar{\delta}_{t-1}}z_0 + \frac{\sqrt{\delta_t}}{\gamma_t}z_t + \left(\frac{\sqrt{\delta_t}(\sqrt{\delta_t}-1)}{\gamma_t} + \frac{1-\sqrt{\bar{\delta}_{t-1}}}{1-\bar{\delta}_{t-1}}\right)g_\psi(w)\right)z_{t-1}\right)\right)$$

where

$$\frac{\delta_t(1-\bar{\delta}_{t-1})+\gamma_t}{\gamma_t(1-\bar{\delta}_{t-1})}=\frac{1-\bar{\delta}_t}{\gamma_t(1-\bar{\delta}_{t-1})} \tag{32}$$

and we have the posterior variance

$$\tilde{\gamma}_t=\frac{1-\bar{\delta}_{t-1}}{1-\bar{\delta}_t}\gamma_t \tag{33}$$

Meanwhile, we obtain the following coefficients of the terms in the posterior mean by multiplying each coefficient by the posterior variance $\gamma_t\frac{1-\bar{\delta}_{t-1}}{1-\bar{\delta}_t}$:

$$\lambda_0=\frac{\sqrt{\bar{\delta}_{t-1}}}{1-\bar{\delta}_{t-1}}\cdot\gamma_t\frac{1-\bar{\delta}_{t-1}}{1-\bar{\delta}_t}=\frac{\sqrt{\bar{\delta}_{t-1}}}{1-\bar{\delta}_t}\gamma_t \tag{34}$$

$$\lambda_1=\frac{\sqrt{\delta_t}}{\gamma_t}\cdot\gamma_t\frac{1-\bar{\delta}_{t-1}}{1-\bar{\delta}_t}=\frac{1-\bar{\delta}_{t-1}}{1-\bar{\delta}_t}\sqrt{\delta_t} \tag{35}$$

and

$$\lambda_2=\left(\frac{\sqrt{\delta_t}(\sqrt{\delta_t}-1)}{\gamma_t}+\frac{1-\sqrt{\bar{\delta}_{t-1}}}{1-\bar{\delta}_{t-1}}\right)\cdot\gamma_t\frac{1-\bar{\delta}_{t-1}}{1-\bar{\delta}_t} \tag{36}$$

$$=\frac{\delta_t-\bar{\delta}_t-\sqrt{\delta_t}(1-\bar{\delta}_{t-1})+\gamma_t-\gamma_t\sqrt{\bar{\delta}_{t-1}}}{1-\bar{\delta}_t} \tag{37}$$

$$=1+\frac{(\sqrt{\bar{\delta}_t}-1)\left(\sqrt{\delta_t}+\sqrt{\bar{\delta}_{t-1}}\right)}{1-\bar{\delta}_t} \tag{38}$$

The coefficients λ_0, λ_1, and λ_2 respectively weight the clean label, the current noisy label, and the guidance vector when reconstructing the previous diffusion state. Together, they give the posterior mean

$$\tilde{\mu}(z_t,z_0,g_\psi(w))=\lambda_0 z_0+\lambda_1 z_t+\lambda_2 g_\psi(w) \tag{39}$$

IV. Performance Evaluation

A. Dataset and Implementation

The first dataset is the Kaggle Accident Detection From CCTV Footage dataset [31]. It contains CCTV video frames labeled as Accident or Non-Accident and is distributed in train, validation, and test folders. The copy used in this study contains 1,049 images, comprising 480 Accident and 569 Non-Accident images. We used 128×128 inputs and a class-wise 80-10-10 split, giving 851 training images, 98 validation images, and 100 test images. We additionally trained ImageNet-initialized GoogLeNet, ResNet18, ResNet50, ConvNeXt, and ViT classifiers under the same image size and split.

We also prepared the BeamNG drive dataset [32]. It consists of frames gathered from BeamNG.drive YouTube videos and contains 13,228 images. For this dataset, the same class-wise 80-10-10 split with seed 2000 yields 10,582 training images, 1,322 validation images, and 1,324 test images.

All images were resized to 128×128. Training images used center cropping, random horizontal flipping, and random rotation; validation and test images used deterministic preprocessing only. All baseline classifiers were trained with the same 128×128 input resolution and the same train/validation/test split. The EfficientNet guidance classifier was initialized from ImageNet weights, with its input and

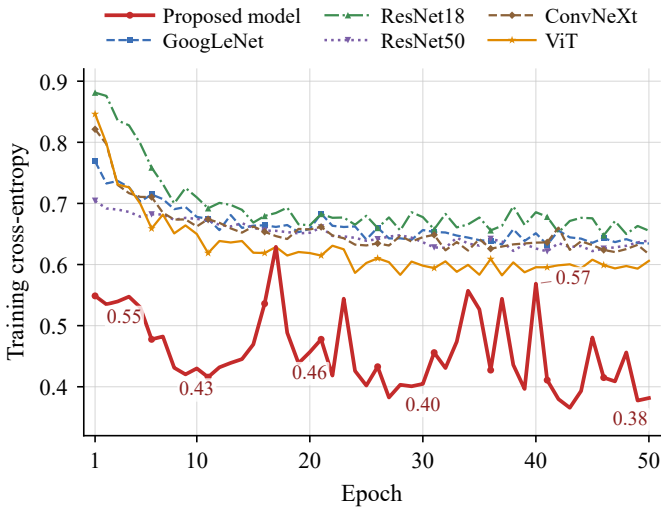


Figure 3: Training cross-entropy on the Accident Detection From CCTV Footage dataset.

