A Data Exploration

A.1 Synthetic Image Generation via Stable Diffusion

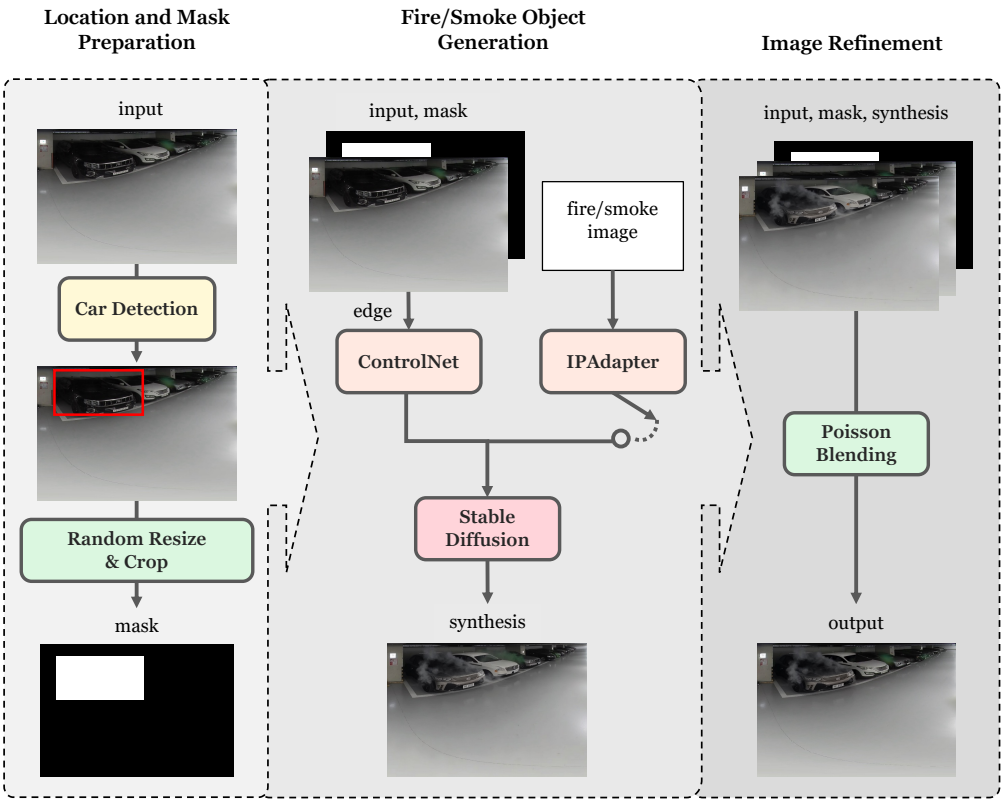

Figure 3: Fire/Smoke Dataset Generation Workflow for Underground Parking Scenarios. The process breaks down to three stages: **Location and mask preparation, fire/smoke object generation, and image refinement.**

Figure 3 depicts our workflow for generating fire/smoke object datasets, tailored to create realistic synthetic data for car fire scenarios in underground parking zones. The pipeline comprises three key stages: (1) detecting vehicles within parking zones and identifying the calamity's location, (2) generating images of the target vehicle engulfed in smoke or fire using a controllable diffusion model, and (3) applying blending-based post-processing to maintain original visual reality.

In the first stage, a fire/smoke-free input image serves as the base image. A pretrained YOLOv5 (Redmon, 2016) model detects objects of interest, such as vehicles. Subsequently, random resizing and padding operations are applied to generate a mask that specifies irregular regions for synthesizing fire or smoke effects.

In the second stage, a controllable diffusion model generates synthetic fire or smoke images. Canny edges (Canny, 1986) are extracted from the input image and fed into ControlNet (Zhang et al., 2023) to ensure structural alignment and create realistic fire- or smoke-engulfed scenes. The edge condition applies only during the first two-thirds of the diffusion process to preserve the vehicle's structure, while the final steps allow flexibility in shaping fire and smoke. For cases requiring specific fire styles or flame patterns, For specific fire styles or flame patterns, an optional IPAdapter (Ye et al., 2023) customizes the visual characteristics. Synthesis uses inpainting Stable Diffusion (SDInpaint) (Rombach et al., 2022) to generate fire or smoke effects within the designated mask.

Finally, in the image refinement stage, poisson blending(Pérez et al., 2023) seamlessly merges synthetic and original images, eliminating artifacts and ensuring a realistic final output.

