http://dx.doi.org/10.1016/j.knosys.2024.111547

FireDM: A Weakly-Supervised Approach for Massive Generation of Multi-scale and Multi-scene Fire Segmentation Datasets

Hongtao Zheng[a,b], Meng Wang[a], Zilong Wang[a] and Xinyan Huang[a,b,*]

[a]Department of Building Services Engineering, Hong Kong Polytechnic University, Hong Kong
[b]The Hong Kong Polytechnic University Shenzhen Research Institute, Shenzhen, China

ARTICLE INFO

Keywords:
Fire Segmentation
FireDM
Stable Diffusion XL 1.0
Stable Diffusion 2.1
ChatGPT-Fire
Multi-Scale

ABSTRACT

Data availability and quality are crucial for the development of semantic segmentation techniques. However, creating high-quality fire scene datasets in a safe and efficient manner remains an unsolved challenge. To fill this gap, we introduce FireDM, the first method to generate unlimited fire segmentation datasets at virtually no cost. FireDM takes full advantage of the combined strengths of a combination of pre-trained diffusion models (Stable Diffusion XL 1.0 and Stable Diffusion 2.1) and text-guided diffusion using ChatGPT4-Fire to generate multi-scale and detail-rich fire images. The innovative fire-decoder module in FireDM then efficiently converts the cross-attention and multi-scale feature maps obtained during diffusion into accurate segmentation masks. This process requires only about 100 images and their corresponding segmentation masks for training. In our experiments, we trained the segmentation algorithms using the large-scale segmentation dataset generated by FireDM and all publicly available fire segmentation datasets respectively, and found that the segmentation algorithms trained with the former dataset outperformed the latter by at least 5% or more in terms of IoU, accuracy, F1-score and AP. This demonstrates the capability of FireDM in expanding a limited fire segmentation dataset. In addition, the multi-scale datasets generated by FireDM are close to the input size of different segmentation algorithms, which significantly reduces the loss of information caused by resizing the image (e.g., cropping and scaling). Finally, we have created the world's first high-quality fire segmentation dataset benchmark using FireDM. The complete code and dataset of FireDM are publicly available at https://github.com/ZhengHongtao2001/FireDM.

1. Introduction

Fires, whether natural or man-made, pose a significant threat to human life, the natural environment and infrastructure. Fires can be broadly categorised by where they occur, for example, wildland fire, structural fire, vehicle fire, and industrial fire. Wildland (or forest) fires can seriously jeopardise the ecological balance and damage residential area [1, 2]. Structural fires mainly affect man-made infrastructures, causing significant economic losses and endangering human lives [3, 4]. Vehicle fire is a growing problem, as the number of new vehicles and tunnels soars. Many vehicle fires in roads, parking lots, and tunnels led to significant injuries, deaths, and economic losses, underscoring their seriousness [5, 6].

To support firefighting operations and emergency response, the accurate detection and localization of fire sources have been a hot research topic. Traditional techniques such as smoke and heat sensors are widely used in buildings and infrastructures, but they cannot handle open fires or provide very limited information about fire hazards [7, 8]. With the burgeoning development of data-driven learning algorithms, a lot of recent research has adopted the computer vision in monitoring fire phenomena [9, 10, 11], fire evacuation processes [12] and so on. Image detection and segmentation algorithms based on deep learning have emerged as the most popular methods [13, 14, 15]. Technically, the study of fire segmentation algorithms is more meaningful compared to fire detection algorithms. Fire segmentation not only encompasses the concept of fire detection but further extracts more information from fire images and videos and provides detailed information about the fire scenes, such as the shape, size and power of flame [16, 17], fire location [18], and indoor fire loads [19, 20]. Therefore, it can enhance our understanding of fire dynamics and evolution compared to a simple fire detection.

Research in fire segmentation algorithms faces numerous challenges, with two particularly significant issues being: **1)** The influence of external factors causing fire forms to vary with the intensity of burning. **2)** The distinct characteristics of flames produced by different materials, adding to the complexity of fire segmentation. Currently, the focus in this field leans towards enhancing the complexity of algorithms to achieve better training results. However, a major problem is that most advanced algorithms are not open-source, and there is a lack of unified standards for evaluation, making effective comparisons between different studies challenging. Additionally, even the most advanced algorithms can fall short if the training datasets are limited in scope, quantity, or quality. This has significantly impeded the progress in the field of fire segmentation. Therefore, there is an urgent need to propose a high-quality fire segmentation dataset that can serve as a standard benchmark for the entire field, akin to ImageNet [21] and COCO [22], to advance the field further.

*Corresponding author
E-mail addresses: zht20010605@126.com (H. Zheng), meng-rick.wang@connect.polyu.hk (M. Wang), zilong.wang@connect.polyu.hk (Z. Wang), xy.huang@polyu.edu.hk (X. Huang)

It is important to emphasize that a segmentation dataset comprises two critical components: images and their corresponding segmentation masks. These aspects are precisely where the difficulty in creating a benchmark for fire segmentation datasets lies. Firstly, image collection itself is a challenge, particularly in the domain of fire scenes, where the process is fraught with difficulties and dangers. Secondly, compared to common subjects like dogs and cats, annotating segmentation in fire scenes is significantly more complex. This complexity is due to the less distinct segmentation contours and proportional distribution found in fire scenes. Additionally, the process of pixel-by-pixel annotation in segmentation is a tedious and time-consuming task. For a long time, this issue was often overlooked due to its complexity. However, fortunately, the rapid development of image generation algorithms such as StyleGAN [23, 24], DALL-E [25, 26], and Stable Diffusion (SD, [27, 28]) has brought new hope for achieving previously seemingly impossible goals in the field of fire segmentation.

In recent years, pioneering research has begun exploring the use of generative models for synthesizing data to assist in model training and even replace real data in perceptual tasks. During the GAN era around 2020, innovative studies such as DatasetGAN [29] and BigDatasetGAN [30] emerged, designing decoders to generate pixel-level labels for image segmentation tasks using pre-trained GAN feature spaces. This was followed by dataset augmentation algorithms based on diffusion models like Stable Diffusion (SD), as seen in works such as [31] and [32]. The method in [31] generates diverse foreground object images through text-to-image diffusion models and combines them with segmentation algorithms to extract foreground masks, which are then merged with background images. While innovative in data synthesis, this approach might disrupt the natural correlation between the target and its environment, limiting overall image realism.

In contrast, the [32] method relies on a universal perceptual decoder to extract extensive perceptual annotations from the latent space of diffusion models, combined with rich textual cues from large language models like GPT-4, thereby enhancing the diversity and quality of generated data. This approach significantly improves upon previous methods, yet still faces challenges: **1)** The pre-trained diffusion model (SD 1.4) used may lack realism in generating fire scene images, leading to severe distortions in people and vehicles, as shown in Fig. 1. This is largely attributed to the fact that SD 1.4 was trained on a smaller and lower-resolution dataset, while the limited number of training steps limits the depth of training of the model. In addition, the text encoder it uses is OpenAI's CLIP ViT-L/14 ([33]), which is only one-fifth the size of the OpenClip text encoder used in SD 2.0 and subsequent Stable Diffusion versions. Thus, compared to SD 2.0 and subsequent versions, SD 1.4 falls short in its ability to understand complex text descriptions and accurately translate these understandings into image content. **2)** The pre-trained SD 1.4 model was trained using images in the training dataset with a resolution of mainly 512×512 pixels, and this approach also naturally limits the pixel size of the images generated by SD 1.4 during the inference process to 512×512 pixels. **3)** The text generation tool used by DatasetDM is ChatGPT 4.0. However, unfortunately, it can generate descriptions of fire scenarios that are unrealistic, unprofessional, and unscientific. This issue primarily stems from ChatGPT-4 being a general-purpose large language model, which has not been specifically optimized or adjusted for the fire domain. Consequently, these inaccurate descriptions can directly impact the quality and logical coherence of the generated fire scene images, negatively affecting the practicality and effectiveness of the overall dataset. **4)** Algorithms like DatasetDM, which serve as general-purpose dataset augmentation tools, excel primarily in generating segmented datasets with relatively simple and known contour details. Examples include targets such as airplanes and bicycles in VOC [34], and objects like cakes and carrots in COCO [22], all of which possess uniform structures and clear outlines. However, flames, as segmentation objects within fire scenes, demand extremely high accuracy in details due to their rapidly changing and highly complex morphology and contours, without any fixed shape standards. The intrinsic lighting effects of flames, combined with the constantly changing background and the propagation characteristics of flames in dynamic scenes, significantly increase the difficulty of segmentation. Additionally, the range of flame sizes is vast, from tiny sparks to large fires that cover extensive areas, requiring segmentation algorithms to adapt to features across various scales.

To address the aforementioned unresolved issues, we introduce a novel fire diffusion model, FireDM, which significantly optimizes the DatasetDM approach: **1)** We have upgraded the pre-trained diffusion model from Stable Diffusion 1.4 to a more advanced combined pre-trained diffusion model system (incorporating SD 2.1 and SD XL 1.0 (base)). SD 2.1 and SDXL 1.0 have significant advantages over SD 1.4 in terms of overall structure, such as increased cross-attention dimensions, and they have been trained on larger, higher-quality datasets compared to SD 1.4. These factors collectively enhance the quality of the final generated images and improve the plausibility and realism of image details. Notably, the image resolutions in the dataset for pre-training SD 2.1 are predominantly 512×512 pixels and 768×768 pixels, while the dataset for pre-training SDXL 1.0 includes images with a resolution of 1024×1024 pixels. This difference enables the combined diffusion models of SD 2.1 and SDXL 1.0 to generate images of 512×512 pixels, 768×768 pixels, and 1024×1024 pixels, respectively, thus accommodating the need for different resolutions. **2)** By applying techniques like Prompt Engineering that do not require retraining, we have upgraded the text prompting tool from ChatGPT 4.0 to ChatGPT4-Fire, a specialized AI expert system for the fire domain. ChatGPT4-Fire, optimized for GPT-4, provides more professional and precise outputs by deeply understanding a vast array of standardized textual descriptions and scenarios related to fires. **3)** Based on the structure of Mask2Former [35], we introduce a new mask generation module, Fire-Decoder, which efficiently processes multi-scale features through cross-scale fusion and incorporates

textual information to better interpret image information for generating more accurate segmentation masks.

Figure 1: The image of a fire scene with anomalies generated by the DatasetDM. The image has severely deformed people and cars marked out in red, and these anomalies greatly damage the integrity and realism of the image.

2. Related Work

2.1. All publicly fire segmentation datasets

This section focuses on combing all publicly available fire segmentation datasets as follows:

BoWFire [36]: The BowFire dataset consists of 226 images of varying resolution (119 fire images and 107 non-fire images). The fire images represent a variety of fire scenarios (forest fires, structure fires.) BowFire contains segmentation masks for the flames in the 119 fire images. But the dataset is too small and far from adequate for training current popular segmentation algorithms.

CorsicanFire dataset [37]: The CorsicanFire dataset contains a large number of fire images with multiple resolutions (1135 RGB images and their corresponding masks) and is widely used for fire segmentation. It describes visual information about the fire such as colour, fire distance, brightness, presence of smoke and different weather conditions. But this dataset is small and also its image clarity is not as good as it should be, which can affect the training of the segmentation algorithm to some extent.

FLAME [38]: The FLAME dataset is a comprehensive fire dataset containing multiple tasks, where the segmentation dataset consists of 2003 RGB images with a resolution of 3480×2160 pixels and their corresponding fire segmentation task masks. However, this dataset has a single scene and flame size, which limit the generalisation ability of the segmentation algorithm.

As modernisation accelerates, fire scenarios are becoming increasingly intricate and complex. It is gradually recognised that the three currently available public fire datasets are no longer sufficient to support the development of efficient algorithms that can accurately cope with modern complex fire scenarios. Further, existing evaluations of various fire segmentation algorithms are mostly based on private datasets, which largely prevents open and transparent performance comparisons between these algorithms. Therefore, there is an urgent need to develop a larger and higher quality fire image segmentation dataset that can become a new standard for training and evaluating various segmentation algorithms.

2.2. Generative models for Text-to-Image

When individuals immerse themselves in textual stories, they often conjure vivid mental images through imagination, enhancing comprehension and enjoyment. This has spurred increased interest in the realm of Text-to-Image research.

In this landscape, AlignDRAW [39] stands out as a pioneering endeavor, proficient in generating images from natural language. However, it grapples with the challenge of occasionally producing unrealistic outcomes. Subsequently, Text-conditional GAN emerged as a groundbreaking end-to-end differential architecture. Distinct from GAN-based methods [40, 41, 42, 43], primarily tailored for small-scale data, autoregressive methods harness extensive datasets for text-to-image generation. Exemplary models in this category include OpenAI's DALL-E [25] and Google's Parti [44]. Despite their strengths, these autoregressive methods are not immune to issues such as high computational costs and sequential error accumulation.

More recently, there has been a noteworthy shift toward diffusion models (DM) as the new frontier in Text-to-Image generation, such as DALL-E2 ([26]), Stable Diffusion. These models have garnered significant attention due to their robust performance and innovative approaches.

2.3. Optimization Methods for General LLMs

General large language models (LLMs) [45, 46, 47], is extensively applied across various aspects of daily life. However, due to the inherent limitations of general LLMs in highly specialized areas such as fire detection, where depth and accuracy are crucial, it is common to employ methods like tuning operation, Prompt Engineering to enhance their domain-specific expertise.

Tuning operation for LLMs involve refining a pre-trained model to achieve specialized expertise in specific domains through additional training. To optimize the tuning effects of models while reducing the required hardware resources, numerous methods have been proposed, such as prefix-tuning [48], prompt-tuning [49], p-tuning [50, 51], instruct-tuning [52], instruction prompt tuning [53], Lora-Adapters [54] and PEFT [55]. These strategies aim to achieve an optimal balance between resource consumption and performance enhancement through innovative tuning approaches. While all of the above methods improve parameter efficiency over general fine-tuning methods, they are still computationally intensive in a broad sense, especially when huge LLMs are involved. Also when fine-tuning models on specific domains or datasets, these methods still run the risk of overfitting. And all these methods require a high level of operator expertise. It is also worth noting that a prerequisite for these methods is that the pre-trained models are open-source.

Prompt Engineering [56, 57, 58] refers to the careful design and improvement of input prompts or the provision of knowledge content related to a question, to effectively guide ChatGPT in generating the desired output. It involves

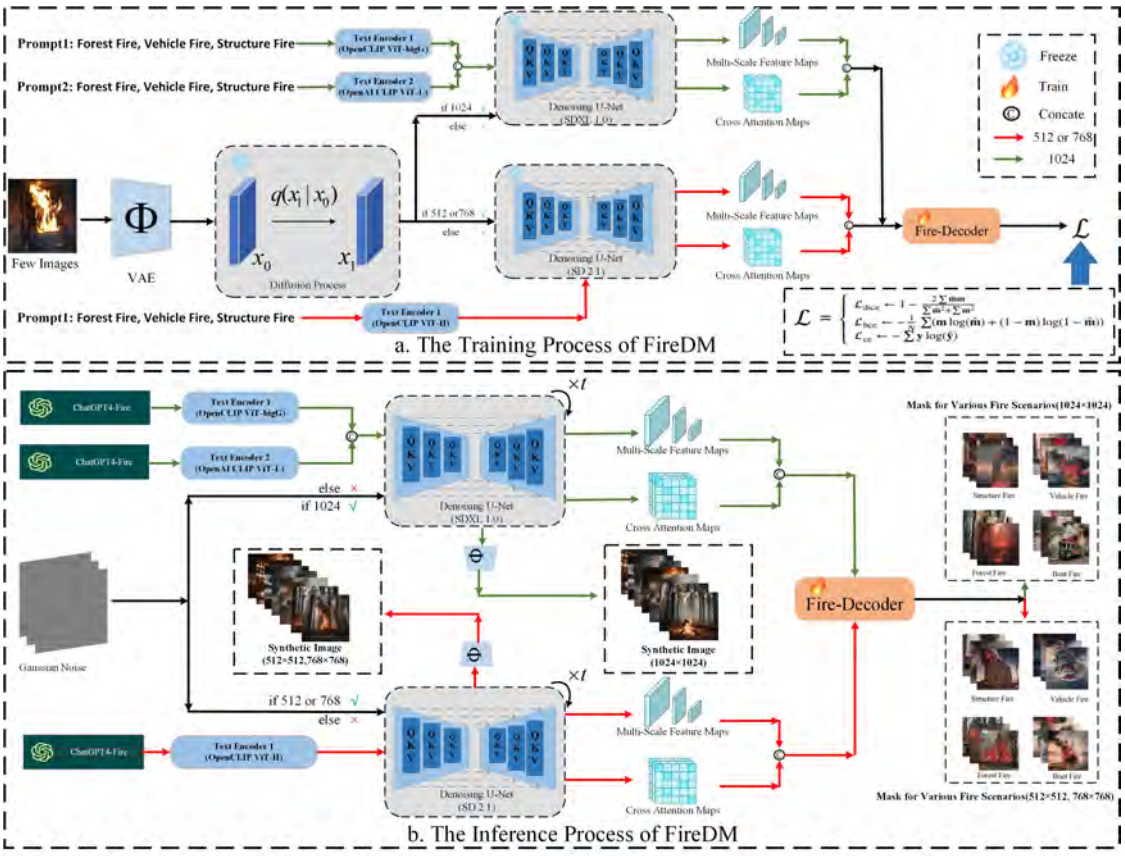

Figure 2: The overall process and detailed architecture of FireDM. The **a** shows the training process of FireDM and the **b** shows the inference process of FireDM, where the prompt input in **a** is the category of fire, e.g. Forest Fires, Vehicle Fires, Structure Fires. In the FireDM, we utilize the Stable Diffusion model, specifically, a combination of the SD 2.1 and SDXL 1.0 diffusion models. However, they are selectively employed based on the image resolution of the generated dataset, and they operate completely independently without any interaction between them.

strategically formulating questions for the model, with the aim of maximizing the quality and relevance of the model's responses to those queries. However, this approach has its limitations. It requires a certain level of quality in the design of Prompts and is constrained by the existing knowledge and training of the pre-trained model. Additionally, large models typically have limitations on the length of input sequences. Excessively long Prompts may exceed these limits and get truncated, leading to a decrease in the quality of the model's output and negatively impacting the results.