classification layers fine-tuned for the two accident classes. For the hyperparameters of the proposed model, we set the number of diffusion timesteps to 10. This choice is driven by inference cost rather than by the timestep study of Table V: the reverse process performs one full evaluation of the noise-prediction network per timestep, so sampling cost grows linearly with the timestep count. This matters for transportation deployments, where inference typically runs under tight compute budgets [33]. That work discusses transportation models deployed on embedded hardware under real-time limits, where a ten-step sampler needs a third of the network evaluations of a thirty-step one. For the diffusion noise schedule, the endpoint parameters β_1 and β_T were set to 10^{-4} and 0.02, respectively; these are schedule endpoints and not the Adam optimizer momentum parameters. A cosine noise schedule and a learnable linear timestep embedding were also used, matching the Lin-Cos-Lin configuration of Table IV. For optimization, Adam used learning rate 10^{-3}, $\beta_1=0.9$, $\epsilon=10^{-8}$, zero weight decay, learning-rate scheduling, and gradient clipping at 1.0. The reported configuration used 10 diffusion timesteps, a cosine noise schedule, a learnable linear timestep embedding, and a 50-epoch training budget. The compared classifiers used ImageNet initialization, replaced their final classification layer for two classes, and were trained with Adam at learning rate 10^{-3} under the same split and resolution. The standalone EfficientNet guidance classifier is not reported as a separate row, so the present comparison does not isolate the contribution of the conditional diffusion stage from that of the guidance network. We conducted a comprehensive evaluation of our model by comparing the training and testing performance across several metrics, including accuracy, precision, recall, F1-score, Mean Squared Error (MSE), and Cross Entropy.

The performance of the proposed model and five baselines is shown in Table III. The proposed classification diffusion model achieved an accuracy of 97.32%. These results indicate that the proposed diffusion-guided classifier

Model	Accuracy (%)	Precision (%)	Recall (%)	F1 Score (%)	Cross-Entropy	MSE
Proposed diffusion	**97.32**	**99.40**	**97.32**	**98.14**	**0.353**	**0.064**
GoogLeNet	96.00	96.00	96.00	96.00	0.106	0.031
ResNet18	96.00	96.00	96.00	96.00	0.125	0.032
ResNet50	96.00	96.28	96.00	95.98	0.104	0.028
ConvNeXt	53.00	28.09	53.00	36.72	0.692	0.249
ViT	61.00	61.34	61.00	61.02	0.673	0.240

Table III: Test-set performance metrics on the Accident Detection From CCTV Footage dataset.

Model	Accuracy (%)	Precision (%)	Recall (%)	F1 Score (%)	Cross-Entropy	MSE
Lin-Lin-Lin	91.07	99.40	91.10	94.70	0.387	0.113
Lin-Lin-Sin	90.18	99.60	90.20	94.40	0.362	0.081
Lin-Cos-Lin	**97.32**	**99.40**	**97.32**	**98.14**	0.353	**0.064**
Lin-Cos-Sin	96.43	99.40	96.40	97.70	0.352	0.079
Att-Lin-Lin	92.86	99.30	92.90	95.30	0.384	0.082
Att-Lin-Sin	87.50	99.30	87.50	92.60	0.623	0.877
Att-Cos-Lin	95.54	99.40	95.50	97.20	**0.348**	0.086
Att-Cos-Sin	88.39	99.60	88.40	93.30	1.097	19.469

Table IV: Ablation configurations, named architecture-schedule-embedding (Lin = linear, Att = attention, Cos = cosine, Sin = sinusoidal).

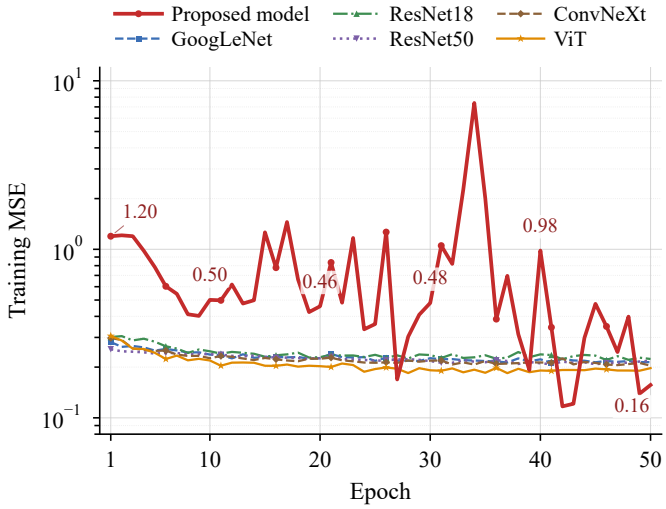


Figure 4: Training MSE on the Accident Detection From CCTV Footage dataset (log axis).