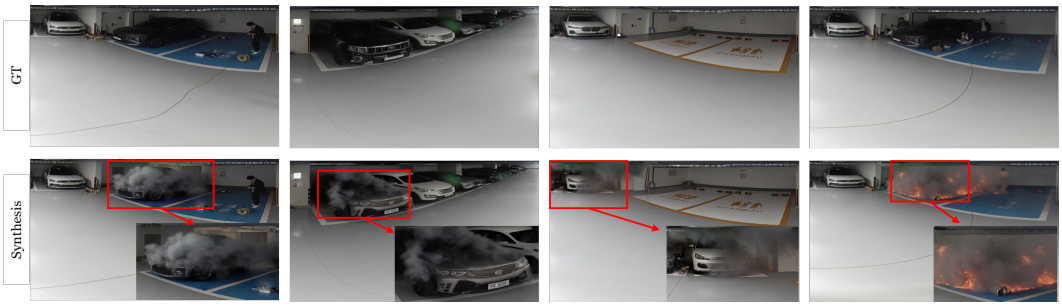

Figure 4: Examples of synthetically generated fire and smoke images.

As shown in Figure 4, our pipeline effectively synthesizes realistic fire- and smoke-engulfed scenes. However, due to the inherent instability of SDInpaint, failure cases are filtered out before the training phase. Since diffusion models are known to occasionally misinterpret text prompts despite proper conditioning, users may manually remove unrealistic samples in the final stage of the pipeline.

A.2 Dataset Details

Dataset Name	Train	Val	Test	Total	Scenario	Target	Label
Tau_house_40	1782	200	200	2182	indoor	car	fire,smoke
CCTV-fire_50	1418	166	168	1752	indoor/outdoor	diverse	fire,smoke
fire and smoke	1607	159	159	1925	outdoor	big fires	fire,smoke
firecops	222	21	9	252	outdoor	car	fire
Synthetic Data - smoke	2354	130	130	2614	indoor	car	smoke
cctv-pano_50	4071	407	407	4885	indoor	car	
Synthetic Data - fire	100	9	8	117	indoor	car	fire
Total	**11.5K**	**1K**	**1K**				

Table 3: Details of the datasets used in our experiment, including the total number of samples, scenarios, targets, and labels for each dataset.

The dataset consists of 13,727 images across seven sub-datasets for fire and smoke detection in various environments. It includes 11,554 training, 1,092 validation, and 1,081 test images. Covering indoor, outdoor, and mixed settings, it targets specific objects like cars and diverse items, with annotations for fire, smoke, or both.
Real-world datasets, such as *Tau_house_40* (Bekbol, 2024) and *cctv-fire_50* (project eyep8, 2023), provide annotated images for challenging indoor and mixed environments, whereas *fire and smoke* (MiddleEastTechUniversity, 2023) and *firecops* (firecops, 2024) focus on outdoor settings, capturing large-scale and car fires, respectively.Additionally, synthetic datasets, including *Synthetic Data - smoke* and *Synthetic Data - fire*, simulate realistic indoor and underground car fire and smoke scenarios. The largest subset, *CCTV-pano_50*, comprises 4 885 images, offering extensive data for indoor car fire scenarios. This dataset provides a well-rounded representation of both real-world and synthetic conditions, supporting robust model training for early fire and smoke detection across diverse environments, with a particular focus on confined spaces such as indoor and underground parking facilities.

B Evaluation Metrics

B.1 Basic Definitions

Let $\mathcal{D} = \{I_1, I_2, \ldots, I_N\}$ be the set of all images to be evaluated, and let $N = |\mathcal{D}|$. If the target object exists in image I_i, then $G_i = 1$. Otherwise, $G_i = 0$. Let $\mathcal{B}_i = \{b_{i1}, b_{i2}, \ldots, b_{iM_i}\}$ be the set of bounding boxes predicted by the model for image I_i. Here, M_i is the number of predicted boxes for I_i. Each predicted bounding box b_{ij} is associated with a confidence score s_{ij}. Here, s_{ij} represents the confidence score of the bounding box b_{ij}, and only boxes with scores exceeding a predefined confidence threshold τ_{pred} are considered:

$$\mathcal{B}_i^{\tau} = \{b_{ij} \mid s_{ij} \geq \tau_{pred}\}.$$

This ensures that only bounding boxes with sufficient confidence score are used for evaluation. The IoU between a predicted bounding box b_{ij} and the ground truth bounding box b_i^* (in image I_i) is defined as:

$$\mathrm{IoU}(b_{ij}, b_i^*) = \frac{|\, b_{ij} \cap b_i^* \,|}{|\, b_{ij} \cup b_i^* \,|}.$$

Let τ_{iou} be the minimum IoU threshold above which a predicted bounding box is considered a valid detection (i.e., "matched" with the ground truth). If IoU is not considered at all, detections in completely different locations may still be recognized as correct answers despite being false positives.