3. Fire Diffusion Model (FireDM)

This section primarily introduces the overall process and detailed architecture of FireDM, and its specific details are illustrated in Fig. 2.

3.1. Diffusion model infrastructure

The Stable Diffusion model, essential to FireDM's scalability of generating fire scene dataset, is rooted in the principles of diffusion models (DMs) and generative modeling of latent representations.

Diffusion models: DMs [59, 60, 61, 62] are probabilistic models that learn a data distribution $p(x)$ by progressively denoising data. This process is akin to reversing a Markov chain. These models use a series of denoising autoencoders $\epsilon_\theta(x_t, t)$ to predict denoised versions of noisy inputs x_t, where x_t is a perturbed version of the original input x.

The core of DMs lies in approximating $p(x)$, the target data distribution, as follows:

$$p(x) \approx \prod_{t=1}^{T} p(x_{t-1}|x_t) \tag{1}$$

where $p(x_{t-1}|x_t)$ is the conditional probability of obtaining a less noisy image x_{t-1} from its noisier counterpart x_t. The objective of DMs is to closely learn this distribution for generating new, realistic samples.

The key performance metric for these models is the accuracy of denoising, represented as:

$$L_{DM} = \mathbb{E}_{x,\epsilon\sim\mathcal{N}(0,1),t}\left[\|\epsilon - \epsilon_\theta(x_t, t)\|_2^2\right] \tag{2}$$

This equation measures the squared distance between the actual noise ϵ and the model's predicted noise $\epsilon_\theta(x_t, t)$.

Stable Diffusion: A DM variant leverages a Variational Autoencoder (VAE) for encoding and decoding. The encoder ϕ reduces high-dimensional images to a lower-dimensional

latent space. Conversely, the decoder ϕ^{-1} reconstructs the image from this latent representation. This streamlined process differentiates Stable Diffusion by operating in a reduced-dimensional space, unlike traditional models in high-dimensional pixel space. This approach emphasizes essential image semantics and boosts computational efficiency, vital for likelihood-based generative models. The objective of Stable Diffusion is captured by:

$$L_{SD} := \mathbb{E}_{\phi(x),\epsilon\sim\mathcal{N}(0,1),t}\left[\left\|\epsilon - \epsilon_\theta\left(z_t, t\right)\right\|_2^2\right] \quad (3)$$

where $\phi(x)$ is the encoded representation of the input image x, transforming high-dimensional data into a more manageable latent representation z_t at each timestep t. The model aims to predict the denoising process accurately for these latent representations.

Conditional Generative Modeling: Stable Diffusion enhances DMs with conditional generative modeling [63, 64], enabling them to model conditional distributions $p(z \mid y)$ using a conditional denoising autoencoder, $\epsilon_\theta(z_t, t, y)$. This setup allows for the modulation of the synthesis process with various inputs y, ranging from textual data [40] to semantic maps [65, 66].

A key feature in Stable Diffusion is the incorporation of a cross-attention mechanism [67] within its U-Net architecture. This mechanism enhances the model's ability to process various inputs [68], such as language prompts, through a domain-specific encoder τ_θ. This encoder transforms y into an intermediate representation $\tau_\theta(y)$, which interacts with the U-Net via cross-attention layers:

$$Q = W_Q^{(i)}\cdot\varphi_i(z_t), \quad K = W_K^{(i)}\cdot\tau_\theta(y), \quad V = W_V^{(i)}\cdot\tau_\theta(y) \quad (4)$$

These layers utilize learnable projection matrices $W_Q^{(i)}$, $W_K^{(i)}$, and $W_V^{(i)}$ to project the representations into the attention mechanism. The attention maps Attention(Q, K, V) generated from the attention layers are also crucial for capturing the relationship between textual inputs and generated images during the training and inference phases of FireDM, enhancing the FireDM's capacity to handle complex text-visual interplays. The conditional generative model of Stable Diffusion is trained on image-conditioning pairs with the objective:

$$L_{SD} := \mathbb{E}_{\phi(x),y,\epsilon\sim\mathcal{N}(0,1),t}\left[\left\|\epsilon - \epsilon_\theta\left(z_t, t, \tau_\theta(y)\right)\right\|_2^2\right] \quad (5)$$

Here, both τ_θ and ϵ_θ are optimized jointly, with the flexibility to parameterize τ_θ using domain-specific experts, such as transformers, especially when y includes text prompts.

Variations of Stable Diffusion: FireDM differentiates itself by strategically using different pre-trained diffusion models (a combination of SD 2.1 and SDXL 1.0) to provide customised performance capabilities depending on the resolution requirements of the generated dataset. When we want to generate a dataset with a resolution of 512×512 or 768×768, FireDM enables SD 2.1, mainly because the dataset trained in SD 2.1 pre-training contains mainly images with these two resolutions. When we want to generate a dataset with a resolution of 1024×1024, FireDM enables SDXL 1.0, mainly because the SDXL 1.0 pre-training dataset contains mainly images with this resolution. This optionality ensures that the user chooses the most effective model architecture for their specific resolution needs, underscoring FireDM's commitment to performance and flexibility. To clearly highlight the advantages of FireDM, particularly in generating perceptual masks using Fire-Decoder, we have meticulously compared the diffusion structures of FireDM and DatasetDM. A key reason lies in FireDM's selection of SD 2.1 and SDXL 1.0, which exhibit substantial improvements over SD 1.4 used by DatasetDM. These improvements are first evident in their use of more powerful text encoders and training on larger, more comprehensive datasets, enabling the U-Net to process a greater volume of effective information. Moreover, the U-Net structures of SD 2.1 and SDXL 1.0 have undergone significant enhancements, including a more refined attention mechanism, increased model capacity, improved scalability in upscaling and downscaling processes, and better handling of large-size images. These structural optimizations not only enhance the efficiency of information processing but also result in the generation of more precise cross-attention maps and multi-scale feature maps, providing higher quality inputs for the Fire-Decoder. For specific details on the structural differences between SD 2.1 and SDXL 1.0 compared to SD 1.4, further reference can be made to [28].

3.2. ChatGPT-Fire Build Details

Based on the brief analysis of the two types of specialization methods commonly used in LLMs in Section 2.3, we ultimately chose to use Prompt Engineering methods to enhance the expertise of ChatGPT 4.0 in the field of fire safety. The main reasons are as follows: **1)** First, the foundational knowledge base of ChatGPT 4.0 is already very robust, providing a solid platform for specialized applications. Moreover, the current version of ChatGPT 4.0 has expanded its input sequence limit compared to previous versions (such as ChatGPT 3.5), allowing single interactions to accommodate up to 4096 tokens or more. This improvement enables the creation of more detailed and comprehensive prompts without the risk of truncation. **2)** Secondly, OpenAI has only opened APIs for ChatGPT 3.5 and older models, and these are also limited to using fine-tuning methods that consume significant computational resources. **3)**Additionally, when we fine-tuned the ChatGPT 3.5 model, we found its performance could only be improved to a level close to that of ChatGPT 4.0. Therefore, the fine-tuning method is clearly not our best choice. All in all, we only need to provide ChatGPT 4.0 with well-designed hints, and then the Prompt Engineering approach is perfect for optimising the specialisation of ChatGPT 4.0.

3.2.1. Details of ChatGPT4-Fire

We rely on the least-to-most method [69] and the knowledge enhancement method in Prompt Engineering to optimise the level of expertise of ChatGPT 4.0 with respect to

the fire domain. The overall implementation details of this method are shown in Fig. 3.

Fig. 3C succinctly demonstrates how two methods are applied to optimize ChatGPT 4.0 into a fire safety expert (ChatGPT4-Fire). The detailed process is as follows: **(1)** Firstly, we start by applying the least-to-most method (see Fig. 3A) to constrain the output sentence structure of ChatGPT 4.0, dividing the structure into four parts, Fire Scene Description, Image Resolution, Fire Scenario Category, and Image Style. We then direct ChatGPT4-Fire to generate content that matches these four aspects accordingly. For instance, when we prompt ChatGPT4-Fire to "describe a forest fire scene," ChatGPT4-Fire will produce content related to each of the four segments, creating a comprehensive description of a forest fire. **(2)** Secondly, due to the presence of catastrophic forgetting and gaps in domain-specific knowledge, such as forestry and fire safety in ChatGPT 4.0, we have utilized knowledge enhancement techniques by incorporating relevant professional books and other resources as external knowledge for ChatGPT 4.0 to query and reference. The specific operational workflow of this method is illustrated in Fig. 3B. The primary sources of specialized knowledge provided to ChatGPT 4.0 include Books, News Articles and Reports, Professional Blogs and Posts, Official Technical Manuals, Academic Papers and Reports, and Online Courses and Resources. Due to space limitations, we have selected and showcased some of the most important books, etc., in Table. A15 of the appendix.

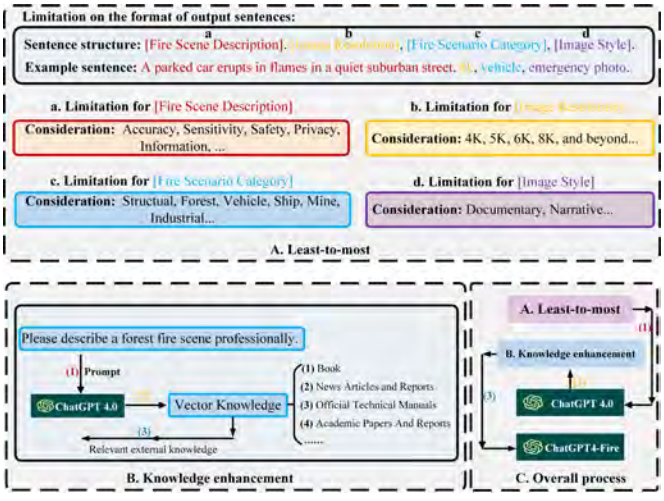

Figure 3: Details on Optimizing ChatGPT 4.0 into Fire Safety Expert ChatGPT4-Fire Using Prompt Engineering Method.

3.3. The Training Process of FireDM

During the training phase of the FireDM model, we utilize the Stable Diffusion model, specifically, a combination of the SD 2.1 and SDXL 1.0 diffusion models. However, they are selectively employed based on the image resolution of the generated dataset, and they operate completely independently without any interaction between them. To clearly articulate the core logic behind FireDM's operation, we use SDXL 1.0 as the representative of the Stable Diffusion model. SDXL 1.0 includes a Variational AutoEncoder (VAE) denoted as ϕ, a U-Net denoted as ϵ, and two text encoders, represented as τ_1 and τ_2, respectively. To maintain the integrity of its pre-training, all parameters of these components are kept frozen throughout the training process.

Initially, we sampled image-description pairs, and input images are processed through the VAE ϕ, transforming them into a latent representation, denoted as x_t (where $t = 0$ represents the original latent state). These representations are then subjected to a controlled noise addition, following a denoising process conditioning by the concatenated encoded descriptions $\tau(S)$ by the two Text Encoders, adhering to the established Stable Diffusion methodology.

Within this framework, we introduce our novel component, the Fire-Decoder model ψ, which functions as a segmentation model. The U-Net ϵ is employed to methodically reverse the diffusion process, denoising the latent representations from their maximum noise step back to $t = 0$. During this reverse diffusion, spanning from steps $t = 1$ to $t = 0$, we extract multi-scale features and generate attention maps of various sizes from the U-Net. These elements, along with the average pooled encoded image descriptions as language prompts $\tau_1(S)$, serve as inputs for the Fire-Decoder ψ.

The primary role of the Fire-Decoder ψ is to predict labels and masks. The overall loss $\mathcal{L}$, which guides the training, is formulated using a combination of Dice loss, binary cross-entropy, and classification cross-entropy. Significantly, the parameters of the Fire-Decoder, denoted as θ, are updated via gradient descent, driven by this computed loss $\mathcal{L}$. Integrating the Fire-Decoder model into the SDs framework allows us to capitalize on its inherent strengths, particularly in generating new instances with high-quality masks.

As outlined in the pseudo-codes presented in Algorithm 1, further details of the training process, along with the nuances of each step, will be comprehensively elaborated in the subsequent sub-sections.

3.3.1. *The Diffusion Inversion Process*

Fig. 2a illustrates the initial focus of FireDM's training process: a diffusion inversion strategy using the SDXL. This procedure commences with the selection of a random image $\mathcal{I} \in \mathcal{R}^{H\times W\times 3}$ from the training dataset. The image is then input into the VAE ϕ of SDXL for efficient latent feature extraction. Once the latent features are extracted as $\boldsymbol{x}_0 = \phi(\mathcal{I})$, a multi-step noise addition is executed in a single operation. This step approximates the posterior distribution $q\left(\boldsymbol{x}_{1:T} \mid \boldsymbol{x}_0\right)$ through a fixed, non-trainable Markov chain. This chain incrementally introduces Gaussian noise into the image according to a predefined schedule $\beta_1, \ldots, \beta_T$, described as follows:

$$
\begin{aligned}
q\left(\mathbf{x}_t \mid \mathbf{x}_{t-1}\right) &= \mathcal{N}\left(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t \mathbf{I}\right), \\
q\left(\mathbf{x}_{1:T} \mid \mathbf{x}_0\right) &= \prod_{t=1}^{T} q\left(\mathbf{x}_t \mid \mathbf{x}_{t-1}\right).
\end{aligned} \tag{6}
$$

Here, β_t values, which range as $0 < \beta_1 < \beta_2 < \cdots < \beta_T < 1$, determine the noise intensity and follow a gradual increase, typically along a linear or cosine curve. In the case of SDXL, the default noise scheduler employs a total of 1000 steps to ensure a fine-grained denoising process. The cumulative effect of noise addition across all steps is represented by a marginal Gaussian distribution:

$$q\left(\mathbf{x}_t \mid \mathbf{x}_0\right) = \mathcal{N}\left(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, \left(1-\bar{\alpha}_t\right)\mathbf{I}\right) \tag{7}$$

with $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{i=1}^{t} \alpha_i$. This indicates that the noisy image $\mathbf{x}_t$ at any given step t is generated by directly adding controlled noise to the initial state $\mathbf{x}_0$. After the noise addition, the latent representation $\mathbf{x}_t$ is acquired.

Concurrently, the dual text encoders of SDXL, τ_1 (OpenCLIP ViT-bigG [70]) and τ_2 (OpenAI CLIP ViT-L [33]), process the input text $\mathbf{s}$. Each encoder maps $\mathbf{s}$ into intermediate representations, $\tau_1(\mathbf{s})$ and $\tau_2(\mathbf{s})$, situated in $\mathbb{R}^{M \times D_\tau}$. These representations are then merged to form a unified representation $\tau(\mathbf{s})$.

In the subsequent phase of the process, both $\mathbf{x}_t$ and $\tau(\mathbf{s})$ are inputted into the U-Net with cross-attention layers. Here, $\mathbf{x}_t$ serves as the noisy sample to be denoised, while $\tau(\mathbf{s})$ acts as the conditioning element. The cross-attention layer is a fundamental module for updating object queries by aggregating image context. Since the cross-attention layer is permutation-invariant, both queries and keys require positional information, introducing order and providing a positional prior to encourage high attention scores for areas where position is significant. Specifically, N, D, H, and W represent the number of queries, hidden dimensions, the height, and the width of the image features, respectively. We obtain the image features K and their position encoding K_p, resulting in the key $K = K + K_p$, where $K \in \mathbb{R}^{HW \times D}$. We also have object queries $Q \in \mathbb{R}^{N \times D}$, where each query consists of a content query Q_c and a position query Q_p. Therefore, the cross-attention operation can be represented as:

$$\mathrm{A}(Q, K, V) = \mathrm{softmax}(QK^T/\sqrt{D}) \cdot V \tag{8}$$

Integrating this framework, FireDM adopts a one-step forward feature extraction strategy during training, a method highlighted by [71], known for its dual benefits of faster convergence and improved performance. We specifically define a single noise-adding step at $t = 1$, enabling the extraction of the finest multi-scale features and attention maps from each cross-attention layer of the U-Net while denoising from step $t = 1$ to $t = 0$. The input is processed by U-Net from downsampling to upsampling in three distinct latent resolutions (32×32, 64×64, and 128×128), which correspond to the intervals of Up/Down/Mid Blocks as depicted in Fig. 2a, thereby generating multi-scale feature maps F. Additionally, we collect the attention maps A from each cross-attention block shown in Fig. 2a, to augment the fidelity and detail of the features captured during this process, particularly those interacting with the textual modality. Therefore, the final output of this stage is represented as:

$$\{F, A\} \leftarrow \upsilon(\mathbf{x}_t, t, \tau(\mathbf{s})) \tag{9}$$

This integration of feature extraction with informative attention exemplifies our FireDM approach, leveraging state-of-the-art DMs to achieve exceptional performance in image synthesis tasks. Moving forward in the training process, we focus on training the Fire-Decoder component.