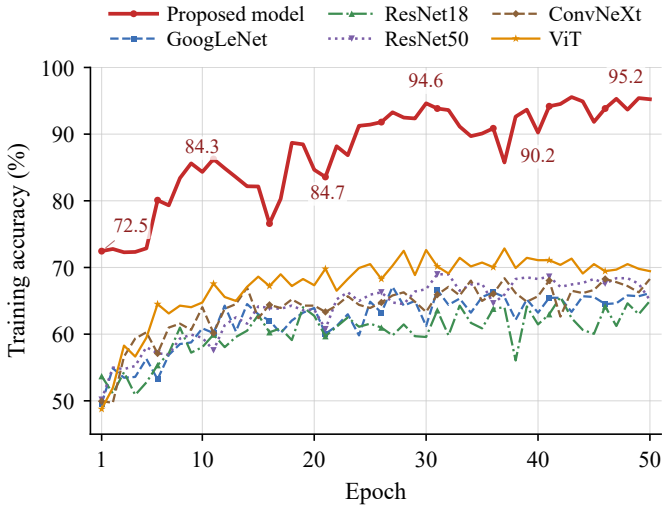


Figure 5: Training accuracy on the Accident Detection From CCTV Footage dataset.

remains competitive against both standard CNN and modern transformer/ConvNeXt-style classifiers under the same image size and dataset split. ConvNeXt's scores are exactly those of a classifier that labels every test image Non-Accident (the majority class), and ViT learns only a weak rule; we attribute this to the shared protocol (128×128 inputs, 851 training images, and a learning rate of 10^{-3}, well above the usual fine-tuning rate for these models) rather than to the architectures. All rows are evaluated on the same 100-image test split, but the diffusion metrics are averaged over test mini-batches whereas the baseline metrics are pooled over the full split, so the two are not exact like-for-like aggregates. Accuracy, however, converts exactly: with six test mini-batches of 16 and one of 4, the only outcome consistent with 97.32% is 97 of 100 images correct, a pooled accuracy of 97.00% against 96.00% for the best CNN baselines. Precision and F1-score cannot be converted without per-batch class counts.

The training performance of the five baseline classifiers was evaluated over 50 epochs, as depicted in Figure 5. The CNN-based models converged to high accuracy on the Accident Detection From CCTV Footage dataset, while ConvNeXt and ViT showed weaker convergence under the same 128×128 image setting and training budget. Training cross-entropy and MSE for the same runs appear in Figs. 3 and 4, with the corresponding curves for the BeamNG drive dataset in Figs. 6 to 8. Across both datasets the proposed model separates from every baseline within the first ten epochs, and the MSE curves use a logarithmic axis because the diffusion objective produces transient spikes that a linear axis would clip. Figure 2 provides a corresponding model-level

Timesteps	Accuracy (%)	Precision (%)	Recall (%)	F1 Score (%)	Cross-Entropy	MSE
10	88.39	99.60	88.40	93.30	1.097	19.469
20	95.54	99.30	95.50	97.10	0.378	0.062
30	95.54	99.30	95.50	96.70	0.347	0.051

Table V: Performance metrics for different diffusion timestep counts, for the Att-Cos-Sin variant.

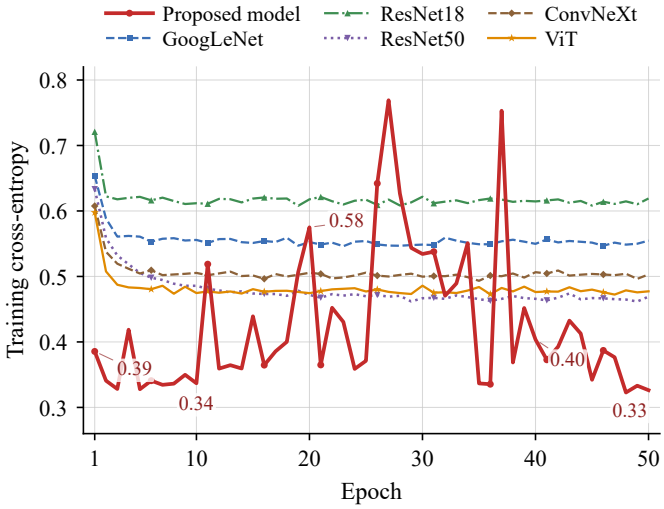


Figure 6: Training cross-entropy on the BeamNG drive dataset.