B.2 Per-Image Detection Success/Failure(Binary Classification)

For each image I_i, if there exists at least one predicted box whose IoU with the ground truth box is $\geq \tau_{iou}$, we regard this image as having a "successful detection" ($d_i = 1$). Otherwise, we say the detection failed ($d_i = 0$). Formally:

$$d_i = \mathbf{1}\Big(\max_{b_{ij} \in \mathcal{B}_i^{\tau}} \mathrm{IoU}(b_{ij}, b_i^*) \;\geq\; \tau_{iou}\Big),$$

where $\mathbf{1}(\cdot)$ is the indicator function, returning 1 if the condition is true, and 0 otherwise.

- $d_i = 1$ means "the model claims there is at least one instance of the object in image I_i."
- $d_i = 0$ means "the model claims no object is found in image I_i."

We can interpret (G_i, d_i) as a binary classification scenario. Thus, the standard definitions of TP, FP, FN, and TN apply:

$$\mathrm{TP} = \sum_{i=1}^{N} [\, G_i \cdot d_i \,], \quad \textbf{True Positive (TP):} \text{ The total count of images in which the object exists } (G_i = 1) \text{ and the model detects it } (d_i = 1). \tag{1}$$

$$\mathrm{FP} = \sum_{i=1}^{N} [(1 - G_i) \cdot d_i], \quad \textbf{False Positive (FP):} \text{ The total count of images in which the object does not exist } (G_i = 0) \text{ but the model claims detection } (d_i = 1). \tag{2}$$

$$\mathrm{FN} = \sum_{i=1}^{N} [G_i \cdot (1 - d_i)], \quad \textbf{False Negative (FN)} \text{ The total count of images in which the object exists } (G_i = 1), \text{ but the model fails to detect it } (d_i = 0). \tag{3}$$

$$\mathrm{TN} = \sum_{i=1}^{N} [(1 - G_i) \cdot (1 - d_i)], \quad \textbf{True Negative (TN)} \text{ The total count of images in which the object does not exist } (G_i = 0), \text{ and the model also does not detect it } (d_i = 0). \tag{4}$$

Using the TP, FP, and FN values defined above, we can compute Precision and Recall for the entire dataset:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \quad \text{Recall} \quad = \frac{\text{TP}}{\text{TP} + \text{FN}}, \quad \text{F1} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}.$$

Precision indicates the fraction of "object-present" predictions that are correct, while recall indicates the fraction of actual positives (images with the object) that are correctly identified.

In summary, if an image I_i contains at least one instance of the target object, then $G_i = 1$; otherwise, $G_i = 0$. If there is at least one predicted box with $\text{IoU} \geq \tau_{iou}$ against the ground truth box, then $d_i = 1$; otherwise, $d_i = 0$. After computing d_i for each image $I_i \in \mathcal{D}$, we sum up to get TP, FP, FN, and TN. Finally, we calculate the Precision and Recall values using the definitions above.

B.3 Per-Image Average Precision Calculation with Confidence Thresholds

To evaluate model performance across different confidence levels, we compute Precision-Recall (PR) curves by varying the confidence threshold τ_{pred} from 0 to 1 with a step size of 0.01. The Average Precision (AP) is then computed as the area under the PR curve. For each confidence threshold τ_{pred} , we compute the precision and recall using the previously defined formulas. The Precision-Recall Curve is constructed by plotting Precision against Recall at different confidence levels τ_{pred}. The AP is then computed as the area under this curve:

$$\text{AP} = \int_0^1 \text{Precision}(\text{Recall})\, d(\text{Recall}).$$

In practice, we approximate this integral using discrete summation:

$$\text{AP} \approx \sum_{k=1}^{K} (R_k - R_{k-1}) P_k.$$

whereP_k and R_k are precision and recall at different confidence thresholds and K is the total number of evaluated thresholds.

