Dynamic Transformer-Based Multi-Modal Sensor Fusion for Collision Prediction

Our framework ensures high driving safety through transformer-based multi-modal sensor fusion for timely (3 seconds before impact) collision prediction, while minimizing the computational latency of deep neural network processing through dynamic inference. The system consists of three integral components: (1) a synchronized sensor data pipeline, (2) a perceptual backbone, and (3) a binary prediction head. The dynamic inference used in the perceptual backbone draws upon recent developments in layer-skipping transformers, particularly the oracle-based dynamic computation proposed by Mersha [1].

A. Data Pipeline

Our experiments are conducted in the CARLA (version 0.9.15) simulation environment, where an ego vehicle navigates multi-lane highways populated with 80 other vehicles. The simulation is designed to reflect real-world fast-paced driving scenarios, including high traffic density and potential collision situations. The ego vehicle is equipped with the following sensors operating at ~30 Hz:

- A front-facing monocular RGB camera (800×600 resolution, 90° FOV) that captures high-definition visual context for object recognition and semantic understanding.
- A 50 m range LiDAR that provides dense 3D point cloud data representing the spatial layout of the environment.
- An Inertial Measurement Unit (IMU) that captures 3D accelerations and angular velocities, offering insight into the vehicle's motion dynamics.
- A collision sensor that detects physical contact with other objects and is used to generate binary ground-truth labels for training the framework.

To create a temporally synchronized dataset, sensor data is logged and timestamped on a per-frame basis. The collected multi-sensor data is then preprocessed before being used by the framework. Each image captured by the camera is resized to 224×224 and normalized using ImageNet statistics. The 3D point clouds from LiDAR are projected into 2D range images via spherical projection, wherein points are mapped to an elevation–azimuth grid based on their angles and distances. This projection preserves the point cloud's spatial structure in a compact 2D representation while drastically reducing data dimensionality. An aligned-validity channel is also produced, which marks pixels receiving at least one return, for each range image, to help the framework distinguish between empty bins and near-zero ranges. The accelerometer and gyroscope readings from the IMU sensor are concatenated into a 6-dimensional vector per frame and normalized using global dataset statistics calculated during training. If a modality is missing

for a frame, we write a zero-filled tensor of the proper shape and record a per-frame modality availability mask $a_{image}, a_{lidar}, a_{imu} \in \{0,1\}$ for downstream handling. All sensor modalities and the modality mask are aligned by timestamp and packaged into a single synchronized dataset of frames that capture the real-time, multi-modal state of the vehicle at each moment.

B. Perceptual Backbone Architecture

The perception network of our framework is composed of modality-specific embedding layers and a dynamic transformer encoder with input-adaptive computation. The use of a transformer enables global attention-based fusion, allowing the model to capture long-range dependencies between modalities that convolution or geometry-based fusion might miss. This architecture is designed to balance real-time processing constraints with the need for deep computation of dynamic driving environments.

1) Modality-Specific Embedding

Each input modality is passed through a dedicated network to convert each processed sensor data into a fixed-dimensional feature representation. This stage ensures that each modality is processed in a way that maximally preserves relevant spatial or temporal information.

- **Camera Embedding**: A pre-trained ResNet-18 backbone (truncated at the penultimate layer) extracts hierarchical visual features from the image input. This is followed by a linear projection to embed the features into a 384-dimensional vector called e_{image}.
- **LiDAR Embedding**: We encode the single-channel LiDAR range image using a ResNet-18 variant that accepts one input channel and a validity channel. The ResNet output is global-pooled and linearly projected into the 384-dimensional embedding space as e_{lidar}.
- **IMU Embedding**: The IMU stream passes through a shallow two-layer MLP that embeds the vehicle's motion dynamics. We use a hidden layer with ReLU activation to allow the network to learn useful combinations of the inertial readings. The output is then concatenated and projected into the common embedding space as e_{imu}.
- **Availability Embedding**: We form an availability vector $a = [a_{image}, a_{lidar}, a_{imu}, \Sigma a_m]$ from the modality availability mask and embed it via a learned linear layer into the 384-dimension space as e_{avail}.

For each modality $m \in \{image, lidar, imu\}$, the per-frame token is decided by

$$t_m = e_m, \text{ when } a_m = 1$$

$$t_m = e_m^{miss}, \text{ when } a_m = 0 \text{ and } e_m^{prev} \text{ is not valid}$$

$$t_m = \beta e_m^{prev} + (1 - \beta) e_m^{miss}, \text{ when } a_m = 0 \text{ and } e_m^{prev} \text{ is valid}$$

where e_m^{miss} is a learned 384-dimensional parameter vector, e_m^{prev} is the most recent valid embedding carried from the previous frame (if any), and $\beta \in [0,1]$ is a decay parameter (default 0.5). This carry-forward imputation smooths short gaps without fabricating content.