Algorithm 1: The Training Process of FireDM

Input: Training dataset $\mathcal{D}$ with tuples $(\mathcal{I}, \mathbf{m}, \mathbf{s})$
Models: Stable Diffusion (TextEncoder τ_1 and τ_2, VAE ϕ, U-Net υ), Fire-Decoder ψ_θ
Freeze VAE ϕ, U-Net υ, TextEncoder τ_1, τ_2
for each $((\mathcal{I}, \mathbf{m}, \mathbf{s}) \in \mathcal{D}$ **do**
  Encode image to latent: $\mathbf{x}_0 \leftarrow \phi(\mathcal{I})$
  Sample noise level: $t \sim \mathrm{Uniform}(\{1, \ldots, T\})$, $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
  if $t = 1$ **then**
    Add noise: $\mathbf{x}_t \leftarrow \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}$
    Encode prompt1: $\tau_1(\mathbf{s}) \leftarrow \mathrm{EncodePrompt}(\mathbf{s})$
    Encode prompt2: $\tau_2(\mathbf{s}) \leftarrow \mathrm{EncodePrompt}(\mathbf{s})$
    Concate prompt: $\tau(\mathbf{s}) \leftarrow \mathrm{Concate}(\tau_1(\mathbf{s}), \tau_1(\mathbf{s}))$
    Extract features: $\{F, A\} \leftarrow \upsilon(\mathbf{x}_t, t, \tau(\mathbf{s}))$
    Predict mask and class: $(\hat{\mathbf{y}}, \hat{\mathbf{m}}) \leftarrow \psi(F, A, \tau_1(\mathbf{s}))$
    $\mathcal{L}_{\mathrm{dice}} \leftarrow 1 - \frac{2\sum \hat{\mathbf{m}}\mathbf{m}}{\sum \hat{\mathbf{m}}^2 + \sum \mathbf{m}^2}$
    $\mathcal{L}_{\mathrm{bce}} \leftarrow -\frac{1}{N}\sum(\mathbf{m}\log(\hat{\mathbf{m}}) + (1-\mathbf{m})\log(1-\hat{\mathbf{m}}))$
    $\mathcal{L}_{\mathrm{ce}} \leftarrow -\sum \mathbf{y}\log(\hat{\mathbf{y}})$
    Calculate loss: $\mathcal{L} \leftarrow \lambda_1\mathcal{L}_{\mathrm{dice}} + \lambda_2\mathcal{L}_{\mathrm{bce}} + \lambda_3\mathcal{L}_{\mathrm{ce}}$
    Optimize θ using $\nabla_\theta\mathcal{L}$
  end if
end for
return Fire-Decoder ψ_θ

3.3.2. Training the Fire-Decoder

As we previously outlined, our Fire-Decoder is specifically engineered to learn segmentation conditioned on the average pooled intermediate representations $\tau_1(\mathbf{s})$ of the prompt. It utilizes multi-scale feature maps F and attention maps A extracted from the denoising U-Net. The attention maps A play a pivotal role, capturing the intricate interactions between text and image, which are crucial for class determination and mask localization. Concurrently, F includes image features across various resolutions, facilitating the detailed capture of the image's multi-scale nuances.

Before their introduction into the pixel decoder, a scale-wise fusion of cross-attention features A and multi-resolution image features F is performed. This fusion, depicted in the bottom left corner of Fig. 4a, scale-wisely concatenates these features. In parallel, to effectively learn the correlation between mask positions, their categories, and the textual information, the textual prompt representations $\tau_1(\mathbf{s})$ are augmented with a learnable text queue embeddings forming a soft prompt, as visualized in the top left corner of Fig. 4a.

These fused features then enter our pixel decoder, a key component of the Fire-Decoder, is specifically designed for executing scale-wise fusion. Scale-wise fusion is a technique that integrates feature maps from different resolutions, aiming to fully utilize the information present at various scales of an image, thereby enhancing the model's ability to understand global and local features of the image. This shares a similar goal with attention-based fusion methods [72, 73, 74], which dynamically select and integrate feature information from different sources or scales through attention mechanisms to heighten the model's sensitivity to key information. It is worth emphasizing that compared to attention-based fusion, scale-wise fusion demonstrates more significant applicability in fire segmentation tasks. This is because scale-wise fusion can integrate features from different levels, better adapting to the diverse scale variations of flames as they appear visually. In contrast, although attention-based fusion can highlight critical areas or specific scales within a scene, it may not be as comprehensive as scale-wise fusion when dealing with visual features like flames that exhibit a wide range of scale variations.

Within Fire-Decoder, we construct a fused feature pyramid that corresponds to the three initial resolution levels. Each scale level of these fused features is augmented with a sinusoidal positional embedding $e_{\text{pos}} \in \mathbb{R}^{H_l W_l \times C}$, suggested by [75], and a learnable scale-level embedding $e_{\text{lvl}} \in \mathbb{R}^{l \times C}$, following [76]. The initial input query of the Transformer decoder is the augmented textual prompt representation. This input sets the stage for the sequential processing within the Transformer decoder.

In terms of processing within the Transformer decoder, the three scale levels of fused features from the pyramid are fed into the Transformer decoder in ascending order of resolution, corresponding to their respective scale levels. This input sequence is repeated across the order of the decoder blocks. Specifically, the ordering of the fused features follows a cyclical pattern $(0, 1, 2, 0, 1, 2, ...)$ by the three different scales, aligning with the sequential order of the decoder blocks $(0, 1, 2, 3, 4, 5, ...)$. In each decoder block, these fused features are utilized primarily for cross-attention, with the main input being the output from the previous decoder block. As shown in Fig. 4b, the fused features correspond to the image features, and the output of the previous decoder blocks corresponds to the query features (except the first block whether we use the augmented textual prompt representation). This arrangement ensures that each decoder block effectively incorporates the relevant scale-level features for processing, thereby enabling the Fire-Decoder to handle the complexities of image segmentation with enhanced precision and efficacy.

Finally, the output of the decoder is utilized for generating class predictions and segmentation masks, as illustrated in the top right of Fig. 4a. Class predictions, are converted into a probability distribution for each pixel's class through a specialized class prediction head. In parallel, segmentation masks, are generated via a mask embedding head, which assigns the most probable class to each pixel based on these derived probabilities.

The computational process encompasses several crucial steps. Initially, a Hungarian Matcher, as proposed by [77], is employed to optimally match the outputs of the last layer with the given targets. This matcher operates on the principle of minimizing cost in a bipartite graph. Formally, let $\hat{Y} = \{\hat{y}_1, \hat{y}_2, ..., \hat{y}_N\}$ represent the set of N predicted labels, and $Y = \{y_1, y_2, ..., y_M\}$ the set of M ground truth labels. The matcher identifies a bijection $f : \hat{Y} \to Y$ that minimizes the total matching cost, where the cost function $C(\hat{y}, y)$ is defined for each predicted and ground truth label pair. This matching process is formulated as:

$$f^* = \underset{f}{\operatorname{argmin}} \sum_{\hat{y} \in \hat{Y}} C(\hat{y}, f(\hat{y})) \tag{10}$$

Once the matching is determined, the algorithm calculates the average number of target masks across all nodes for normalization purposes. This step is essential for maintaining a consistent scale of loss computation across different batch sizes. The number of masks is quantified as the sum of the lengths of target labels, expressed as $\sum_{i=1}^{M} |y_i|$. The cost function C comprises a negative log-likelihood approximation for classification and a combination of focal loss and dice loss for the segmentation masks. These losses are calculated based on the efficient point sampling of the predicted and ground truth masks.

The final objective of FireDM is a composite of these individual losses, therefore, serving as the basis for backpropagation to refine the model's accuracy in class prediction and mask generation. It is a combination of three components that correspond to the losses of matching cost: Classification cross-entropy loss ($\mathcal{L}_{\text{ce}}$), binary cross-entropy loss ($\mathcal{L}_{\text{bce}}$), and dice loss ($\mathcal{L}_{\text{dice}}$), combined as follows:

$$\mathcal{L} = \lambda_1 \mathcal{L}_{\text{dice}} + \lambda_2 \mathcal{L}_{\text{bce}} + \lambda_3 \mathcal{L}_{\text{ce}} \tag{11}$$

Each component of the loss is calculated based definitions:

$$\begin{aligned} \mathcal{L}_{\text{dice}} &= 1 - \frac{2 \sum \hat{\mathbf{m}} \mathbf{m}}{\sum \hat{\mathbf{m}}^2 + \sum \mathbf{m}^2} \\ \mathcal{L}_{\text{bce}} &= -\frac{1}{N} \sum \left(\mathbf{m} \log(\hat{\mathbf{m}}) + (1 - \mathbf{m}) \log(1 - \hat{\mathbf{m}}) \right) \\ \mathcal{L}_{\text{ce}} &= -\sum \mathbf{y} \log(\hat{\mathbf{y}}) \end{aligned} \tag{12}$$

In these equations, $\mathbf{m}$ represents the ground truth mask, $\hat{\mathbf{m}}$ the predicted mask, $\mathbf{y}$ the true class labels, and $\hat{\mathbf{y}}$ the predicted class labels. The λ_1, λ_2, and λ_3 are weighting factors for each loss component, which can be tuned during the training process to optimize model performance.

This comprehensive procedure ensures that each pixel in the output segmentation masks is accurately classified, contributing to the overall effectiveness and precision of the Fire-Decoder in image segmentation tasks.

3.4. The Inference Process of the FireDM

Fig. 2b presents the inference pipeline for Chatgpt-4-Fire guided data generation. This process includes two primary

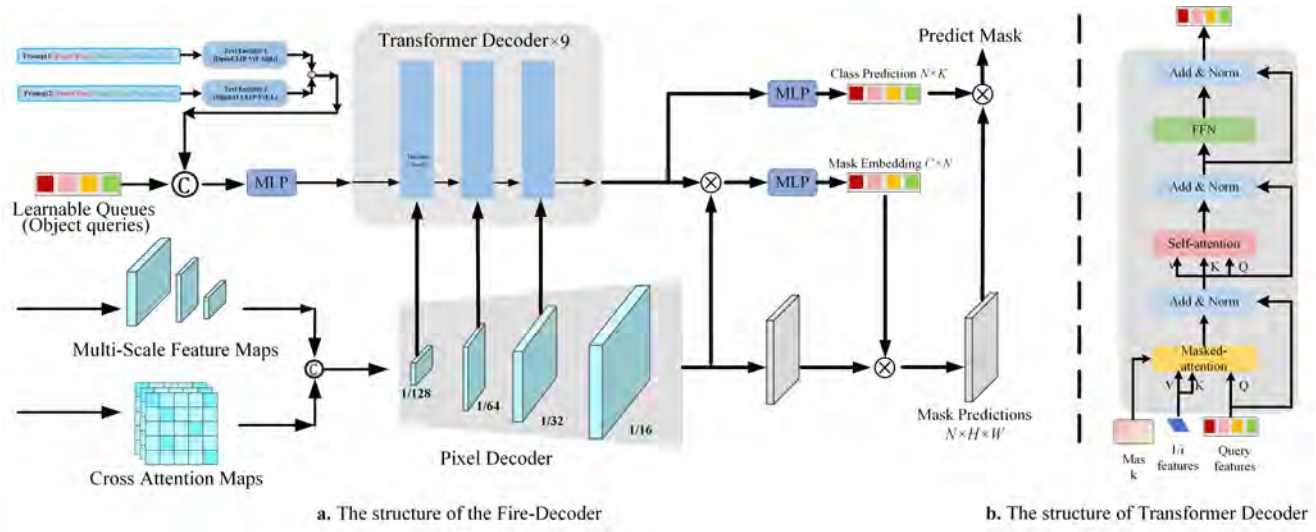

Figure 4: The Details of the Fire-Decoder.

modifications compared to the training phase, enhancing the precision and efficacy of the generated images.

Text-Guided Dataset Generation: A primary distinction lies in the source of the prompts. Unlike the training phase, where we use image descriptions as prompts, during inference, prompts for τ_1 and τ_2 are generated by ChatGPT4-Fire (section 3.2), a specialized adaptation of ChatGPT 4.0. Fig. A8 shows the detailed process of obtaining clear fire scene description statements through a short exchange with ChatGPT4-Fire. In the whole process, we only need to use no more than 50 words to guide ChatGPT4-Fire to give a clear and vivid statement describing the fire scene. Overall, the model has been fine-tuned specifically for expertise in fire-related events, allowing it to generate text summaries that are not only more accurate than human-generated text summaries, but also more professional. This targeted capability of ChatGPT-4-Fire is instrumental in guiding SDXL toward more precise and relevant image generation, particularly in the context of fire-related scenarios. During the inference process of FireDM, the text $\mathbf{s}$ is initially input into language models τ_1 and τ_2 to be transformed into embedding vectors, respectively. Subsequently, these embeddings are mapped to the U-Net layers through the multi-head attention mechanism Attention(Q, K, V) for further processing.

Denoising to Generate Images: Another significant alteration in the inference pipeline is the extension of the denoising process. Unlike in the training phase, the inference stage involves a denoising process that extends to the maximum steps specified by the noise scheduler. This adjustment allows for a controlled and refined diffusion process, resulting in high-quality synthetic images. Specifically, we input a random Gaussian matrix $\mathbf{x}_T$ along with the results from τ_1 and τ_2, then feed this into the pre-trained U-Net structure of SDXL 1.0 (base). The process involves 1000 steps of denoising, ultimately generating a realistic image $\hat{\mathcal{I}}$ from latent $\hat{\mathbf{x}_0}$.

Fire-Decoder Outputs Segmentation Results: Subsequent operations are similar to those described in Section 3.2.2 regarding the Fire-Decoder. Specifically, the multi-scale feature matrices generated during the multiple denoising steps of U-Net, along with the attention maps, are input into the trained Fire-Decoder to obtain the final segmentation masks that match the generated images.

The specifics of the inference process are detailed in the pseudocode provided in Algorithm 2. This pseudocode outlines the step-by-step procedure, reflecting the unique aspects of the ChatGPT4-Fire model's integration and the extended denoising sequence, both of which are pivotal for the effective generation of synthetic images in this specialized domain.

Algorithm 2: The Inference Process of FireDM

Input: Detailed prompt $\mathbf{s}$ generated by GPT-4
Models: Stable Diffusion (TextEncoder τ_1 and τ_2, VAE ϕ, U-Net υ), Well-trained Fire-Decoder ψ
Freeze all model parameters
Encode prompt1: $\tau_1(\mathbf{s}) \leftarrow$ EncodePrompt($\mathbf{s}$)
Encode prompt2: $\tau_2(\mathbf{s}) \leftarrow$ EncodePrompt($\mathbf{s}$)
Concate prompt: $\tau(\mathbf{s}) \leftarrow$ Concate$(\tau_1(\mathbf{s}), \tau_1(\mathbf{s}))$
Initialize random noise: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, T^2\mathbf{I})$
for $t = T$ **down to** 1 **do**
  if $t = 1$ **then**
    Extract image and features: $\hat{\mathbf{x}}_0, \{F, A\} \leftarrow \upsilon(\mathbf{x}_1, 1, \tau(\mathbf{s}))$
    Decode latent to image: $\hat{\mathcal{I}} \leftarrow \phi^{-1}(\hat{\mathbf{x}}_0)$
  else
    $\mathbf{x}_{t-1} \leftarrow \upsilon(\mathbf{x}_t, t, \tau(\mathbf{s}))$
  end if
end for
Predict mask: $(\hat{\mathbf{y}}, \hat{\mathbf{m}}) \leftarrow \psi(F, A, \tau_1(\mathbf{s}))$
return Synthetic image $\hat{\mathcal{I}}$ and mask $\hat{\mathbf{m}}$

4. FireDM Training and Validation Details

4.1. Performance Evaluation Metrics for FireDM

The ultimate goal of the FireDM algorithm is to generate a large-scale and high-quality fire segmentation dataset through weakly supervised learning methods. Therefore, the most intuitive criterion for evaluating FireDM's performance lies in assessing the quality of the segmentation dataset it produces. Consequently, determining the quality of this dataset becomes a key factor in evaluating FireDM's performance. In practice, the most direct approach to evaluating a dataset involves using it as a training set (experimental group) and comparing it with other segmentation datasets (baseline group) when training a segmentation detection algorithm that is both open-source and powerful. Subsequently, the performance of the trained segmentation algorithm is tested using the same testing set. If the dataset generated by FireDM significantly improves the segmentation accuracy of the algorithm, it can be demonstrated that the quality of this dataset is higher compared to other control group datasets, thereby highlighting the exceptional performance of FireDM. In this paper, we have chosen the advanced semantic segmentation algorithm, Mask2Former [35]. The segmentation evaluation metrics utilized in this paper include Intersection over Union (IoU), Pixel Accuracy (PA), Recall, F1-Score, and Average Precision (AP). To highlight the image fidelity and perceptual quality of the generated images in terms of the images themselves, we use the evaluation metrics: Structural Similarity Index (SSIM) [78], Peak Signal-to-Noise Ratio (PSNR), Fréchet Inception Distance (FID), Learned Perceptual Image Patch Similarity (LPIPS), and Perceptual Image-Error Assessment through Pair Ranking (PieAPP) [79].