C Model Comparison

C.1 Vision-Language-Model Comparison

Model	Initial Detection Time	Detection	Params	Latency	Memory
BLIP - IC - base	17.58 s	smoke/fire	253M	0.164 s	3 GB
BLIP - IC - large	8.00 s	smoke/firee	580M	0.211 s	3.8 GB
BLIP2 - OTP - COCO	X	X	2.7 B	X	17 GB
BLIP2 - FLAN T5	8.87 s	fire	3B	0.43 s	17.5GB
LLAVA 7B (fp16)	8.1 s	fire	7B	0.85 s	16.3 GB
Florence 2 - base (fp16)	unclear	smoke/fire	0.23B	0.18s	1.2GB
Florence 2 - large (fp16)	**3.03s**	smoke/fire	0.77B	0.28s	2.3 GB

Table 4: Comparison of Vision-Language Models (VLMs) based on initial detection time, detected labels, number of parameters, frame latency, and GPU memory usage in a real underground cark park fire CCTV footage.

Table 4 presents a comparison of Vision-Language Models (VLMs) for detecting smoke and fire on a real underground car park fire CCTV footage, evaluating them based on initial detection time, detected labels, number of parameters, frame latency, and GPU memory usage. Across the models, **Florence 2 - large (fp16)** (Xiao et al., 2024) stands out with the best overall performance, featuring the fastest initial detection time of 3.03 seconds, the ability to detect both fire and smoke, a moderate latency of 0.28 seconds, and efficient GPU memory usage of 2.3GB, making it highly suitable for real-time application.
In contrast, *BLIP - IC base* (Li et al., 2022) exhibits the slowest detection time at 17.58 seconds, while *LLAVA 7B* (Liu et al., 2024) consumes the most GPU memory (16.3GB) and has the highest latency (0.85 seconds), indicating limitations for deployment in low-resource environments and real-time. Although *Florence 2 - base (fp16)* offers the smallest parameter size (0.23B) and the lowest memory usage (1.2GB), its unclear detection capability makes it less reliable for this specific task. Similarly, models like *BLIP2 - FLAN T5* (Li et al., 2023) and *LLAVA 7B* is only able to identify fire detection, reducing their versatility.

Overall, this analysis highlights the trade-offs between detection speed, computational requirements, and model versatility across different VLMs, providing evidence as to why we chose *Florence 2* as our main VLM when merging with the YOLO model.

C.2 YOLO Model Comparison

model_name	Precision	Recall	mAP50	mAP50:95
YOLOv5s	0.651	0.62	0.641	0.377
YOLOv5m	0.664	0.634	0.65	0.377
YOLOv6s	0.645	**0.662**	0.645	0.37
YOLOv6m	0.679	0.654	**0.669**	0.377
YOLOv8s	**0.681**	0.634	0.649	0.377
YOLOv8m	0.634	0.639	0.643	**0.38**
YOLOv10s	0.676	0.62	0.637	0.368
YOLOv10m	0.647	0.617	0.625	0.356

Table 5: Performance comparison of YOLO models on standard object detection evaluation metrics.

Table 5 presents a performance comparison of various YOLO models on standard object detection evaluation metrics including precision, recall, mAP50, and mAP50:95. Among the models, *YOLOv8s* achieves the highest precision (0.681), highlighting its accuracy in correctly identifying objects. *YOLOv6m* stands out as the most balanced model, achieving the highest mAP50 (0.669) and AP per-image (0.9227), showcasing its strong object detection and classification capabilities. *YOLOv8m* outperforms all models in mAP50:95 (0.38), making it the most robust under stricter IoU thresholds. *YOLOv6s* leads in recall (0.662), demonstrating its effectiveness in minimizing missed detections. *YOLOv6m* stands out in classification and detection accuracy, while *YOLOv8m*

performs well under stricter IoU conditions. Assessing broad performance aspects, **YOLOv6m** and **YOLOv8s** exhibit competitive results across all metrics, making them versatile choices for general-purpose tasks when the object detection model is used independently.

C.3 Full YOLO model comparison with and without VLM

Model	precision		recall		F1	Model	precision		recall		F1
	fire	smoke	fire	smoke	Score		fire	smoke	fire	smoke	Score
YOLOv5s	0.835	0.894	0.766	0.811	0.8248	+VLM	0.838	0.883	0.881	0.885	0.8716
YOLOv5m	0.842	0.919	0.784	0.811	0.8369	+VLM	0.839	0.905	0.885	**0.89**	0.8797
YOLOv6s	**0.861**	0.915	0.853	0.797	0.8553	+VLM	0.85	0.901	0.885	0.847	0.8707
YOLOv6m	0.86	0.911	0.872	0.849	0.8728	+VLM	0.837	0.904	0.894	0.879	0.8784
YOLOv8s	0.843	**0.944**	0.835	0.833	0.8627	**+VLM**	0.838	0.931	0.904	0.885	**0.8895**
YOLOv8m	0.843	0.936	0.839	0.838	0.8632	+VLM	0.839	0.918	**0.908**	**0.89**	0.8885
YOLOv10s	0.851	0.934	0.789	0.814	0.8446	+VLM	0.85	0.912	0.858	0.882	0.8755
YOLOv10m	0.841	0.943	0.803	0.822	0.8504	+VLM	0.842	0.919	0.881	0.866	0.8770