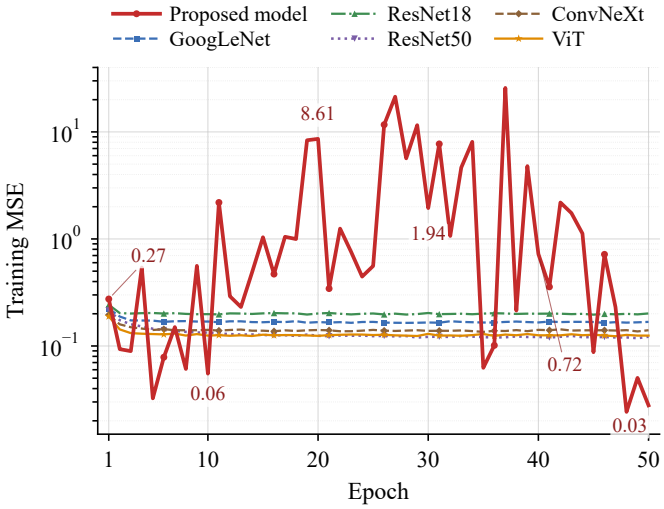


Figure 7: Training MSE on the BeamNG drive dataset (log axis).

interpretation: it identifies the image-derived guidance vector supplied by EfficientNet and shows where the conditional diffusion stage produces the final class decision. The ablation results identify configurations that perform well under the controlled split. In practice, the architecture, noise schedule, and timestep embedding influence how visual guidance is incorporated during denoising and therefore provide clear design levers for scene quality and variation. The same controlled protocol can be extended to missing information, weather, illumination, viewpoint changes, and occlusion.

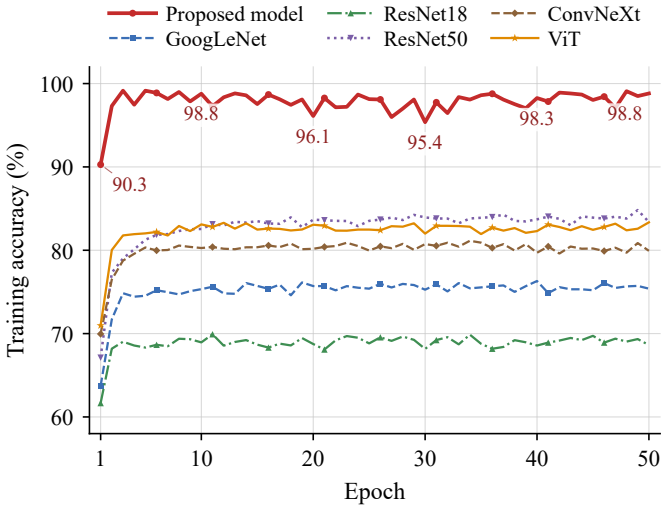


Figure 8: Training accuracy on the BeamNG drive dataset.

B. Ablation Studies

In our experiments, we systematically varied key parameters of the proposed model, as described below.

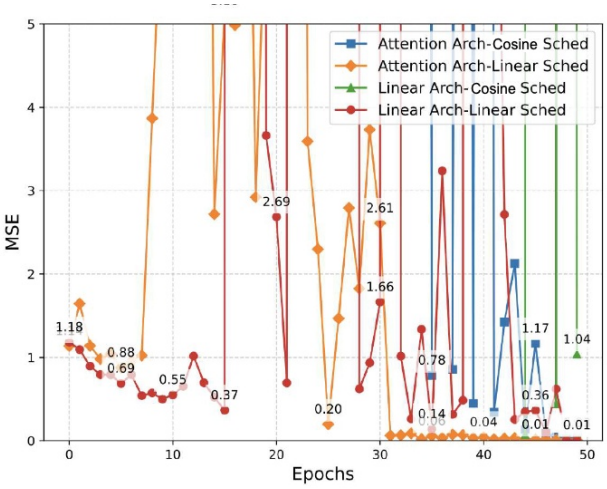
1) Timestep Embedding: The choice between sinusoidal and linear embeddings affects how the model encodes temporal information. We used linear and sinusoidal embeddings.

2) Network Architecture: The network architecture determines how the encoder encodes the image data. We varied between sequential linear layers and the multi-head attention framework.

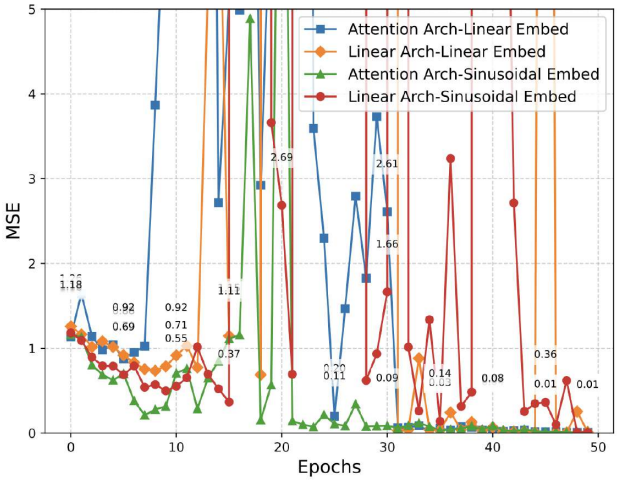
3) Scheduling Method: The scheduling method refers to the beta schedule, which determines the amount of noise added across time steps in the diffusion process. The cosine-beta schedule strikes an optimal balance by avoiding insufficient noise addition in the early stages and excessive noise in the final stages while allowing a substantial increase in noise during the middle phase. This ensures that samples at any timestep are equally valuable to the training process, while a linear schedule adds noise at a steady, uniform rate throughout the process.