The resulting tokens from each modality, the e_{avail}, and a learnable classification or [CLS] token, are concatenated to form the token sequence of a single frame. This [CLS] token is a trainable embedding that does not correspond to any specific sensor input. Instead, it interacts with all modality features through attention and is expected to absorb the fused multi-modal information for that frame (much like the [CLS] token in BERT or Vision Transformers represents an entire input sequence). We also add positional encodings to each token to retain temporal alignment across frames. The frames are ordered and concatenated to form the input token sequence, which is then passed to a stack of L dynamic transformer layers.

2) Dynamic Transformer Encoder

The dynamic transformer encoder integrates or fuses information across modalities and time. Through multi-head self-attention at each transformer layer, tokens can attend to one another and achieve sensor fusion as well as temporal integration (tokens can attend to those from preceding frames). These layers incorporate input-adaptive computation by skipping or executing layers on a per-frame basis, based on the complexity of the [CLS] token and its skip history. Inspired by Mersha's work [1], we compute the per-frame decision to either execute or skip each transformer layer T_j by using an oracle network q_j^* at every layer. The oracle network is a small two-layer MLP (with a sigmoid output) that acts as a gating function and takes three inputs:

- The current [CLS] token c_j from layer T_j
- A skip history vector, h_j, composed of the skip scores from all preceding layers (s_1, ..., s_{j-1})
- The availability embedding e_{avail}

The oracle network outputs a skip probability, $s_j \in [0, 1]$, using a sigmoid activation:

$$s_j = \sigma(q_j^*(c_j, h_j, e_{avail}))$$

If $s_j < 0.5$, the frame tokens are passed through Tj. Otherwise, the frame tokens bypass the layer, and the output from the previous layer is forwarded unchanged. This dynamic routing allows each frame in the input sequence to follow a distinct computation path, tailored to its content complexity and availability of modalities. To handle selective execution, we split the input sequence into two subsets at each layer:

- x_{exec}: frame tokens with $s_j < 0.5$, processed by T_j
- x_{skip}: frame tokens with $s_j \geq 0.5$, which retain their current representations

After processing, the outputs from both subsets are merged back into the original input sequence, maintaining the original ordering.

For any modality token with $a_m=0$, we add a large negative bias to attention logits involving that token as a key, effectively discouraging other tokens from attending to imputed content while still allowing the model to attend from that token to others. This reduces leakage from missing sensors into fused representations.

This adaptive depth transformer framework allows the model to allocate greater depth when critical modalities are present during complex or high-risk scenes (e.g., heavy traffic) while reducing computation on simple inputs (e.g., open roads) or incomplete modalities. We emphasize that all frames still propagate through the transformer simultaneously. The skipping mechanism only affects whether a given frame's tokens participate in a layer's attention and feed-forward network operations. Frames that skip a layer essentially carry their previous representation forward unchanged, but can still influence (and be influenced by) other frames in subsequent layers that execute. In this manner, even if a frame skips some intermediate layers, it will rejoin and interact with the sequence later, maintaining temporal context. The final output of the transformer consists of [CLS] tokens for each frame in the input sequence. Each [CLS] token encodes a fused representation of the vehicle's state and environment at that specific frame, which are then used by the binary prediction head to anticipate the probability of any imminent collision.

C. Binary Prediction Head

The binary prediction head is the final component of the framework, operating on the per-frame [CLS] token output from the dynamic transformer encoder. Each [CLS] token is passed through a lightweight feedforward neural network consisting of two fully connected layers with ReLU activation, followed by a final sigmoid-activated output layer. The architecture is as follows: FC(384→128) + ReLU → FC(128→1) + Sigmoid, where the sigmoid output yields a value in [0,1] interpreted as P(collision within 3s ∣ current frame). We deliberately keep the prediction head shallow and fast because the heavy multi-modal feature processing has already been done by the backbone. The sigmoid output represents the probability that a collision will occur within 3 seconds from the current frame. During training, this output is supervised using ground-truth binary labels generated from the collision sensor data. By transforming the fused scene representation into a binary risk prediction, the head enables proactive and early warning in high-speed driving environments.

D. Training and Testing

W.I.P

References

[1] A. Mersha, “Dynamic Transformer Networks,” in *Proc. Workshop on Dynamic Neural Networks at the 39th Int. Conf. on Machine Learning (ICML)*, Baltimore, MD, USA, Jul. 2022.