4.2. The Dataset Used in Experiment

4.2.1. Training dataset for FireDM

Unless specifically stated otherwise in the experiments, we default to using the BoWFire dataset [36] as the training set for FireDM, which contains only 119 images, and the distribution of scene categories in this dataset is shown in Table. 1. From the scene distribution in the table, we can observe that the occurrence of fire scenes in this training dataset is singular, and there is a significant lack of scenes such as forests and vehicles. However, these characteristics precisely align with the background design and requirements of the Weakly Supervised Learning method. In the case of a severe deficiency in the training dataset, the quality of the subsequently generated fire segmentation dataset will be directly determined by the superior framework of FireDM.

4.2.2. Test dataset for segmentation algorithms

The FireDM algorithm proposed in this paper aims to generate an extensive fire segmentation dataset for application to real-world fire segmentation scenarios. In some subsequent experiments, this dataset will be used as a training set for a specific fire segmentation algorithm (Mask2Former [35]) in order to facilitate the verification of the quality of the dataset generated by FireDM. The test dataset was downloaded from different search engines using different keywords. But unfortunately, most of the datasets here lack the corresponding segmentation masks. Therefore, we manually labelled the masks for these image sets in a painstaking process. These images were collected from real fire scenes. The test set consists of 6000 images, mainly representing various scenarios: vehicle fires, structure fires, forest fires, ship fires, industrial fires, etc. We manually labelled all the images with a mask. We manually labelled all images with pixel masks. The minimum size of the images is 512×512 pixels. To ensure the authenticity of our collected test datasets, we have made them publicly available on the project's github page.

Table 1
The distribution of the dataset used for training FireDM.

	Vehicle Fire	Structure Fire	Forest Fire	Total
Number	16	103	0	119

4.3. Training Protocols

The text-to-image generative pre-trained model (Diffusion model) we used is the SDXL 1.0 (base) and SD 2.1. We choose Mask2Former as the infrastructure of Fire-Decoder and optimise it accordingly. For all tasks, we trained FireDM for approximately 5000 iterations. FireDM can successfully generate images in three different sizes: 512×512, 768×768, and 1024×1024, along with their corresponding segmentation masks. The training was conducted using a single NVIDIA A800 80GB Tensor Core GPU. We utilized an optimizer with a learning rate of 0.0001.

4.4. Text Prompt

In the subsequent experiments of this paper, image diffusion relies on targeted text guidance. Therefore, the format and structure of the text are crucial. This section provides the corresponding formats for five scene-related text prompts that may appear in the subsequent experiments. The specific sentence format is shown in Table. A16 of the Appendix A.

5. Results and Analysis

5.1. FireDM Generates High-quality Fire Segmentation Datasets

This section focuses on evaluating the quality of the dataset generated after training FireDM using the BoWFire dataset [36]. The datasets generated at three different image resolutions are indicated in Table. 2. Examples of the generated images are shown in Fig. 5. Upon close inspection of images at three different resolutions in Fig. 5, we find their overall image quality to be very close to that of actual fire scene images. Equally noteworthy is that the pink colour mask annotations corresponding to each image almost accurately mark the areas with flames in the images. A more

Figure 5: The visualization of images and their corresponding masks in the dataset. Specifically, the entire dataset includes three different image sizes. The sub-images in this figure appear in pairs, where the first image is solely of the fire scene, and the second image is a composite of the fire scene and its corresponding mask, with the flame areas highlighted in pink colour for visualization.

Table 2
Three datasets of three image sizes generated by FireDM. Each dataset contains three fire scenes.

Dataset	Image Size	Vehicle Fire	Structure Fire	Forest Fire	Total
FireDM-512	512×512	4572	4658	4720	13950
FireDM-768	768×768	4971	4987	4968	14926
FireDM-1024	1024×1024	4988	4994	4985	14967
FireDM-512+768+1024	-	14531	14639	14673	43843

detailed display of this dataset is presented in Appendix B, as shown in Fig. B9.

In this study, we conducted an initial evaluation of the three datasets listed in Table. 2 using the Mask2Former segmentation algorithm, with the results presented in Table. 3. The baseline datasets, specifically Experiments 1, 5, 9, and 13, correspond to the three publicly available datasets mentioned in Section 2.1 and the combination of these three datasets. In Experiments 1-16, to ensure the number of images in the datasets used for the experiments was consistent, we randomly selected 119, 1,135, 2,003, and 3,257 images from the three datasets listed in Table. 2, respectively. The results indicate that, with an equal number of training images, the quality of the datasets generated by FireDM is close to or even surpasses that of real datasets.

It's worth emphasizing that Experiments 17-19 in Table. 3 correspond to the three datasets listed in Table. 2, and Experiment 20 represents a larger dataset merged from these three datasets. We then compared Experiments 17-20 with Experiments 1-16 to demonstrate the practical application value of FireDM. This shows that existing publicly available fire segmentation datasets are insufficient for segmentation algorithms to achieve optimal performance. In contrast, FireDM is capable of generating datasets much larger than the existing public datasets at a very low cost, thereby significantly improving the accuracy of segmentation algorithms. This finding underscores the importance and utility of FireDM in the field of fire scene segmentation.

5.2. Ablation Experiments on FireDM Structures

5.2.1. Varying degrees of textual guidance during the diffusion process

In terms of structure, the ability of FireDM to widely generalize fire datasets across various scenes is primarily attributed to the clear and articulate textual guidance provided by ChatGPT4-Fire. In this section, our aim is to further validate through comparative experiments whether there exists a relationship between the quality of the datasets generated through diffusion in FireDM and factors such as the type of large language models providing text, the number of sentences in text prompts, or the word count per sentence. The specific results of the experiment are shown in Table. 4, Table. 5 and Table. 6.

In the experiments presented in Table. 4, we only varied the type of large language model that provides language descriptions, while keeping other factors constant. Ultimately, the generated datasets of three different image sizes were used for training Mask2Former. Subsequently, we evaluated the segmentation accuracy of the trained model on the test dataset (Section 4.2.2). Similarly, in Table. 5, we only varied the number of words in each sentence output by ChatGPT4-Fire, and in Table. 6, we only varied the number of sentences in the descriptions output by ChatGPT4-Fire, as the dataset generated by FireDM comprises 3 scenarios, the number of sentences for each scenario is evenly distributed. For instance, if a total of 30 prompt sentences are chosen, then each scenario-related prompt would have 10 sentences.

Observing Table. 4, we find that the text expression capabilities of large language models, led by ChatGPT-4.0 and Claude2.0, have far surpassed those of ordinary humans. As these large language models undergo updates and iterations, the corresponding text expression capabilities are further strengthened. This is beneficial for generating clearer text prompts, resulting in higher-quality generation and more accurate mask generation for segmentation datasets. As ChatGPT4-Fire is built upon ChatGPT-4 to serve as a firefighting expert with a deeper understanding of fire-related information, its text expression capabilities are stronger compared to ChatGPT-4.

Table 3
The dataset generated by FireDM after training using only the BoWFire dataset is compared to three other publicly available fire segmentation datasets.

Dataset	Real Image	Synthetic	Image Size	IoU (%)	Pixel Accuracy (%)	F1-Score (%)	Average Precision (%)
1. Baseline-BoWFire	119	0	-	54.12	56.71	53.26	55.45
2. FireDM-512-B	0	119	512×512	54.15 (↑ 0.03)	56.54 (↓ 0.17)	53.36 (↑ 0.10)	55.49 (↑ 0.04)
3. FireDM-768-B	0	119	768×768	54.35 (↑ 0.23)	56.92 (↑ 0.21)	53.58 (↑ 0.32)	55.87 (↑ 0.42)
4. FireDM-1024-B	0	119	1024×1024	54.94 (↑ 0.82)	57.26 (↑ 0.50)	53.91 (↑ 0.65)	56.02 (↑ 0.57)
5. Baseline-Corsican	1135	0	-	61.51	63.13	60.62	62.86
6. FireDM-512-C	0	1135	512×512	61.75 (↑ 0.24)	62.91 (↓ 0.22)	60.93 (↑ 0.31)	63.12 (↑ 0.26)
7. FireDM-768-C	0	1135	768×768	62.12 (↑ 0.61)	63.78 (↑ 0.65)	61.33 (↑ 0.71)	63.87 (↑ 1.01)
8. FireDM-1024-C	0	1135	1024×1024	62.87 (↑ 1.36)	64.31 (↑ 1.18)	61.76 (↑ 1.14)	64.12 (↑ 1.26)
9. Baseline-FLAME	2003	0	3480×2160	66.33	67.12	65.57	66.60
10. FireDM-512-F	0	2003	512×512	66.91 (↑ 0.58)	67.47 (↑ 0.35)	65.28 (↓ 0.29)	66.93 (↑ 0.33)
11. FireDM-768-F	0	2003	768×768	67.05 (↑ 0.72)	67.77 (↑ 0.65)	66.12 (↑ 0.55)	67.20 (↑ 0.60)
12. FireDM-1024-F	0	2003	1024×1024	67.43 (↑ 1.10)	68.12 (↑ 1.00)	66.38 (↑ 0.81)	67.51 (↑ 0.91)
13. Baseline-BoWFire+Corsican+FLAME	3257	0	-	71.15	72.08	69.93	70.14
14. FireDM-512-BCF	0	3257	512 × 512	71.23 (↑ 0.08)	71.98 (↓ 0.10)	69.85 (↓ 0.08)	70.20 (↑ 0.06)
15. FireDM-768-BCF	0	3257	768 × 768	71.59 (↑ 0.44)	72.35 (↑ 0.27)	70.34 (↑ 0.41)	70.52 (↑ 0.38)
16. FireDM-1024-BCF	0	3257	1024 × 1024	71.88 (↑ 0.73)	72.71 (↑ 0.63)	70.85 (↑ 0.92)	71.21 (↑ 1.07)
17. FireDM-512	0	13950	512 × 512	73.10	74.09	71.83	72.45
18. FireDM-768	0	14926	768 × 768	74.82	75.29	73.99	73.78
19. FireDM-1024	0	14967	1024 × 1024	75.40	75.76	74.45	74.11
20. FireDM-512+768+1024	0	43843	-	77.82	77.65	76.76	76.45

Observing Table. 5, we observe a significant improvement in all four evaluation metrics as the number of words per prompt sentence increases from 5 to 20. However, further increasing the word count from 20 to 40 reveals a very subtle rise or fall in all four evaluation metrics. This phenomenon suggests that the number of words per prompt sentence indeed correlates with the performance of FireDM within the range of 0 to 20 words, with 20 being nearly the optimal threshold for FireDM's performance.

Observing Table. 6, we observe a significant improvement in all four evaluation metrics as the number of prompt sentences increases from 10 to 30. However, further increasing the number of sentences from 30 to 100 reveals a very subtle rise or fall in all four evaluation metrics. This phenomenon indicates that the number of prompt sentences does indeed correlate with FireDM's performance within the range of 10 to 30 sentences, with 30 being nearly the optimal threshold for FireDM's performance. redIt is worth noting that since the diffusion model has a certain degree of randomness at each step of the diffusion process, even the same descriptive statement can cause FireDM to generate image scenes that do not violate the descriptive statement but are completely different. Thus, even as few as 30 text statements can lead FireDM to generate countless image scenes.

We believe the reason behind the experimental results in Table. 5 and Table. 6 primarily lies in the Text-Encoder applied in FireDM, which has the optimal capacity for extracting textual information, and the subsequent structure of the Fire-Decoder also possesses the optimal ability to comprehend textual information. We also observed from Table. 4, Table. 5 and Table. 6, with other parameters held constant, the performance of Mask2Former improves as the dataset size approaches 1024×1024. This is primarily because larger segmentation dataset sizes exhibit more effective information, which is beneficial for the training of segmentation algorithms. However, it does not imply that pursuing excessively larger dataset sizes based on image dimensions is always favorable, as excessively high image sizes may lead to undesirable outcomes. We will present further evidence of this in the upcoming Section 5.4 experiments.

5.2.2. *Varying degrees of textual guidance during the fire-decoder*

FireDM not only utilizes textual information during the diffusion process but also incorporates text embeddings at the Fire-Decoder stage to enhance its ability to accurately perceive segmentation masks. Therefore, this section aims to validate the effectiveness of text embedding in the Fire-Decoder stage and to determine if there is a correlation between the length of sentences and the accuracy of the induced segmentation masks for each image. The final experimental results are presented in Table. 7. Analyzing Table. 7, we observed that applying textual guidance within the Fire-Decoder, even with guidance sentences only 5 words long, significantly improves the quality of the generated dataset compared to when textual guidance is not applied. Moreover, as the length of the guidance sentences increases, the effect of textual guidance gradually improves until the sentence length reaches about 10 words. Beyond this length, further increasing the sentence length does not enhance the effect of textual guidance. This indicates the existence of an optimal sentence length; exceeding this length, additional textual information no longer significantly affects the performance enhancement of the Fire-Decoder. We believe this is mainly because the Text-Encoder applied in the Fire-Decoder has an optimal capacity for extracting textual information, and the subsequent structure of the Fire-Decoder also has an optimal ability to understand textual information. Additionally, 10 words are already sufficient to express the important information of an image.

5.2.3. *Varying sizes of training datasets*

In this section, we will explore the impact of larger and more diverse datasets on the training of FireDM. We initially used the BoWFire dataset as the base training set. It is worth mentioning that the BoWFire dataset contains only 119 images. Therefore, to build a larger and more diverse training set, we proceeded to select 119 images and their

Table 4
The quality of the dataset generated by FireDM is influenced by the textual descriptions of flame scenes provided by different large language models.

Name	Version	IoU (%)			Pixel Accuracy (%)			F1-Score (%)			Average Precision (%)		
		512 × 512	768 × 768	1024 × 1024	512 × 512	768 × 768	1024 × 1024	512 × 512	768 × 768	1024 × 1024	512 × 512	768 × 768	1024 × 1024
Human	-	87.5	88.6	89.3	88.1	89.3	90.4	86.4	87.3	88.9	83.4	84.7	85.2
ChatGPT	3.5	88.3	89.1	90.1	89.0	89.6	91.1	86.9	87.8	89.2	84.0	85.1	86.4
	4.0	88.6	89.4	90.3	89.3	89.9	91.5	87.4	88.2	89.5	84.2	85.3	86.7
	4.0-Fire (**Ours**)	89.1(↑ 1.6)	89.8 (↑ 1.2)	90.8(↑ 1.5)	90.0(↑ 1.9)	90.2 (↑ 0.9)	91.9(↑ 1.5)	87.9(↑ 1.5)	88.7(↑ 1.4)	90.3(↑ 1.4)	85.1(↑ 1.7)	86.0 (↑ 1.3)	87.2(↑ 2.0)
Claude	2.0	88.2	89.0	90.2	88.8	89.4	90.8	87.1	87.6	89.0	83.8	85.3	86.7
	2.1	88.4	89.5	90.4	89.2	89.6	91.4	87.3	88.4	89.3	84.4	85.5	86.6

Table 5
The impact of the number of text words used to prompt FireDM on the quality of the generated dataset.

Name	Number	IoU (%)			Pixel Accuracy (%)			F1-Score (%)			Average Precision (%)		
		512 × 512	768 × 768	1024 × 1024	512 × 512	768 × 768	1024 × 1024	512 × 512	768 × 768	1024 × 1024	512 × 512	768 × 768	1024 × 1024
	5	75.3	75.5	75.7	76.2	76.4	76.7	77.2	77.4	77.6	78.1	78.4	78.2
	10	88.3	88.1	89.4	89.0	89.3	90.2	87.0	87.6	88.2	82.7	83.8	84.9
	15	88.6	88.4	89.8	89.5	89.7	90.6	87.5	88.2	89.7	84.8	85.5	86.5
	20 (**Ours**)	**89.1**	**89.8**	**90.8**	**90.0**	**90.2**	**91.9**	**87.9**	**88.7**	**90.3**	**85.1**	**86.0**	**87.2**
ChatGPT4-Fire	25	88.8	89.6	90.6	89.8	90.0	91.7	87.6	88.5	90.1	85.0	85.8	87.0
	30	88.3	89.1	90.0	89.3	89.6	91.4	87.2	87.4	89.6	84.6	85.3	86.3
	35	87.7	87.6	89.6	89.0	89.0	90.3	85.9	87.0	88.9	84.0	84.7	85.2
	40	86.3	86.9	89.4	88.7	88.5	89.7	85.3	86.5	88.2	83.6	83.8	84.3

corresponding masks from two other datasets to expand the training set and used these for training FireDM. This process was repeated, continuing to extract more images and their masks from these two datasets to further enlarge the training set. Through this method, we were able to obtain datasets with numbers of images being 119, 119+119×2, 119+119×2×2, and 119+119×2×2×2, respectively. Subsequently, we used these datasets to train FireDM separately. Then, we used the trained FireDM to generate datasets for training segmentation models. The final evaluation of the experimental results was based on the performance of these segmentation models.