4) Timestep Count: This is the number of diffusion timesteps used in training and sampling.

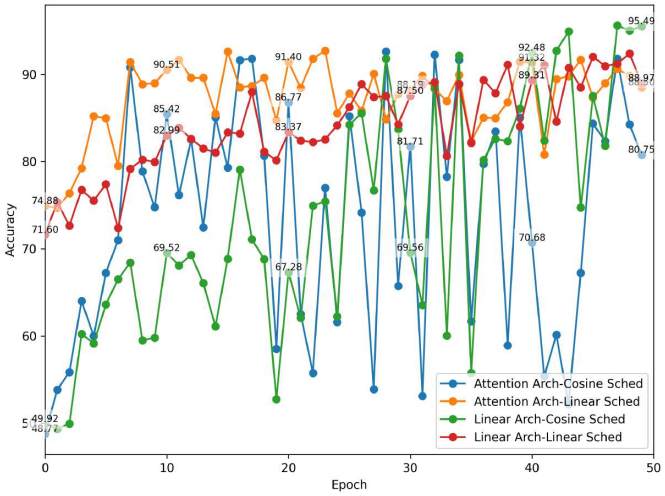
The final selected configuration in Table IV is Lin-Cos-Lin, which uses a linear architecture, cosine noise schedule, and learnable linear timestep embedding. We also evaluated attention-based variants as part of the ablation study, but the attention architecture was not selected as the final configuration. For the attention-based ablation variants, the model first divides the input image into patches via a convolutional layer, with the kernel size set equal to the