Table 6: Comparison of various YOLO models with and without VLM integration, evaluated using our proposed per-image binary detection metric.

Table 6 highlights the performance of various YOLO models with and without VLM integration, focusing on our proposed metric: precision, recall, and F1 score. *YOLOv6s* achieves the highest precision for fire detection at 0.861, while *YOLOv8s* demonstrates the best precision for smoke detection at 0.944, showcasing its strong capability in identifying smoke. Among YOLO models alone, *YOLOv6m* records the highest F1 score at 0.8728, indicating its well-rounded effectiveness. With VLM integration, however, *YOLOv8m* achieves the highest overall F1 score at 0.8895, demonstrating the advantages of combining YOLO models with VLM for enhanced detection accuracy. These results highlight the superior performance of ***YOLOv8s*** model, particularly when paired with VLM, making it the most effective choice for car fire and smoke detection tasks.

C.4 Algorithm of Inference on Real-Time CCTV Feed

Algorithm 1 Real-Time Inference with YOLO and Florence2 Integration

Input: Trained YOLO model, Florence2 VLM, CCTV feed, initial threshold τ_{init}, modified threshold τ_{mod}
Output: Alert trigger for fire/smoke detection
Initialize $\tau_{pred} \leftarrow \tau_{init}$
for each frame in CCTV feed **do**
  Pass frame to Florence2 VLM with prompt: "Is there smoke or fire?"
  if VLM detects smoke or fire **then**
    $\tau_{pred} \leftarrow \tau_{mod}$ {Lower threshold for enhanced sensitivity}
  else
    $\tau_{pred} \leftarrow \tau_{init}$ {Maintain initial threshold to reduce false positives}
  end if
  Perform object detection using YOLO model with confidence threshold τ_{pred}
  if YOLO predicts fire or smoke **then**
    if Validated by VLM **then**
      Trigger alert to notify security
    end if
  end if
end for

D QUALITATIVE RESULTS

Figure 5: Qualitative results on our constructed test set. The first row presents the YOLOv8s predictions, while the second row displays the corresponding ground truth annotations. We present examples of successful cases, where the predicted bounding boxes closely match the ground truth, achieving an IoU exceeding the threshold $\mathcal{T}_{iou}$ and demonstrating high confidence scores.

REFERENCES

Bekbol. tau house dataset. https://universe.roboflow.com/bekbol/tau_house_1, 2024.

John Canny. A computational approach to edge detection. *IEEE Transactions on pattern analysis and machine intelligence*, (6):679–698, 1986.

firecops. firecops dataset. https://universe.roboflow.com/fire-sssor/firecops, 2024.

Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *International conference on machine learning*, pp. 12888–12900. PMLR, 2022.

Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International conference on machine learning*, pp. 19730–19742. PMLR, 2023.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36, 2024.

MiddleEastTechUniversity. fire and smoke detection dataset. https://universe.roboflow.com/middle-east-tech-university/fire-and-smoke-detection-hiwia, 2023.

Patrick Pérez, Michel Gangnet, and Andrew Blake. Poisson image editing. In *Seminal Graphics Papers: Pushing the Boundaries, Volume 2*, pp. 577–582. 2023.

project eyep8. cctv fire dataset. https://universe.roboflow.com/project-eyep8/cctv-fire, 2023.

J Redmon. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10684–10695, 2022.

Bin Xiao, Haiping Wu, Weijian Xu, Xiyang Dai, Houdong Hu, Yumao Lu, Michael Zeng, Ce Liu, and Lu Yuan. Florence-2: Advancing a unified representation for a variety of vision tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4818–4829, 2024.

Hu Ye, Jun Zhang, Sibo Liu, Xiao Han, and Wei Yang. Ip-adapter: Text compatible image prompt adapter for text-to-image diffusion models. *arXiv preprint arXiv:2308.06721*, 2023.

Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 3836–3847, 2023.