The experimental results, as summarized in Table. 8, show that as the size and diversity of the dataset increased, the datasets generated by the trained FireDM could train stronger segmentation models, thereby enhancing the generalization ability and robustness of these models. However, once the number of images in the dataset used to train FireDM reached 833, we observed that further increasing the size of the training set did not significantly improve the quality of the datasets generated by FireDM.

5.2.4. Varying varieties of Diffusion Models

FireDM utilizes SDXL 1.0 and SD 2.1 as its foundational diffusion models. However, whether they are the optimal choices for FireDM is a key question explored in this section. To validate this question, we conduct our analysis from two perspectives. First, we need to verify whether the image quality generated by the Stable Diffusion series surpasses that of other common generative algorithms. It is worth noting that since the Stable Diffusion series consists of pre-trained diffusion models, for fairness, we also train the other generative algorithms using the same dataset as used for Stable Diffusion 1.4. Then, we evaluate the quality of the generated images using four metrics, with the final experimental results presented in Table. 9. According to this table, we find that the image quality produced by the Stable Diffusion series significantly exceeds that of both the DDPM series and GAN series algorithms. Moreover, as the version

Table 6
The impact of the number of sentences used to prompt FireDM on the quality of the generated dataset.

Name	Number	IoU (%)			Pixel Accuracy (%)			F1-Score (%)			Average Precision (%)		
		512 × 512	768 × 768	1024 × 1024	512 × 512	768 × 768	1024 × 1024	512 × 512	768 × 768	1024 × 1024	512 × 512	768 × 768	1024 × 1024
	10	83.2	84.1	86.3	83.5	85.5	86.1	84.2	84.6	84.8	81.7	82.2	82.7
	20	87.5	87.6	88.5	88.1	90.2	88.5	85.6	86.4	87.5	84.3	84.6	85.3
	30 (**Ours**)	**89.1**	**89.8**	**90.8**	**90.0**	**90.2**	**91.9**	**87.9**	**88.7**	**90.3**	**85.1**	**86.0**	**87.2**
ChatGPT4-Fire	40	88.9	89.5	90.7	89.7	90.1	90.6	87.7	88.4	90.0	85.1	85.9	86.9
	50	88.4	89.2	90.3	88.5	89.6	90.1	87.2	88.2	89.3	83.2	84.4	86.3
	60	87.0	87.6	89.6	87.0	88.2	89.7	86.7	87.9	88.3	82.3	83.0	85.0
	100	86.1	86.7	88.8	86.1	87.2	88.5	85.6	85.8	87.0	81.2	82.9	83.4

Table 7
The impact of the number of text words used to prompt Fire-Decoder on the quality of the generated dataset. When Number is equal to 0, it means that the Fire-Decoder has not embedded the text prompt in the learnable queue.

Name	Number	IoU (%)			Pixel Accuracy (%)			F1-Score (%)			Average Precision (%)		
		512 × 512	768 × 768	1024 × 1024	512 × 512	768 × 768	1024 × 1024	512 × 512	768 × 768	1024 × 1024	512 × 512	768 × 768	1024 × 1024
	0	75.7	75.8	76.2	77.3	78.3	78.7	77.7	76.9	78.2	77.9	78.3	76.9
	5	85.4	85.3	85.5	86.4	86.3	86.3	84.5	85.2	85.7	83.5	83.9	84.3
ChatGPT4-Fire	10 (**Ours**)	**89.1**	**89.8**	**90.8**	**90.0**	**90.2**	**91.9**	**87.9**	**88.7**	**90.3**	**85.1**	**86.0**	**87.2**
	15	88.9	89.9	90.7	90.1	90.1	92.0	88.0	88.8	90.1	85.0	86.1	87.3
	20	89.2	89.7	90.8	89.9	90.1	91.8	87.8	88.6	90.1	85.1	86.0	87.2

Table 8
Impact of larger datasets with more diverse scenes on the performance of FireDM-generated datasets.

Dataset	Number	IoU (%)			Pixel Accuracy (%)			F1-Score (%)			Average Precision (%)		
		512×512	768×768	1024×1024	512×512	768×768	1024×1024	512×512	768×768	1024×1024	512×512	768×768	1024×1024
BoWFire	119 × 0.5 (≈ 60)	83.2	83.4	83.8	85.2	85.3	85.6	82.6	82.8	83.1	82.3	82.5	82.7
BoWFire	119 × 1 (119) (**Ours**)	89.1	89.8	90.8	90.0	90.2	91.9	87.9	88.7	90.3	85.1	86.0	87.2
+CorsicanFire+FLAME	119 × 3 (357)	91.7	90.2	92.1	91.0	92.0	92.4	88.4	89.3	90.6	85.5	86.5	87.6
	119 × 5 (595)	92.6	91.1	92.6	91.5	92.3	92.6	89.5	90.2	91.3	86.9	87.6	88.9
	119 × 7 (833)	93.2	91.6	92.8	91.7	92.6	92.9	89.6	90.5	91.7	87.2	88.1	89.2
	119 × 9 (1071)	93.3	91.5	92.9	91.8	92.4	93.1	89.5	90.6	91.8	87.3	88.0	89.3

of Stable Diffusion improves, the quality of the generated images also progressively increases.

Secondly, we delve further into the compatibility of the Stable Diffusion series algorithms within the FireDM architecture by applying them as the foundational infrastructure for FireDM. It is worth noting that each diffusion model architecture comes with its default output image size. For instance, the default output image size for SDXL 1.0 is 1024×1024, while for SD 2.1, it is 512×512 and 768×768. Since FireDM incorporates both SDXL 1.0 and SD 2.1, it is capable of generating datasets with three different image sizes. However, when using SDXL 1.0 alone, it can only produce images with a size of 1024×1024. Observing Table. 10, we find that with the upgrade of Stable Diffusion versions, all four evaluation metrics show improvement. In FireDM, using SD 2.1 as the foundational architecture produces datasets of the highest quality for both 512×512 and 768×768 image sizes. Consequently, FireDM opts for SD 2.1 as one of the foundational architectures for the diffusion model to generate datasets of these two image sizes. Compared to SDXL 0.9 (base), the datasets generated by FireDM using SDXL 1.0 (base) for the 1024×1024 image sizes exhibit higher quality but are slightly inferior to those generated using SDXL 0.9 (base+refiner) and SDXL 1.0 (base+refiner) as architectures. This is primarily attributed to the fact that the parameter count for the base+refiner is nearly double that of the base, leading to an approximate twofold increase in FireDM's training time. Considering the trade-off between quality and efficiency, we assert that SDXL 1.0 (base) outperforms other XL series algorithms in overall performance. Therefore, we choose SDXL 1.0 as one of the foundational architectures for the diffusion model to generate datasets of the 1024×1024 image size. In summary, the combination of SD 2.1 and SDXL 1.0 as the foundational architectures for FireDM is the most prudent choice as it helps FireDM generate datasets of the highest quality for all three image sizes.

5.2.5. Varying number of Diffusion Time Steps

In this section, we will further explore whether the number of steps for adding noise to Latent during the FireDM training process has an impact on the quality of the generated datasets. By observing Table. 11, we find that the performance of FireDM steadily improves as the step count increases from 1 to 50. However, further increasing the step count from 50 to 800 results in a decline in FireDM's performance. This indicates that a diffusion step count of 50 is nearly the optimal threshold for FireDM's performance.

Table 9
Comparison of image quality between stable diffusion series algorithms and other common generative algorithms. The symbol ↑ indicates that a higher value of this evaluation metric is better. The symbol ↓ indicates that a lower value of this metric is better.

Dataset	Image Size	SSIM (↑)	PSNR (↑)	FID (↓)	LPIPS (↓)	PieAPP (↓)
GAN [80]	512×512	0.39	20.5	6.1	0.95	0.78
PAGGAN [81]	512×512	0.46	21.3	5.6	0.94	0.71
StyleGAN [23]	512×512	0.48	22.1	5.4	0.92	0.64
StyleGAN2 [24]	512×512	0.51	23.8	5.5	0.93	0.66
DDPM [60]	512×512	0.47	23.6	5.3	0.91	0.63
DDIM [61]	512×512	0.54	24.1	5.1	0.88	0.58
Stable Diffusion 1.4	512×512	0.65	24.7	4.8	0.86	0.54
Stable Diffusion 1.5	512×512	0.68	25.1	4.6	0.78	0.52
Stable Diffusion 2.0	512×512	0.69	25.2	4.7	0.73	0.46
Stable Diffusion 2.0	768×768	0.70	25.4	4.5	0.69	0.44
Stable Diffusion 2.1 (Ours)	512×512	0.71	25.4	4.5	0.68	0.42
Stable Diffusion 2.1 (Ours)	768×768	0.73	25.6	4.3	0.64	0.40
Stable Diffusion XL 0.9 (base)	1024×1024	0.74	25.7	4.3	0.63	0.36
Stable Diffusion XL 0.9 (base+refiner)	1024×1024	0.76	25.9	4.1	0.58	0.33
Stable Diffusion XL 1.0 (Ours, base)	1024×1024	0.76	25.8	4.0	0.56	0.34
Stable Diffusion XL 1.0 (base+refiner)	1024×1024	0.79	26.2	3.8	0.47	0.31

5.3. Generalization Validation of FireDM

In this section, we focus on exploring the cross-scene generalisation capabilities of FireDM. Specifically, we train FireDM using a limited dataset consisting of only 103 building fire images and their corresponding segmentation masks (see Section 4.2.1 for details). In the inference phase, we input prompts related to different scenarios such as forests, vehicles, ships, industries and mines, and then had FireDM generate three datasets with different image sizes for each scenario, resulting in a total of 15 fire segmentation datasets with 500 images each. Subsequently, these datasets were evaluated using Mask2Former. It's important to highlight that the test datasets employed in this section comprise 300 images and their corresponding segmentation masks, which were randomly selected from the test dataset detailed in Section 4.2.2. Examples of the textual prompt formats tailored for the five scenarios addressed in this experiment can be found in rows 2a-6c of Table. A16.

Furthermore, we collected high-quality fire images corresponding to the five scenes through keyword searches, with each scene including three sizes of 500 images each, to create benchmark datasets for comparison with the datasets generated by FireDM. However, due to the lack of segmentation masks for the majority of these images, we had to undertake labor-intensive manual pixel-level annotation. It should be emphasized that these benchmark datasets are not included in the test dataset mentioned in Section 4.2.2. Meanwhile, in order to show the powerful generalisation ability of FireDM algorithm more comprehensively, we introduce three most popular data enhancement algorithms DiffuMask [82], DatasetDM [32] and Text2Image [32]. citepge2023beyond. The final results of these experiments

Table 10
The impact of different types of pre-trained diffusion models on the quality of the FireDM generated dataset. The Train Time refers to the time taken for FireDM training, not the training time for the Mask2Former dataset.

Name	Version	Optimum Size	IoU (%)			Pixel Accuracy (%)			F1-Score (%)			Average Precision (%)			Train
			512^2	768^2	1024^2	512^2	768^2	1024^2	512^2	768^2	1024^2	512^2	768^2	1024^2	Time (h)
	1.4	512×512	85.5	-	-	85.6	-	-	83.4	-	-	80.9	-	-	17
	1.5	512×512	86.3	-	-	86.2	-	-	84.8	-	-	82.2	-	-	20
	2.0	512^2, 768^2	86.4	87.2	-	86.9	87.5	-	86.0	86.7	-	82.5	83.1	-	23
Stable Diffusion	2.1(**Ours**)	512^2, 768^2	**86.6**	**87.4**	-	**87.3**	**87.9**	-	**86.1**	**87.0**	-	**82.9**	**83.4**	-	23
	XL 0.9 (base)	1024×1024	-	-	90.3	-	-	91.3	-	-	90.0	-	-	86.7	32
	XL 0.9 (base +refiner)	1024×1024	-	-	90.8	-	-	92.0	-	-	90.4	-	-	87.5	59
	XL 1.0 (base, **Ours**)	1024×1024	-	-	90.8	-	-	91.9	-	-	90.3	-	-	87.2	35
	XL 1.0 (base +refiner)	1024×1024	-	-	90.9	-	-	92.1	-	-	90.5	-	-	87.6	61

Table 11
The impact of different diffusion time steps on the performance of FireDM.

Name	Diffusion Time Steps	IoU (%)			Pixel Accuracy (%)			F1-Score (%)			Average Precision (%)		
		512×512	768×768	1024×1024	512×512	768×768	1024×1024	512×512	768×768	1024×1024	512×512	768×768	1024×1024
	1	88.9	89.5	90.5	89.7	89.8	91.6	87.5	88.4	90.0	84.8	85.8	86.8
	10	89.0	89.7	90.7	89.8	90.0	91.7	87.7	88.5	90.1	85.0	86.0	87.0
Stable Diffusion	50 (**Ours**)	**89.1**	**89.8**	**90.8**	**90.0**	**90.2**	**91.9**	**87.9**	**88.7**	**90.3**	**85.1**	**86.2**	**87.2**
XL 1.0 (base)	100	88.7	89.3	90.2	89.6	89.9	91.4	87.3	88.1	89.8	84.5	85.5	86.3
	300	88.4	89.0	90.0	89.3	89.5	91.0	86.9	87.8	89.5	84.3	85.3	86.1
	800	87.6	88.6	89.8	89.0	89.2	90.5	86.3	87.4	89.0	84.0	85.0	85.8

are summarized in Table. 12, showcasing FireDM's exceptional performance in fire segmentation. Fig. 6 presents examples from fifteen datasets generated by FireDM. Judging from the overall quality of these sample images, the generated datasets closely resemble real datasets in terms of quality. It is also noteworthy that the pink mask annotations for each image accurately mark the areas with flames in the pictures. A more detailed display of these datasets is shown in the Appendix B section of Fig. B10, Fig. B11 and Fig. B12.

Observing Table. 12, We note that the test results obtained after training Mask2Former with the FireDM generated dataset in the five scenarios are close to or even exceed the results obtained after training with the baseline dataset, which indicates that the quality of the FireDM generated dataset is close to or even exceeds that of the baseline dataset. Meanwhile, the quality of dataset generated by FireDM exceeds the quality of dataset generated by the other three data enhancement algorithms in all scenarios. The above experimental results demonstrate the powerful generalisation ability of FireDM.

5.4. Wide Applicability Validation of FireDM

In Section 5.3, we created 15 separate datasets for the experimental and control groups. In this section, we will internally merge the datasets for the experimental and control groups from Section 5.3. For each group of 5 datasets with the same size but different scenarios, we will merge them into a new dataset. For example, in the experimental group, there are 5 datasets with image size 512×512 and each represents 1 scene respectively, we merge these 5 datasets into a new dataset named 512×512. the number of images in this dataset is 2500. Similarly, the two datasets 768×768 and 1024×1024 are merged in this way.

In this section, our main goal is to empirically demonstrate that FireDM outperforms all existing weakly supervised data enhancement methods such as DiffuMask [82], DatasetDM [32] and Text2Image [31]. Therefore, we conducted a comparative analysis of image fidelity and perceived quality of the images generated by these three algorithms and the algorithm proposed in this study, and the specific experimental results are displayed in Table. 13. Based on the final experimental data in this table, we observe that the quality of the three resolution images generated by the algorithm in this study generally surpasses the image quality found in datasets generated by the three algorithms, DiffuMask, DatasetDM, and Text2Image.

Furthermore, we would like to further verify the superiority of FireDM in generating datasets with multiple image resolutions and whether it can be broadly applied to various segmentation algorithms. It is worth noting that the performance of the other data enhancement algorithms used in our experiments will be debugged to their best in our experiments, while using textual bootstrapping statements that are all consistent with FireDM. The final experimental results are detailed in Table. 14.