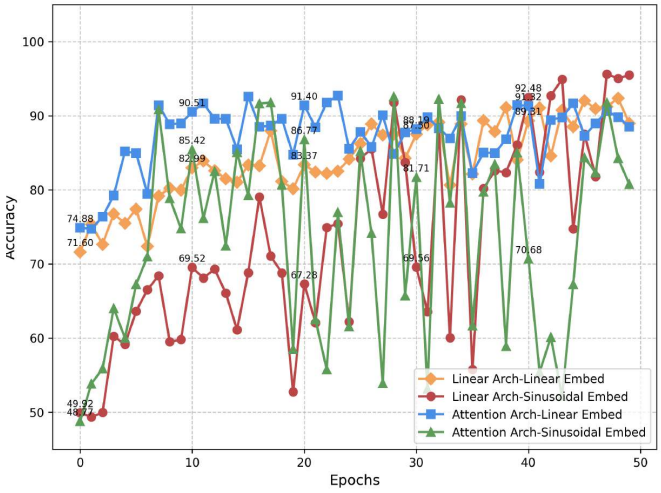
(a) MSE-1



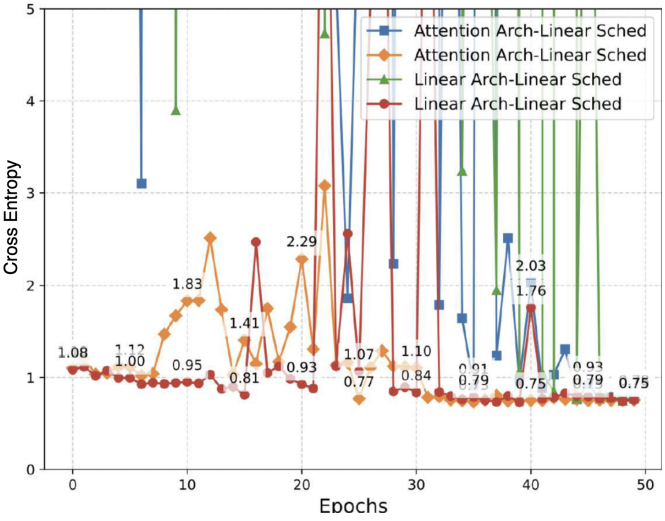
(b) MSE-2



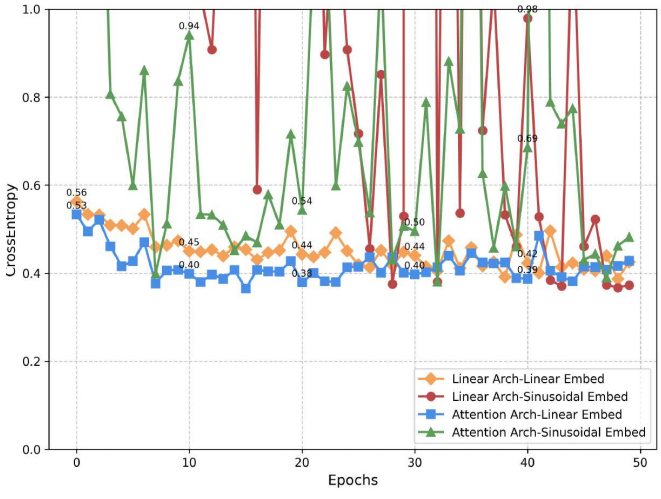
(c) Accuracy-1



(d) Accuracy-2

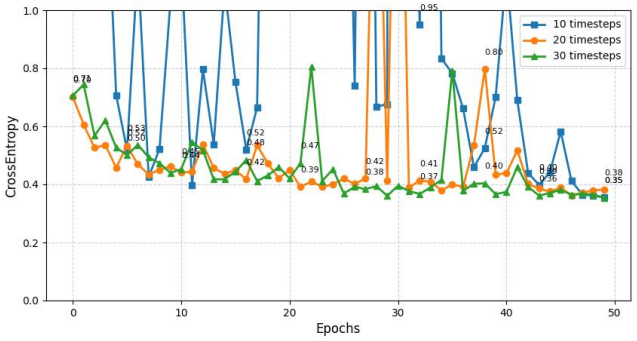


(e) Cross Entropy-1

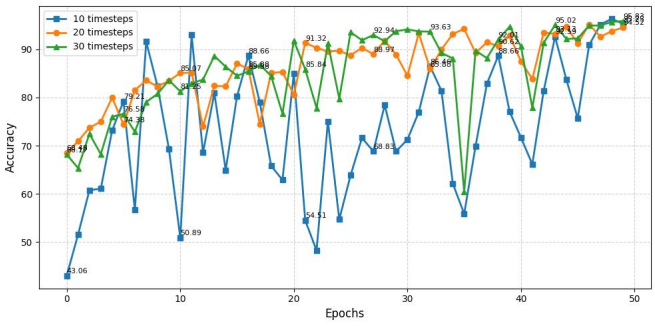


(f) Cross Entropy-2

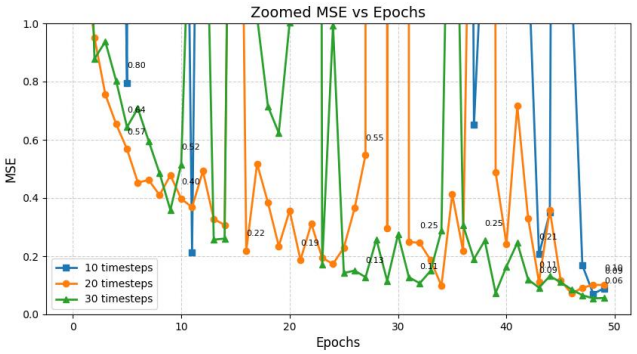
Figure 9: Ablation metrics. Column 1 varies the scheduler, column 2 varies the timestep embedding.



(a) Cross Entropy over Timesteps



(b) Accuracy over Timesteps



(c) Mean Squared Error (MSE) over Timesteps

Figure 10: Training metric trends for different diffusion timestep counts.

desired patch size. These patches are then flattened and treated as tokens. A multi-head attention block processes these tokens, enabling the model to capture global dependencies and contextual relationships across the image. The attention outputs are aggregated using a token combiner (which includes layer normalization and a linear transformation) to generate a robust feature representation.

Table IV presents accuracy, cross-entropy loss, MSE, precision, recall and F1-score across all the model variations based on the ablation parameters. These variations are as follows: varying time embeddings (linear and sinusoidal) with linear architecture and a linear time schedule; varying time embeddings (linear and sinusoidal) with linear architecture and a cosine time schedule; varying time embeddings (linear and sinusoidal) with attention-based architecture and a linear time schedule, and varying time embeddings (linear and sinusoidal) with attention-based architecture and a cosine time schedule. The Lin-Cos-Lin configuration achieves the highest accuracy (97.32%) and F1-score (98.14%), making it the strongest performer overall. Other models like "Lin-Cos-Sin" and "Att-Cos-Lin" perform competitively in some metrics. Figure 9 plots the model variations in terms of training MSE, accuracy, and cross-entropy. In column 1 of Figure 9 (i.e., Figures 9a, 9c, and 9e), the type of scheduler is varied: linear or cosine. In column 2 (Figures 9b, 9d, and 9f), the type of timestep embedding is varied: linear or sinusoidal.

The accuracy (Figure 9d) and cross-entropy (Figure 9f) panels show that the linear embedding learns faster across epochs and gives higher accuracy values early on compared to the sinusoidal embedding. Across the graphs of all metrics, the sinusoidal embedding shows the highest variation across epochs for training accuracy. Figure 9a shows that the Mean Squared Error (MSE) converges fastest for the attention architecture under the linear noise schedule. The runs employing the linear schedule give higher accuracy values initially and show less variation overall, whereas the cosine-schedule runs show higher variation in accuracy. We experimented with three configurations: 10, 20, and 30 timesteps, the results of which are listed in Table V. This study was conducted on the attention-based variant (Att-Cos-Sin) rather than on the selected Lin-Cos-Lin model, so it characterizes how the attention architecture responds to the timestep count and does not bear on the accuracy reported in Table III. Within that variant, the test accuracy is notably higher at 20 and 30 timesteps (both 95.54%). Further, the model with 30 timesteps yields lower cross-entropy (0.347) and MSE (0.051) and hence gives the best overall loss metrics. We nevertheless retain 10 timesteps for the reported model, since sampling cost grows linearly with the timestep count.