By analyzing the data in Table. 14, we observed that datasets generated by FireDM at three different image resolutions have a significant positive impact on the training of segmentation algorithms, far surpassing the results obtained using the baseline dataset. In contrast, datasets produced by the other three data augmentation methods fell short in enhancing the training performance of segmentation algorithms, generally exhibiting results below those of the baseline dataset. To delve deeper into the cause of this phenomenon, we further analyze the structural details of three comparative algorithms and FireDM. First, from the perspective of image quality, the other three algorithms rely on the pretrained SD 1.4, whereas our FireDM utilizes SD 2.1 and SDXL 1.0, which are pretrained on a larger dataset and boast a more superior and advanced structure. This results in FireDM producing images of quality that comprehensively surpasses those generated by the other three algorithms, a point that is corroborated by the results in Table. 13. It is well-known that at the same image resolution, higher

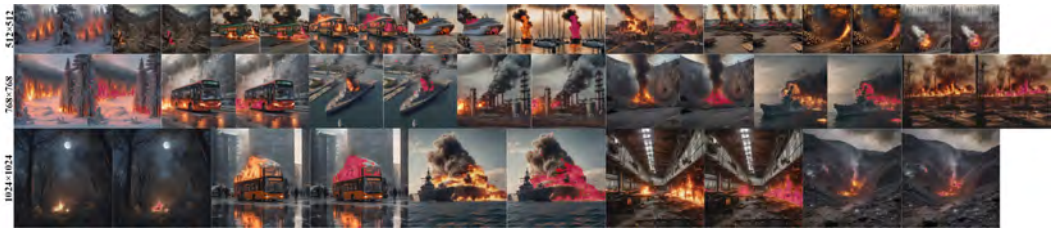

Figure 6: The visualization results of the 15 datasets generated by FireDM for five scenes (each scene has three image sizes). Each size in the figure includes five scenes of its own, and the order from left to right is: Forest, Vehicle, Ship, Industrial, Mine.

Table 12
The quality of datasets generated by FireDM for various scenarios. Specifically, we train FireDM using the building fire incidents portion of the BoWFire dataset, then utilize FireDM to generate fire segmentation datasets for various other scenarios and assess their quality.

Scene	Method	IoU (%)			Pixel Accuracy (%)			F1-Score (%)			Average Precision (%)		
		512×512	768×768	1024×1024	512×512	768×768	1024×1024	512×512	768×768	1024×1024	512×512	768×768	1024×1024
Forest	DiffuMask	70.5 (↓ 0.6)	-	-	71.6 (↓ 0.4)	-	-	69.9 (↓ 0.2)	-	-	69.8 (↓ 0.5)	-	-
	DatasetDM	70.7 (↓ 0.4)	-	-	71.5 (↓ 0.5)	-	-	69.7 (↓ 0.4)	-	-	69.7 (↓ 0.6)	-	-
	Text2Image	70.9 (↓ 0.2)	-	-	71.4 (↓ 0.6)	-	-	69.8 (↓ 0.3)	-	-	69.8 (↓ 0.5)	-	-
	FireDM (Ours)	71.2 (↑ 0.1)	71.8 (↑ 0.1)	72.2 (↑ 0.3)	72.3 (↑ 0.3)	72.6 (↑ 0.2)	74.5 (↑ 0.2)	70.3 (↑ 0.2)	71.3 (↑ 0.4)	71.3 (↑ 0.1)	70.7 (↑ 0.4)	71.0 (↑ 0.4)	71.9 (↑ 0.5)
	Baseline-F	71.1	71.7	71.9	72.0	72.4	74.3	70.1	70.9	71.2	70.3	70.6	71.4
Vehicle	DiffuMask	70.6 (↓ 0.5)	-	-	71.6 (↓ 0.6)	-	-	69.5 (↓ 0.5)	-	-	70.0 (↓ 0.5)	-	-
	DatasetDM	70.8 (↓ 0.3)	-	-	71.9 (↓ 0.3)	-	-	69.8 (↓ 0.2)	-	-	70.1 (↓ 0.4)	-	-
	Text2Image	70.7 (↓ 0.4)	-	-	71.5 (↓ 0.7)	-	-	69.7 (↓ 0.3)	-	-	69.9 (↓ 0.6)	-	-
	FireDM (Ours)	71.0 (↓ 0.1)	71.7 (↑ 0.3)	72.2 (↓ 0.1)	72.3 (↑ 0.1)	72.6 (↑ 0.1)	73.0 (↓ 0.1)	70.1 (↑ 0.1)	70.5 (↑ 0.1)	70.8 (↑ 0.1)	70.8 (↑ 0.3)	71.0 (↓ 0.2)	71.3 (↑ 0.2)
	Baseline-V	71.1	71.4	72.3	72.2	72.5	73.1	70.0	70.4	70.7	70.5	71.2	71.1
Ship	DiffuMask	68.5 (↓ 0.9)	-	-	70.6 (↓ 0.6)	-	-	68.1 (↓ 0.6)	-	-	67.5 (↓ 0.6)	-	-
	DatasetDM	68.8 (↓ 0.6)	-	-	70.8 (↓ 0.4)	-	-	68.3 (↓ 0.4)	-	-	67.6 (↓ 0.5)	-	-
	Text2Image	68.7 (↓ 0.7)	-	-	70.8 (↓ 0.4)	-	-	68.4 (↓ 0.3)	-	-	67.4 (↓ 0.7)	-	-
	FireDM (Ours)	69.8 (↑ 0.4)	70.4 (↑ 0.2)	71.3 (↑ 0.2)	71.4 (↑ 0.2)	72.0 (↓ 0.1)	72.1 (↓ 0.1)	68.9 (↑ 0.2)	69.5 (↓ 0.1)	70.0 (↓ 0.1)	70.4 (↑ 0.3)	69.3 (↓ 0.2)	70.3 (↓ 0.2)
	Baseline-S	69.4	70.2	71.1	71.2	72.1	72.2	68.7	69.6	70.1	68.1	69.5	70.5
Industrial	DiffuMask	69.5 (↓ 0.6)	-	-	70.8 (↓ 0.7)	-	-	68.2 (↓ 0.7)	-	-	67.8 (↓ 0.6)	-	-
	DatasetDM	69.7 (↓ 0.4)	-	-	70.9 (↓ 0.6)	-	-	68.3 (↓ 0.6)	-	-	68.0 (↓ 0.4)	-	-
	Text2Image	69.6 (↓ 0.5)	-	-	70.7 (↓ 0.8)	-	-	68.3 (↓ 0.6)	-	-	67.9 (↓ 0.5)	-	-
	FireDM (Ours)	70.0 (↓ 0.1)	70.8 (↑ 0.2)	71.8 (↑ 0.2)	71.7 (↑ 0.2)	72.3 (↑ 0.2)	72.6 (↑ 0.4)	68.8 (↓ 0.1)	70.4 (↑ 0.1)	70.8 (↑ 0.1)	68.6 (↑ 0.2)	69.7 (↑ 0.4)	70.9 (↓ 0.2)
	Baseline-I	70.1	70.6	71.6	71.5	72.1	72.2	68.9	70.3	70.7	68.4	69.3	71.1
Mine	DiffuMask	69.4 (↓ 0.6)	-	-	69.9 (↓ 0.8)	-	-	65.5 (↓ 0.7)	-	-	67.7 (↓ 0.6)	-	-
	DatasetDM	69.6 (↓ 0.4)	-	-	69.8 (↓ 0.9)	-	-	65.6 (↑ 0.6)	-	-	67.5 (↓ 0.8)	-	-
	Text2Image	69.7 (↓ 0.3)	-	-	69.9 (↓ 0.8)	-	-	65.4 (↓ 0.8)	-	-	67.6 (↓ 0.7)	-	-
	FireDM (Ours)	70.1 (↑ 0.1)	70.6 (↑ 0.2)	71.7 (↑ 0.1)	71.5 (↓ 0.2)	72.1 (↑ 0.1)	72.5 (↑ 0.2)	66.5 (↑ 0.3)	67.8 (↓ 0.1)	70.5 (↓ 0.1)	68.5 (↑ 0.2)	69.5 (↓ 0.1)	70.8 (↑ 0.2)
	Baseline-M	70.0	70.4	71.6	71.7	72.0	72.3	66.2	67.9	69.6	68.3	69.6	70.6

Table 13
Comparison of image quality between fire datasets generated by three data augmentation algorithms and those generated by FireDM.

Dataset	Image Size	SSIM (↑)	PSNR (↑)	FID (↓)	LPIPS (↓)	PieAPP (↓)
DiffusionMask	512x512	0.62	24.2	5.1	0.90	0.58
DatasetDM	512x512	0.65	24.7	4.8	0.86	0.54
Text2Image	512x512	0.64	24.4	4.9	0.88	0.56
FireDM-512	512x512	0.71	25.4	4.5	0.68	0.42
FireDM-768	768x768	0.73	25.6	4.3	0.64	0.40
FireDM-1024	1024x1024	0.76	25.8	4.0	0.56	0.34

image quality, meaning less noise and richer details, can be beneficial for the training of segmentation algorithms to a certain extent. Secondly, the decoders for perceiving masks in DatasetDM and DiffuMask use Mask2Former, a generic segmentation algorithm that tends to perform poorly in the challenging domain of flames. In contrast, our FireDM's perception decoder, Fire-Decoder, has been finely tuned for fire-related tasks (as detailed in Section 3.3.2), making it more efficient and precise. Meanwhile, the Text2Image method generates diverse foreground object images through a text-to-image diffusion model and combines them with segmentation algorithms to extract foreground masks, which are then merged with background images. This approach may disrupt the natural correlation between the target object and its environment, limiting the overall realism of the images.

Upon further investigation into the use of three datasets with different sizes for the same segmentation algorithm, as shown in Table. 14, we noted some intriguing phenomena. Taking DeepLabV3+ as an example, when the size increased from 512 to 1024, the segmentation accuracy exhibited a trend of initially rising and then declining. This phenomenon can be attributed to the input size set for DeepLabV3+ at 512×512. As the resolution of the dataset used to train DeepLabV3+ gradually increased from 512, two impacts emerged: firstly, the accumulation of local information in each image progressively facilitated the training process. However, on the other hand, since the dataset size exceeded 512, cropping occurred during training, potentially compromising the overall information of the images. Fortunately, during the initial phase of the dataset's image size increment, the extent of cropping was not significant, resulting in only a slight enhancement in overall training accuracy when considering the combined effects of these two factors.

On the contrary, the situation varies for the Mask2Former algorithm. With an input size set at 1024×1024, the gradual increase in dataset resolution from 512×512 to 1024×1024 enhances the accumulation of relevant information, leading to a gradual improvement in the algorithm's segmentation accuracy. The final experimental results indeed support this observation. In summary, the above conclusions further underscore the significance of FireDM in generating datasets

Table 14
Multiple fire segmentation datasets of various scales generated by diverse data augmentation algorithms in the evaluation results table of multiple segmentation algorithms. The column Input Size refers to the optimal image size of the segmentation algorithms as input to the training set, a property determined by the structure of their respective segmentation algorithms. The "baseline" row in the table represents a dataset composed of real images, with an equal number of images as the other control groups.

Segmentation algorithm	Input size	Data augmentation algorithm	IoU (%)			Pixel Accuracy (%)			F1-Score (%)			Average Precision (%)		
			512×512	768×768	1024×1024	512×512	768×768	1024×1024	512×512	768×768	1024×1024	512×512	768×768	1024×1024
DeepLab V3 ([83])	512×512	DiffuMask	68.67 (↓ 3.68)	-	-	69.93 (↓ 2.69)	-	-	67.91 (↓ 2.96)	-	-	69.01 (↓ 2.77)	-	-
		DatasetDM	70.22 (↓ 2.13)	-	-	71.29 (↓ 1.33)	-	-	69.34 (↓ 1.53)	-	-	70.13 (↓ 1.65)	-	-
		Text2Image	69.89 (↓ 2.46)	-	-	70.27 (↓ 2.35)	-	-	68.22 (↓ 2.65)	-	-	69.20 (↓ 2.58)	-	-
		FireDM(**Ours**)	73.21 (↑ 0.86)	73.43 (↑ 0.96)	73.25 (↑ 0.61)	74.58 (↑ 1.96)	75.24 (↑ 2.13)	74.88 (↑ 1.67)	72.32 (↑ 1.45)	72.67 (↑ 1.49)	72.55 (↑ 1.20)	72.59 (↑ 0.81)	72.66 (↑ 0.71)	73.33 (↑ 1.32)
		Baseline	72.35	72.47	72.64	72.62	73.11	73.21	70.87	71.18	71.35	71.78	71.95	72.01
DeepLab V3+ ([84])	512×512	DiffuMask	72.67 (↓ 2.54)	-	-	72.99 (↓ 2.46)	-	-	71.30 (↓ 2.63)	-	-	72.21 (↓ 2.71)	-	-
		DatasetDM	73.15 (↓ 2.06)	-	-	74.54 (↓ 0.91)	-	-	72.45 (↓ 1.48)	-	-	73.03 (↓ 1.89)	-	-
		Text2Image	72.95 (↓ 2.26)	-	-	73.32 (↓ 2.13)	-	-	71.34 (↓ 2.59)	-	-	72.45 (↓ 2.47)	-	-
		FireDM(**Ours**)	76.11 (↑ 0.90)	76.89 (↑ 1.56)	76.46 (↑ 1.15)	77.97 (↑ 2.52)	78.41 (↑ 2.41)	77.98 (↑ 1.49)	75.63 (↑ 1.70)	75.46 (↑ 1.03)	75.31 (↑ 0.55)	75.99 (↑ 1.07)	76.19 (↑ 1.22)	76.13 (↑ 1.10)
		Baseline	75.21	75.33	75.31	75.45	76.00	76.49	73.93	74.43	74.76	74.92	74.97	75.03
BiseNet ([85])	512×512	DiffuMask	72.17 (↓ 2.84)	-	-	73.22 (↓ 1.98)	-	-	71.34 (↓ 2.22)	-	-	72.46 (↓ 2.21)	-	-
		DatasetDM	72.67 (↓ 2.34)	-	-	74.23 (↓ 0.97)	-	-	72.10 (↓ 1.46)	-	-	72.72 (↓ 1.95)	-	-
		Text2Image	72.23 (↓ 2.78)	-	-	73.11 (↓ 2.09)	-	-	71.12 (↓ 2.44)	-	-	72.10 (↓ 2.57)	-	-
		FireDM(**Ours**)	76.00 (↑ 0.99)	76.57 (↑ 1.43)	76.23 (↑ 1.11)	77.65 (↑ 2.45)	78.21 (↑ 2.43)	77.78 (↑ 1.54)	75.37 (↑ 1.81)	75.34 (↑ 1.11)	75.10 (↑ 0.54)	75.56 (↑ 0.89)	75.76 (↑ 0.98)	75.72 (↑ 0.96)
		Baseline	75.01	75.14	75.12	75.20	75.78	76.24	73.56	74.23	74.56	74.67	74.78	74.76
BiseNetv2 ([86])	640×640	DiffuMask	72.77 (↓ 3.45)	-	-	74.31 (↓ 1.70)	-	-	72.45 (↓ 2.42)	-	-	72.99 (↓ 2.88)	-	-
		DatasetDM	73.81 (↓ 2.41)	-	-	75.01 (↓ 1.00)	-	-	73.22 (↓ 1.65)	-	-	73.89 (↓ 1.98)	-	-
		Text2Image	73.44 (↓ 2.78)	-	-	74.37 (↓ 1.64)	-	-	72.25 (↓ 2.62)	-	-	73.24 (↓ 2.63)	-	-
		FireDM(**Ours**)	77.13 (↑ 0.91)	77.78 (↑ 1.44)	77.45 (↑ 1.15)	78.54 (↑ 2.53)	79.05 (↑ 2.16)	78.89 (↑ 2.03)	76.54 (↑ 1.67)	76.75 (↑ 0.65)	76.67 (↑ 0.96)	76.67 (↑ 0.80)	77.89 (↑ 1.11)	77.71 (↑ 1.12)
		Baseline	76.22	76.34	76.30	76.01	76.89	76.86	74.87	76.10	75.71	75.87	76.78	76.59
Segmenter ([87])	640×640	DiffuMask	76.31 (↓ 0.85)	-	-	76.29 (↓ 0.94)	-	-	74.37 (↓ 1.49)	-	-	75.31 (↓ 1.52)	-	-
		DatasetDM	75.33 (↓ 1.83)	-	-	76.88 (↓ 0.35)	-	-	74.88 (↓ 0.98)	-	-	75.37 (↓ 1.46)	-	-
		Text2Image	75.11 (↓ 2.05)	-	-	75.68 (↓ 1.55)	-	-	73.78 (↓ 2.08)	-	-	74.87 (↓ 1.96)	-	-
		FireDM(**Ours**)	78.46 (↑ 1.30)	78.98 (↑ 1.77)	78.76 (↑ 1.56)	79.91 (↑ 2.68)	80.21 (↑ 2.19)	79.88 (↑ 2.10)	77.78 (↑ 1.92)	77.78 (↑ 1.16)	77.12 (↑ 0.53)	77.76 (↑ 0.93)	78.04 (↑ 0.93)	78.01 (↑ 1.02)
		Baseline	77.16	77.21	77.20	77.23	78.02	77.78	75.86	76.62	76.59	76.83	77.11	76.99
Segformer ([88])	640×640	DiffuMask	77.13 (↓ 1.32)	-	-	77.67 (↓ 0.98)	-	-	76.00 (↓ 0.86)	-	-	77.01 (↓ 1.20)	-	-
		DatasetDM	76.25 (↓ 2.20)	-	-	77.64 (↓ 1.01)	-	-	75.56 (↓ 1.30)	-	-	76.14 (↓ 2.07)	-	-
		Text2Image	76.13 (↓ 2.32)	-	-	76.41 (↓ 2.24)	-	-	74.17 (↓ 2.69)	-	-	75.22 (↓ 2.99)	-	-
		FireDM(**Ours**)	79.27 (↑ 0.82)	79.97 (↑ 1.37)	79.63 (↑ 1.19)	80.95 (↑ 2.30)	81.36 (↑ 2.15)	81.12 (↑ 2.02)	78.89 (↑ 2.03)	78.67 (↑ 1.00)	78.53 (↑ 0.19)	79.02 (↑ 0.81)	80.02 (↑ 1.55)	79.34 (↑ 0.99)
		Baseline	78.45	78.60	78.44	78.65	79.21	79.10	76.86	77.67	78.34	78.21	78.47	78.35
MaskFormer ([89])	1024×1024	DiffuMask	77.67 (↓ 1.89)	-	-	77.99 (↓ 1.79)	-	-	76.45 (↓ 1.58)	-	-	77.21 (↓ 2.24)	-	-
		DatasetDM	77.03 (↓ 2.53)	-	-	78.42 (↓ 1.36)	-	-	76.98 (↓ 1.05)	-	-	77.45 (↓ 2.00)	-	-
		Text2Image	76.73 (↓ 2.83)	-	-	78.12 (↓ 1.66)	-	-	75.87 (↓ 2.16)	-	-	76.10 (↓ 3.35)	-	-
		FireDM(**Ours**)	80.55 (↑ 0.99)	81.56 (↑ 1.84)	82.33 (↑ 2.06)	81.47 (↑ 1.69)	82.65 (↑ 2.55)	83.20 (↑ 2.53)	79.91 (↑ 1.88)	80.25 (↑ 1.31)	81.26 (↑ 1.53)	80.11 (↑ 0.66)	81.23 (↑ 1.51)	82.02 (↑ 2.09)
		Baseline	79.56	79.72	80.27	79.78	80.10	80.67	78.03	78.94	79.73	79.45	79.72	79.93
Mask2Former ([35])	1024×1024	DiffuMask	77.89 (↓ 2.32)	-	-	79.23 (↓ 1.22)	-	-	77.66 (↓ 1.46)	-	-	77.91 (↓ 2.65)	-	-
		DatasetDM	78.25 (↓ 1.96)	-	-	79.37 (↓ 1.08)	-	-	78.02 (↓ 1.10)	-	-	78.67 (↓ 1.89)	-	-
		Text2Image	77.91 (↓ 2.30)	-	-	79.11 (↓ 1.34)	-	-	77.21 (↓ 1.91)	-	-	77.56 (↓ 3.00)	-	-
		FireDM(**Ours**)	81.78 (↑ 1.57)	82.88 (↑ 1.56)	83.78 (↑ 1.44)	82.73 (↑ 2.28)	83.87 (↑ 2.53)	84.57 (↑ 2.12)	81.22 (↑ 2.10)	81.34 (↑ 1.22)	82.54 (↑ 1.65)	81.34 (↑ 0.78)	82.43 (↑ 1.64)	83.37 (↑ 2.16)
		Baseline	80.21	81.32	82.34	80.45	81.34	82.45	79.12	80.12	80.89	80.56	80.79	81.21

with varying image sizes. This capability allows the selection of datasets tailored to the training requirements of different segmentation algorithms. It is crucial to note that datasets generated by FireDM are indeed compatible with various segmentation algorithms.

We can see similar experimental results as shown in Fig. 7. This figure shows the validation set loss curves of the four latest segmentation algorithms - Segmenter, SegFormer, MaskFormer and Mask2Former - during training. First, we focus on two algorithms, Segmenter and SegFormer. Since the input size (input size) of these two algorithms is 640, we expect that the optimal loss value should be close to an image of 640×640 size. By looking at parts a and b of the figure, we find that the experimental results are indeed consistent with this prediction. Next, we analyse two algorithms, SegFormer and Mask2Former. Since the input size of these two algorithms is 1024, we predict that the convergence of the loss curve will gradually slow down as the size of the training image increases. At the same time, the best training results should occur on the dataset whose size is closest to 1024. Observing parts c and d of the figure, we can see that the experimental results match our prediction. The experimental phenomena described further validate that different segmentation algorithms each have their optimal dataset sizes for training. This highlights the importance of FireDM's capability to generate multi-scale datasets. In contrast, the comparative data augmentation algorithms—DatasetDM, DiffuMask, and Text2Image—are limited by the constraints of the diffusion model (SD 1.4) they employ, resulting in their ability to only generate datasets with an image resolution of 512×512. This limitation means their generated datasets cannot perfectly adapt to the requirements of various segmentation algorithms, significantly restricting the application prospects of these three methods.

5.5. Practical Application Testing of FireDM Generated Datasets

In this section, we aim to highlight the practical application potential of FireDM in generating datasets by enumerating some of the actual difficulties encountered in fire detection. For example, fire scenes often involve cluttered obstacles obstructing flames, thick smoke and fog, and so on. Therefore, the main focus of this section is to explore whether Mask2Former can accurately test for fires in these three scenarios after being trained using the datasets provided in Table. 2. To address this question, the primary task is to acquire datasets of fires in these three interference scenarios. To accomplish this, we selected relevant scene images from the internet, with 500 images for each scenario, and manually annotated segmentation masks for each of them.

The final experimental results are presented in Table. C17, and Table. C18 in Appendix C. The two datasets generated by FireDM in these three tables have comprehensively outperformed the public datasets. This indicates that the potential application of the datasets generated by FireDM in fire detection is very significant.

6. Conclusions

This study introduces FireDM, an innovative fire segmentation dataset generation method based on advanced diffusion models. FireDM uniquely combines pre-trained

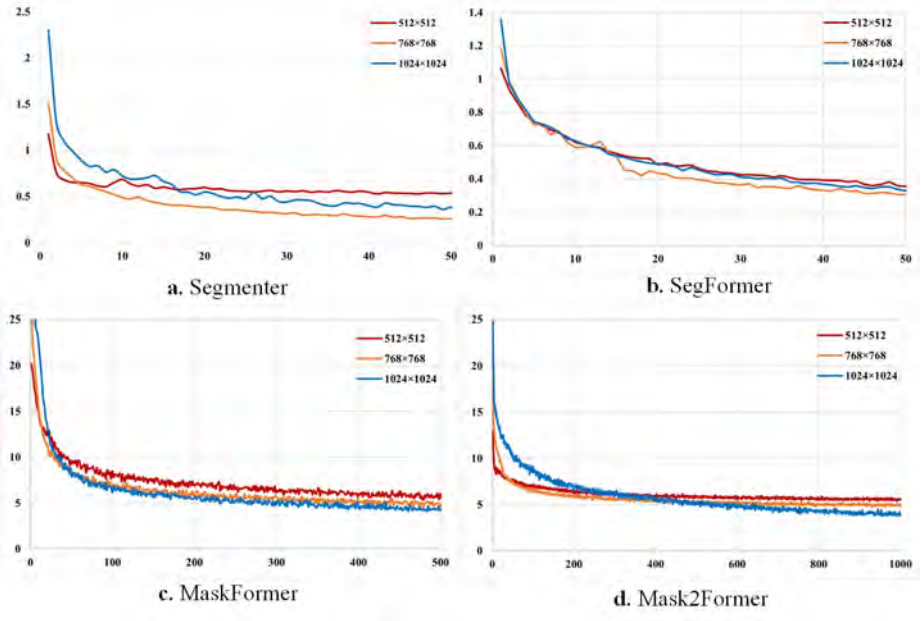

Figure 7: Loss curve graphs of four segmentation algorithms trained on datasets with three different image sizes. The horizontal coordinates in the four subplots of this figure are epochs and the vertical coordinates are the loss values.

diffusion models (SDXL 1.0 and SD 2.1) with a specialized fire domain text generator (ChatGPT4-Fire), facilitating the creation of high-quality, diverse fire scene images and corresponding segmentation masks with minimal human annotation. A key innovation in FireDM is the Fire-Decoder module, adeptly decoding accurate segmentation masks from cross-attention and multi-scale feature maps during the diffusion process. As the pioneering method dedicated to fire scene segmentation dataset generation, FireDM demonstrates several advantages. It not only produces highly realistic and detailed fire images and segmentation masks but also leverages the latent encoding and visual alignment principles of diffusion models, requiring minimal annotated data for training the decoder.

Furthermore, the integration of large-scale language models enhances the detail and diversity of flame image generation, bolstering the generalization ability of the produced images. FireDM's flexibility in adapting to the input size requirements of various segmentation algorithms, and its ability to dynamically select dataset sizes makes it a robust solution. The method's capacity to generate a comprehensive fire segmentation dataset, encompassing different sizes and common fire scenarios, positions FireDM as the largest and highest-quality publicly available fire segmentation dataset globally. Finally, our evaluation through training and testing with mainstream segmentation algorithms, such as DeepLabV3, DeepLabv3+, and Mask2Former, on the FireDM-generated dataset indicates that the dataset closely approximates real fire scenes and offers versatile scale distributions adaptable to a wide range of segmentation algorithms, thus significantly enhancing baseline quality. FireDM sets a new benchmark for fire scene image generation and segmentation, and it will contribute significantly to future smart firefighting operations.

CRediT Authorship Contribution Statement

Hongtao Zheng: Conceptualization, Methodology, Software, Validation, Formal analysis, Resources Investigation, Data Curation, Writing - Original Draft, Visualization. **Meng Wang:** Conceptualization, Methodology, Validation, Investigation, Data Curation, Writing – review & editing. **Zilong Wang:** Validation, Investigation, Data Curation, Writing – review & editing. **Xinyan Huang:** Validation, Resources, Writing – review & editing, Supervision, Project administration, Funding acquisition.

Declaration of Competing Interest

The authors have no relevant financial or non-financial interests to disclose. The authors declare no conflicts of interest related to the content of this article.

Data Availability

The data and code for this paper can be obtained from https://github.com/ZhengHongtao2001/FireDM. The remaining data will be provided upon request.

Acknowledgements

HZ thanks the 2023 National College Student Innovation Training Program (202313021010) and Zhejiang Provincial Key Research and Development Project (2019C01150).

XH thanks the RGC Theme-based Research Scheme (T22-505/19-N). We would like to express our sincere thanks to Wenhao Deng from the University of Nottingham for their correction of grammatical errors in the article and their valuable suggestions at the early stage of the paper submission.

Appendix A. Details of FireDM

See Table. A15, Table. A16 and Fig. A8.

Appendix B. Visualization of the Dataset Generated by FireDM

See Fig. B9, Fig. B10, Fig. B11 and Fig. B12.

Appendix C. Explore the Application Potential of FireDM-Generated Datasets

See Table. C17 and Table. C18.

References

[1] Alexandro B Leverkus, Pablo García Murillo, Vicente Jurado Doña, and Juli G Pausas. Wildfires: Opportunity for restoration? *Science*, 363(6423):134–135, 2019.

[2] Franz Schug, Avi Bar-Massada, Amanda R Carlson, Heather Cox, Todd J Hawbaker, David Helmers, Patrick Hostert, Dominik Kaim, Neda K Kasraee, Sebastián Martinuzzi, et al. The global wildland–urban interface. *Nature*, 621(7977):94–99, 2023.

[3] Dougal Drysdale. *An introduction to fire dynamics*. John wiley & sons, 2011.

[4] Aatif Ali Khan, Mustesin Ali Khan, Kamtak Leung, Xinyan Huang, Mingchun Luo, and Asif Usmani. A review of critical fire event library for buildings and safety framework for smart firefighting. *International Journal of Disaster Risk Reduction*, page 103412, 2022.

[5] Haukur Ingason, Ying Zhen Li, and Anders Lönnermark. *Tunnel fire dynamics*. Springer, 2014.

[6] Marty Ahrens and Ben Evarts. Fire loss in the united states during 2020, 2021.

[7] A Enis Cetin, Bart Merci, Osman Günay, Behçet Ugur Töreyin, and Steven Verstockt. *Methods and techniques for fire detection: signal, image and video processing perspectives*. Academic Press, 2016.

[8] Xiqiang Wu, Younggi Park, Ao Li, Xinyan Huang, Fu Xiao, and Asif Usmani. Smart detection of fire source in tunnel based on the numerical database and artificial intelligence. *Fire Technology*, 57:657–682, 2021.

[9] Zilong Wang, Tianhang Zhang, Xiqiang Wu, and Xinyan Huang. Predicting transient building fire based on external smoke images and deep learning. *Journal of Building Engineering*, 47:103823, 2022.

[10] Yizhou Li, Zilong Wang, and Xinyan Huang. An exploration of equivalent scenarios for building facade fire standard tests. *Journal of Building Engineering*, 52:104399, 2022.

[11] Tianhang Zhang, Zilong Wang, Ho Yin Wong, Wai Cheong Tam, Xinyan Huang, and Fu Xiao. Real-time forecast of compartment fire and flashover based on deep learning. *Fire Safety Journal*, 130:103579, 2022.

[12] Yifei Ding, Yuxin Zhang, and Xinyan Huang. Intelligent emergency digital twin system for monitoring building fire evacuation. *Journal of Building Engineering*, 77:107416, 2023.

[13] Hongtao Zheng, Sounkalo Dembele, Yongxin Wu, Yan Liu, Hongli Chen, and Qiujie Zhang. A lightweight algorithm capable of accurately identifying forest fires from uav remote sensing imagery. *Frontiers in Forests and Global Change*, 6:1134942, 2023.

[14] Hongtao Zheng, Junchen Duan, Yu Dong, and Yan Liu. Real-time fire detection algorithms running on small embedded devices based on mobilenetv3 and yolov4. *Fire Ecology*, 19(1):31, 2023.

[15] Hongtao Zheng, Gaoyang Wang, Duo Xiao, Hong Liu, and Xiaoyin Hu. Fta-detr: An efficient and precise fire detection framework based on an end-to-end architecture applicable to embedded platforms. *Expert Systems with Applications*, page 123394, 2024.

[16] Zilong Wang, Tianhang Zhang, and Xinyan Huang. Explainable deep learning for image-driven fire calorimetry. *Applied Intelligence*, 213(12), 2023.

[17] Carmina Pérez-Guerrero, Jorge Francisco Ciprián-Sánchez, Adriana Palacios, Gilberto Ochoa-Ruiz, Miguel Gonzalez-Mendoza, Vahid Foroughi, Elsa Pastor, and Gerardo Rodriguez-Hernandez. Computer vision-based characterization of large-scale jet flames using a synthetic infrared image generation approach. *Engineering Applications of Artificial Intelligence*, 127:107275, 2024.

[18] Tianhang Zhang, Zilong Wang, Yanfu Zeng, Xiqiang Wu, Xinyan Huang, and Fu Xiao. Building artificial-intelligence digital fire (aid-fire) system: A real-scale demonstration. *Journal of Building Engineering*, 62:105363, 2022.

[19] Yu-Cheng Zhou, Zhen-Zhong Hu, Ke-Xiao Yan, and Jia-Rui Lin. Deep learning-based instance segmentation for indoor fire load recognition. *IEEE Access*, 9:148771–148782, 2021.

[20] Negar Elhami-Khorasani, Juan Gustavo Salado Castillo, Esther Saula, Timothy Josephs, Gauhar Nurlybekova, and Thomas Gernay. Application of a digitized fuel load surveying methodology to office buildings. *Fire technology*, 57:101–122, 2021.

[21] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[22] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014.

[23] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4401–4410, 2019.

[24] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8110–8119, 2020.

[25] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *International Conference on Machine Learning*, pages 8821–8831. PMLR, 2021.

[26] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 1(2):3, 2022.

[27] 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*, pages 10684–10695, 2022.

[28] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. Sdxl: Improving latent diffusion models for high-resolution image synthesis. *arXiv preprint arXiv:2307.01952*, 2023.

[29] Yuxuan Zhang, Huan Ling, Jun Gao, Kangxue Yin, Jean-Francois Lafleche, Adela Barriuso, Antonio Torralba, and Sanja Fidler. Datasetgan: Efficient labeled data factory with minimal human effort. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10145–10155, 2021.

[30] Daiqing Li, Huan Ling, Seung Wook Kim, Karsten Kreis, Sanja Fidler, and Antonio Torralba. Bigdatasetgan: Synthesizing imagenet with pixel-wise annotations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21330–21340, 2022.

[31] Yunhao Ge, Jiashu Xu, Brian Nlong Zhao, Neel Joshi, Laurent Itti, and Vibhav Vineet. Beyond generation: Harnessing text to image models for object detection and segmentation. *arXiv preprint arXiv:2309.05956*, 2023.

[32] Weijia Wu, Yuzhong Zhao, Hao Chen, Yuchao Gu, Rui Zhao, Yefei He, Hong Zhou, Mike Zheng Shou, and Chunhua Shen. Datasetdm: Synthesizing data with perception annotations using diffusion models. *arXiv preprint arXiv:2308.06160*, 2023.

[33] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021.

[34] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88:303–338, 2010.

[35] Bowen Cheng, Ishan Misra, Alexander G Schwing, Alexander Kirillov, and Rohit Girdhar. Masked-attention mask transformer for universal image segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1290–1299, 2022.

[36] Daniel YT Chino, Letricia PS Avalhais, Jose F Rodrigues, and Agma JM Traina. Bowfire: detection of fire in still images by integrating pixel color and texture analysis. In *2015 28th SIBGRAPI conference on graphics, patterns and images*, pages 95–102. IEEE, 2015.