The training metrics, namely accuracy, cross-entropy, and mean squared error (MSE), are illustrated in Figure 10. The plots indicate that 20 and 30 timesteps generally yield better convergence in both accuracy and loss metrics, with 30 timesteps often giving the lowest MSE. The 10 timesteps plots show a higher variation in all metrics across the epochs. These figures provide insights into how the model's learning behavior varies with the number of time steps.

V. Conclusions and Future Work

In this work, we propose a diffusion-guided hybrid classifier for accident detection in ITS, which overcomes the single deterministic decision boundary of conventional discriminative classifiers, which cannot refine an uncertain prediction. To supply the image-conditioned prior, we fine-tune an EfficientNet guidance classifier whose soft output $g_\psi(\mathbf{w})$ centers the endpoint of the forward process, so that denoising begins from a classifier-informed prior rather than from an unconditional standard Gaussian. We modulate every conditional module with a timestep embedding and the image feature embedding, so that the reverse process refines that prior instead of discarding it.

The experimental results show the efficacy of the proposed model: it reaches 97.32% accuracy (97.00% pooled) and

98.14% F1-score on the Accident Detection From CCTV Footage dataset, against 96.00% for the strongest CNN baselines, with the cosine schedule and learnable linear embedding selected by ablation. On the BeamNG drive dataset, it reaches 98.81% training accuracy, against 83.39% for ResNet50. The model does not lead on every metric, since its cross-entropy of 0.353 and MSE of 0.064 remain above the 0.104 and 0.028 of ResNet50.

These results rest on a binary Accident/Non-Accident formulation evaluated at 128×128 resolution under a single fixed split. As part of our future work, we intend to extend the formulation to multi-class event categories and to evaluate it on scenes with wider weather, illumination, and viewpoint variation, and to report runtime, memory, and parameter-count comparisons against the baselines. We also intend to compute the guidance vector over consecutive frames, so the sampler can exploit motion evidence that a still frame does not carry.

REFERENCES

[1] R. A. Khalil, Z. Safelnasr, N. Yemane, M. Kedir, A. Shafiqurrahman, and N. Saeed, "Advanced learning technologies for intelligent transportation systems: Prospects and challenges," *IEEE Open Journal of Vehicular Technology*, vol. 5, pp. 397–427, 2024.

[2] Y. Yu, M. Xu, and J. Gu, "Vision-based traffic accident detection using sparse spatio-temporal features and weighted extreme learning machine," *IET Intelligent Transport Systems*, vol. 13, no. 9, pp. 1417–1428, 2019.

[3] S. Masri, H. I. Ashqar, and M. Elhenawy, "Leveraging large language models (llms) for traffic management at urban intersections," in *2025 Sixteenth International Conference on Ubiquitous and Future Networks (ICUFN)*. IEEE, 2025, pp. 347–351.

[4] S. Bakheet and A. Al-Hamadi, "A deep neural framework for real-time vehicular accident detection based on motion temporal templates," *Heliyon*, vol. 8, no. 11, 2022.

[5] H. Hozhabr Pour, F. Li, L. Wegmeth, C. Trense, R. Doniec, M. Grzegorzek, and R. Wismüller, "A machine learning framework for automated accident detection based on multimodal sensors in cars," *Sensors*, vol. 22, no. 10, p. 3634, 2022.

[6] H. Rathore, S. Sai, and A. Gundewar, "Social psychology inspired distributed ledger technique for anomaly detection in connected vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 7, pp. 7092–7107, 2023.

[7] S. Jiang and Y. Feng, "Small parallel residual convolutional neural network and traffic congestion detection," *Scientific Reports*, vol. 15, no. 1, p. 14285, 2025.

[8] A. N. M. Nusari, I. Y. Ozbek, and E. A. Oral, "Automatic vehicle accident detection and classification from images: A comparison of yolov9 and yolo-nas algorithms," in *2024 32nd signal processing and communications applications conference (SIU)*. IEEE, 2024, pp. 1–4.

[9] U. Jartarghar, D. Sanghvi, M. Nidgundi, K. Kumar, and S. Varur, "Vision transformer-based multi-phase accident detection and false positive mitigation," 2024.

[10] J. Fang, L.-l. Li, J. Zhou, J. Xiao, H. Yu, C. Lv, J. Xue, and T.-S. Chua, "Abductive ego-view accident video understanding for safe driving perception," in *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2024, pp. 22 030–22 040.

[11] Z. Xi, X. Liu, Y. Liu, Y. Cai, and Y. Zheng, "Integrating generative adversarial networks and convolutional neural networks for enhanced traffic accidents detection and analysis," *IEEE Transactions on Intelligent Transportation Systems*, 2025.

[12] H. Wang, D. J. Miller, and G. Kesidis, "Anomaly detection of adversarial examples using class-conditional generative adversarial networks," *Computers & Security*, vol. 124, p. 102956, 2023.

[13] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," *Advances in neural information processing systems*, vol. 33, pp. 6840–6851, 2020.

[14] Y. Song, J. Sohl-Dickstein, D. P. Kingma, A. Kumar, S. Ermon, and B. Poole, "Score-based generative modeling through stochastic differential equations," *arXiv preprint arXiv:2011.13456*, 2020.

[15] Y. Zhu, K. Zhang, J. Liang, J. Cao, B. Wen, R. Timofte, and L. Van Gool, "Denoising diffusion models for plug-and-play image restoration," in *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. IEEE, 2023, pp. 1219–1229.

[16] Z. Xiao, K. Kreis, and A. Vahdat, "Tackling the generative learning trilemma with denoising diffusion GANs," *arXiv preprint arXiv:2112.07804*, 2021.

[17] P. Dhariwal and A. Nichol, "Diffusion models beat GANs on image synthesis," *Advances in neural information processing systems*, vol. 34, pp. 8780–8794, 2021.

[18] Y. Song and S. Ermon, "Generative modeling by estimating gradients of the data distribution," *Advances in neural information processing systems*, vol. 32, 2019.

[19] S. Sai, U. Mittal, and V. Chamola, "Dmdat: Diffusion model-based data augmentation technique for vision-based accident detection in vehicular networks," *IEEE Transactions on Vehicular Technology*, vol. 74, no. 2, pp. 2241–2250, 2025.

[20] S. Sai, K. Shah, and V. Chamola, "A novel diffusion-guided semantic data augmentation method for vehicular iot," *IEEE Internet of Things Magazine*, vol. 8, no. 6, pp. 58–64, 2025.

[21] D. Lei, M. Xu, and S. Wang, "A conditional diffusion model for probabilistic estimation of traffic states at sensor-free locations," *Transportation Research Part C: Emerging Technologies*, vol. 166, p. 104798, 2024.

[22] J. Wu, X. Piao, X. Wei, and Y. Zhang, "Stcdm: Spatio-temporal conditional diffusion model for traffic data filling," *IET Intelligent Transport Systems*, vol. 19, no. 1, p. e70016, 2025.

[23] J. You, H. Shi, K. Wu, K. Long, S. Fu, S. Chen, and B. Ran, "Crossfusor: A cross-attention transformer enhanced conditional diffusion model for car-following trajectory prediction," *arXiv preprint arXiv:2406.11941*, 2024.

[24] H. Chai, T. Li, F. Jiang, S. Zhang, and Y. Li, "Knowledge guided conditional diffusion model for controllable mobile traffic generation," in *Companion Proceedings of the ACM Web Conference 2024*, 2024, pp. 851–854.

[25] B. Lu, Q. Miao, Y. Liu, T. S. Tamir, H. Zhao, X. Zhang, Y. Lv, and F.-Y. Wang, "A diffusion model for traffic data imputation," *IEEE/CAA Journal of Automatica Sinica*, vol. 12, no. 3, pp. 606–617, 2025.

[26] E. Hoogeboom, D. Nielsen, P. Jaini, P. Forré, and M. Welling, "Argmax flows and multinomial diffusion: Learning categorical distributions," *Advances in neural information processing systems*, vol. 34, pp. 12 454–12 465, 2021.

[27] J. Austin, D. D. Johnson, J. Ho, D. Tarlow, and R. Van Den Berg, "Structured denoising diffusion models in discrete state-spaces," *Advances in neural information processing systems*, vol. 34, pp. 17 981–17 993, 2021.

[28] Y. Yang, H. Fu, A. I. Aviles-Rivero, C.-B. Schönlieb, and L. Zhu, "DiffMIC: Dual-guidance diffusion network for medical image classification," in *International conference on medical image computing and computer-assisted intervention*. Springer, 2023, pp. 95–105.

[29] X. Han, H. Zheng, and M. Zhou, "CARD: Classification and regression diffusion models," *Advances in Neural Information Processing Systems*, vol. 35, pp. 18 100–18 115, 2022.

[30] K. Pandey, A. Mukherjee, P. Rai, and A. Kumar, "Diffusevae: Efficient, controllable and high-fidelity generation from low-dimensional latents," *arXiv preprint arXiv:2201.00308*, 2022.

[31] C. Kumar, "Accident detection from cctv footage," 2020, [Online; accessed 2026-08-21]. [Online]. Available: https://www.kaggle.com/datasets/ckay16/accident-detection-from-cctv-footage

[32] M. Tahir, "Synthetic dataset for accident detection," 2022, [Online; accessed 2026-08-21]. [Online]. Available: https://www.kaggle.com/datasets/mehwishtahir722/synthetic-dataset-for-accident-detection

[33] S. Sai, K. Shah, G. Jain, and V. Chamola, "Tinyml for transportation systems: Enabling smart mobility," *IEEE Consumer Electronics Magazine*, pp. 1–10, 2026.