[37] Tom Toulouse, Lucile Rossi, Antoine Campana, Turgay Celik, and Moulay A Akhloufi. Computer vision for wildfire research: An evolving image dataset for processing and analysis. *Fire Safety Journal*, 92:188–194, 2017.

[38] Alireza Shamsoshoara, Fatemeh Afghah, Abolfazl Razi, Liming Zheng, Peter Z Fulé, and Erik Blasch. Aerial imagery pile burn detection using deep learning: The flame dataset. *Computer Networks*, 193:108001, 2021.

[39] Elman Mansimov, Emilio Parisotto, Jimmy Lei Ba, and Ruslan Salakhutdinov. Generating images from captions with attention. *arXiv preprint arXiv:1511.02793*, 2015.

[40] Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee. Generative adversarial text to image synthesis. In *International conference on machine learning*, pages 1060–1069. PMLR, 2016.

[41] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris N Metaxas. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 5907–5915, 2017.

[42] Tao Xu, Pengchuan Zhang, Qiuyuan Huang, Han Zhang, Zhe Gan, Xiaolei Huang, and Xiaodong He. Attngan: Fine-grained text to image generation with attentional generative adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1316–1324, 2018.

[43] Bowen Li, Xiaojuan Qi, Thomas Lukasiewicz, and Philip Torr. Controllable text-to-image generation. *Advances in Neural Information Processing Systems*, 32, 2019.

[44] Jiahui Yu, Yuanzhong Xu, Jing Yu Koh, Thang Luong, Gunjan Baid, Wang, et al. Scaling autoregressive models for content-rich text-to-image generation. *arXiv preprint arXiv:2206.10789*, 2022.

[45] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. 2018.

[46] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[47] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[48] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*, 2021.

[49] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*, 2021.

[50] Xiao Liu, Yanan Zheng, Zhengxiao Du, Ming Ding, Yujie Qian, Zhilin Yang, and Jie Tang. Gpt understands, too. *AI Open*, 2023.

[51] Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Lam Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. P-tuning v2: Prompt tuning can be comparable to fine-tuning universally across scales and tasks. *arXiv preprint arXiv:2110.07602*, 2021.

[52] Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*, 2021.

[53] Simeng Sun, Yang Liu, Dan Iter, Chenguang Zhu, and Mohit Iyyer. How does in-context learning help prompt tuning? *arXiv preprint arXiv:2302.11521*, 2023.

[54] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

[55] Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. Towards a unified view of parameter-efficient transfer learning. *arXiv preprint arXiv:2110.04366*, 2021.

[56] Jules White, Quchen Fu, Sam Hays, Michael Sandborn, Carlos Olea, Henry Gilbert, Ashraf Elnashar, Jesse Spencer-Smith, and Douglas C Schmidt. A prompt pattern catalog to enhance prompt engineering with chatgpt. *arXiv preprint arXiv:2302.11382*, 2023.

[57] Jindong Gu, Zhen Han, Shuo Chen, Ahmad Beirami, Bailan He, Gengyuan Zhang, Ruotong Liao, Yao Qin, Volker Tresp, and Philip Torr. A systematic survey of prompt engineering on vision-language foundation models. *arXiv preprint arXiv:2307.12980*, 2023.

[58] Ruiyang Ren, Yuhao Wang, Yingqi Qu, Wayne Xin Zhao, Jing Liu, Hao Tian, Hua Wu, Ji-Rong Wen, and Haifeng Wang. Investigating the factual knowledge boundary of large language models with retrieval augmentation. *arXiv preprint arXiv:2307.11019*, 2023.

[59] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. PMLR, 2015.

[60] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[61] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

[62] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

[63] Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

[64] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. *Advances in neural information processing systems*, 28, 2015.

[65] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1125–1134, 2017.

[66] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2337–2346, 2019.

[67] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[68] Andrew Jaegle, Felix Gimeno, Andy Brock, Oriol Vinyals, Andrew Zisserman, and Joao Carreira. Perceiver: General perception with iterative attention. In *International conference on machine learning*, pages 4651–4664. PMLR, 2021.

[69] Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, et al. Least-to-most prompting enables complex reasoning in large language models. *arXiv preprint arXiv:2205.10625*, 2022.

[70] Mehdi Cherti, Romain Beaumont, Ross Wightman, Mitchell Wortsman, Gabriel Ilharco, Cade Gordon, Christoph Schuhmann, Ludwig Schmidt, and Jenia Jitsev. Reproducible scaling laws for contrastive language-image learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2818–2829, 2023.

[71] Jiarui Xu, Sifei Liu, Arash Vahdat, Wonmin Byeon, Xiaolong Wang, and Shalini De Mello. Open-vocabulary panoptic segmentation with text-to-image diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2955–2966, 2023.

[72] Li Yuan, Qibin Hou, Zihang Jiang, Jiashi Feng, and Shuicheng Yan. Volo: Vision outlooker for visual recognition. *IEEE transactions on pattern analysis and machine intelligence*, 45(5):6575–6586, 2022.

[73] Xingxing Xie, Chunbo Lang, Shicheng Miao, Gong Cheng, Ke Li, and Junwei Han. Mutual-assistance learning for object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023.

[74] Gong Cheng, Pujian Lai, Decheng Gao, and Junwei Han. Class attention network for image recognition. *Science China Information Sciences*, 66(3):132105, 2023.

[75] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer, 2020.

[76] Peng Gao, Minghang Zheng, Xiaogang Wang, Jifeng Dai, and Hongsheng Li. Fast convergence of detr with spatially modulated co-attention. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 3621–3630, 2021.

[77] Harold W Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955.

[78] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

[79] Ekta Prashnani, Hong Cai, Yasamin Mostofi, and Pradeep Sen. Pieapp: Perceptual image-error assessment through pairwise preference. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1808–1817, 2018.

[80] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

[81] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2017.

[82] Weijia Wu, Yuzhong Zhao, Mike Zheng Shou, Hong Zhou, and Chunhua Shen. Diffumask: Synthesizing images with pixel-level annotations for semantic segmentation using diffusion models. *Proc. Int. Conf. Computer Vision (ICCV 2023)*, 2023.

[83] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017.

[84] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 801–818, 2018.

[85] Changqian Yu, Jingbo Wang, Chao Peng, Changxin Gao, Gang Yu, and Nong Sang. Bisenet: Bilateral segmentation network for real-time semantic segmentation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 325–341, 2018.

[86] Changqian Yu, Changxin Gao, Jingbo Wang, Gang Yu, Chunhua Shen, and Nong Sang. Bisenet v2: Bilateral network with guided aggregation for real-time semantic segmentation. *International Journal of Computer Vision*, 129:3051–3068, 2021.

[87] Robin Strudel, Ricardo Garcia, Ivan Laptev, and Cordelia Schmid. Segmenter: Transformer for semantic segmentation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 7262–7272, 2021.

[88] Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M Alvarez, and Ping Luo. Segformer: Simple and efficient design for semantic segmentation with transformers. *Advances in Neural Information Processing Systems*, 34:12077–12090, 2021.

[89] Bowen Cheng, Alex Schwing, and Alexander Kirillov. Per-pixel classification is not all you need for semantic segmentation. *Advances in Neural Information Processing Systems*, 34:17864–17875, 2021.

[90] Xinyan Huang and Wai Cheong Tam. *Intelligent Building Fire Safety and Smart Firefighting*. Springer Nature, 2024.

[91] Matthew W Jones, John T Abatzoglou, Sander Veraverbeke, Niels Andela, Gitta Lasslop, Matthias Forkel, Adam JP Smith, Chantelle Burton, Richard A Betts, Guido R van der Werf, et al. Global and regional trends and drivers of fire under climate change. *Reviews of Geophysics*, 60(3):e2020RG000726, 2022.

[92] Bart Merci and Tarek Beji. *Fluid mechanics aspects of fire and smoke dynamics in enclosures*. CRC press, 2022.

[93] Ben Evarts and Gary P Stein. *US fire department profile 2018*. National Fire Protection Association Quincy, MA, USA, 2020.

[94] Chile Associated Press in Viña del Mar. Forest fires in chile cause multiple deaths and widespread destruction. *The Guardian*, 2024.

[95] Hayden Vernon. Major incident declared in liverpool after large fire in city centre. *The Guardian*, 2024.

[96] Agence France-Presse in Beijing. Twelve people arrested over beijing hospital fire that killed 29. *The Guardian*, 2024.

[97] Greta Thunberg et al. Our house is on fire. *The Guardian*, 20:79–80, 2019.

[98] Brian J Meacham and Margaret McNamee. *Handbook of Fire and the Environment: Impacts and Mitigation*. Springer Nature, 2022.

[99] MZ Naser and Glenn Corbett. *Handbook of Cognitive and Autonomous Systems for Fire Resilient Infrastructures*. Springer, 2022.

[100] Kevin LaMalva and Danny Hopkin. *International handbook of structural fire engineering*. Springer, 2021.

[101] Morgan J Hurley, Daniel T Gottuk, John R Hall Jr, Kazunori Harada, Erica D Kuligowski, Milosh Puchovsky, John M Watts Jr, CHRISTOPHER J WIECZOREK, et al. *SFPE handbook of fire protection engineering*. Springer, 2015.

[102] Jon E Keeley and Alexandra D Syphard. Large california wildfires: 2020 fires in historical context. *Fire Ecology*, 17(1):1–11, 2021.

[103] Peiyi Sun, Roeland Bisschop, Huichang Niu, and Xinyan Huang. A review of battery fires in electric vehicles. *Fire technology*, 56:1361–1410, 2020.

[104] Lin Zhou, Gui Fu, and Yujingyang Xue. Human and organizational factors in chinese hazardous chemical accidents: A case study of the '8.12'tianjin port fire and explosion using the hfacs-hc. *International journal of occupational safety and ergonomics*, 24(3):329–340, 2018.

[105] C Emmy Prema, SS Vinsley, and S Suresh. Efficient flame detection based on static and dynamic texture analysis in forest fire detection. *Fire technology*, 54:255–288, 2018.

Table A15
Example of a Knowledge Source Utilized for Knowledge Enhancement.

Number	Text prompt	Fire types
1a	Intelligent Building Fire Safety and Smart Firefighting [90]	
1b	Global and regional trends and drivers of fire under climate change [91]	
1c	Fluid mechanics aspects of fire and smoke dynamics in enclosures [92]	Book
1d	US fire department profile 2018 [93]	
	.....	
2a	Forest fires in Chile cause multiple deaths and widespread destruction [94]	
2b	Major incident declared in Liverpool after large fire in city centre [95]	
2c	Twelve people arrested over Beijing hospital fire that killed 29 [96]	News Articles and Reports
2d	Our house is on fire [97]	
	.....	
3a	Handbook of Fire and the Environment: Impacts and Mitigation [98]	
3b	Handbook of Cognitive and Autonomous Systems for Fire Resilient Infrastructures [99]	
3c	International handbook of structural fire engineering [100]	Official Technical Manuals
3d	SFPE handbook of fire protection engineering [101]	
	.....	
4a	Large California wildfires: 2020 fires in historical context [102]	
4b	A review of battery fires in electric vehicles [103]	
4c	Human and organizational factors in Chinese hazardous chemical accidents ... [104]	Academic Papers And Reports
4d	Efficient flame detection based on static and dynamic texture analysis in forest fire detection [105]	
	.....	

Table A16
The input examples of a text prompt.

Number	Text prompt	Fire types
1a	A high-rise building in the city center erupts in flames. 8k, building disaster, building photo.	
1b	A residential complex faces a fierce fire outbreak, firefighters battling the flames. 8k, urban disaster, firefighting operation photo.	Structural fire
1c	An industrial facility experiences a devastating fire, billowing thick smoke into the sky. 8k, building disaster, emergency response photo.	
	.....	
2a	The forest floor crackles and pops as flames engulf the underbrush. 8k, forest disaster, wildfire photo.	
2b	Tall trees become torches in the night, illuminating the forest with a fiery glow. 8k, forest disaster, forest fire photo.	Forest fire
2c	Smoke billows over the treetops, signaling a serious forest fire below. 8k, forest disaster, environmental concern photo.	
	.....	
3a	A parked car erupts in flames in a quiet suburban street. 8k, vehicle disaster, emergency photo.	
3b	A bus on fire causes panic among passengers on a busy city road. 8k, vehicle disaster, safety photo.	Vehicle fire
3c	A truck carrying flammable materials catches fire on the highway. 8k, vehicle disaster, transport photo.	
	.....	
4a	A fishing boat catches fire at sea, with smoke visible from the shore. 8k, ship disaster, distant view photo.	
4b	Firefighters aboard a fireboat battle flames engulfing a cargo ship. 8k, ship disaster, action photo.	Ship fire
4c	A cruise ship fire creates panic among passengers, with lifeboats deployed. 8k, ship disaster, rescue operation photo.	
	.....	
5a	Miners evacuate as smoke billows from a shaft fire. 8k, mine disaster, rescue operation photo.	
5b	A sudden fire in the depths of a coal mine traps miners underground. 8k, mine disaster, crisis photo.	Mine fire
5c	An underground blast leads to a fire in the mine, causing chaos. 8k, mine disaster, emergency situation photo.	
	.....	
6a	Explosions light up the night as a factory fire spreads rapidly. 8k, industrial disaster, dramatic photo.	
6b	A factory's storage tanks are ablaze, creating a dangerous situation. 8k, mine disaster, intense photo.	Industrial fire
6c	Thick black smoke from the factory fire can be seen for miles. 8k, mine disaster, environmental hazard photo.	
	.....	

Figure A8: ChatGPT4-Fire is utilized to guide the diffusion process with its prompts. With just a few simple inputs, ChatGPT4-Fire can generate a multitude of diverse and professionally detailed prompts. Figure A8.A demonstrates the use of text descriptions for three types of fire scenarios to guide the generation of corresponding scene datasets. Figure A8.B showcases our capability to produce custom text prompts for specific fire incident locations, such as the Forbidden City in Beijing. Figure A8.C illustrates the generation of textual descriptions for fire scenes occurring at different times of the day by ChatGPT4-Fire.

Figure B9: FireDM was trained using the BoWFire dataset to generate datasets of three image sizes. Each of these image size datasets contains the three most dominant fire occurrence scenarios, i.e., forest fires, structure fires, and vehicle fires.

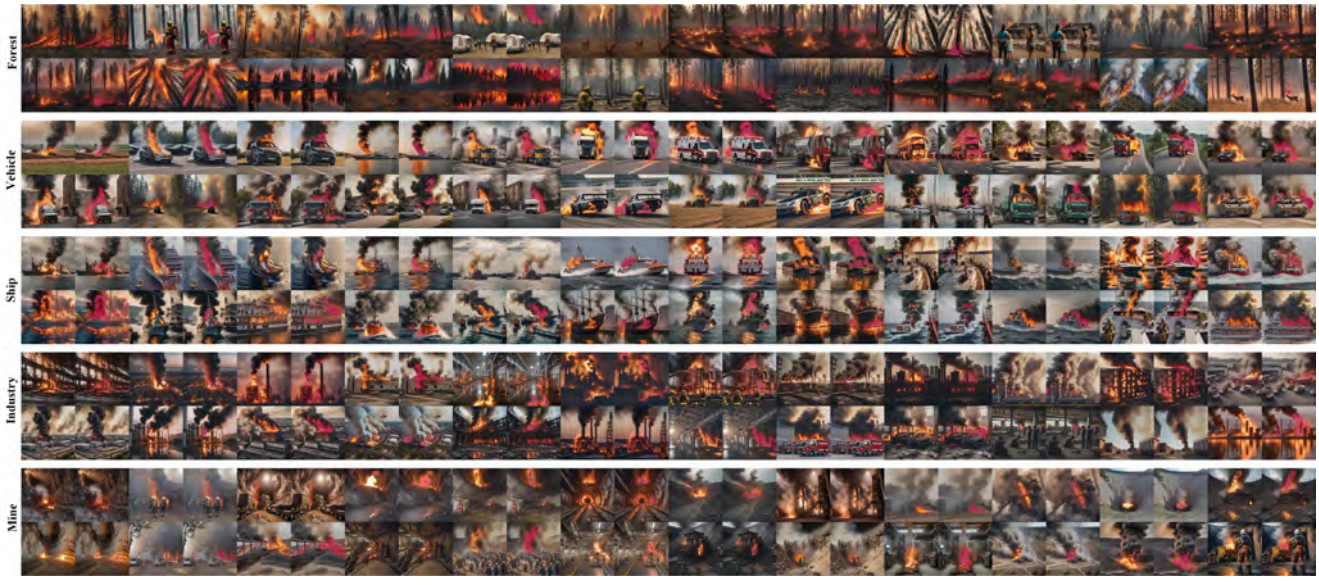

Figure B10: Visualisation results of the dataset generated by FireDM for five scenes with an image size of 512×512. The images are, from top to bottom: forest, vehicle, ship, industry, and mine.

Figure B11: Visualisation results of the dataset generated by FireDM for five scenes with an image size of 768×768. The images are, from top to bottom: forest, vehicle, ship, industry, and mine.

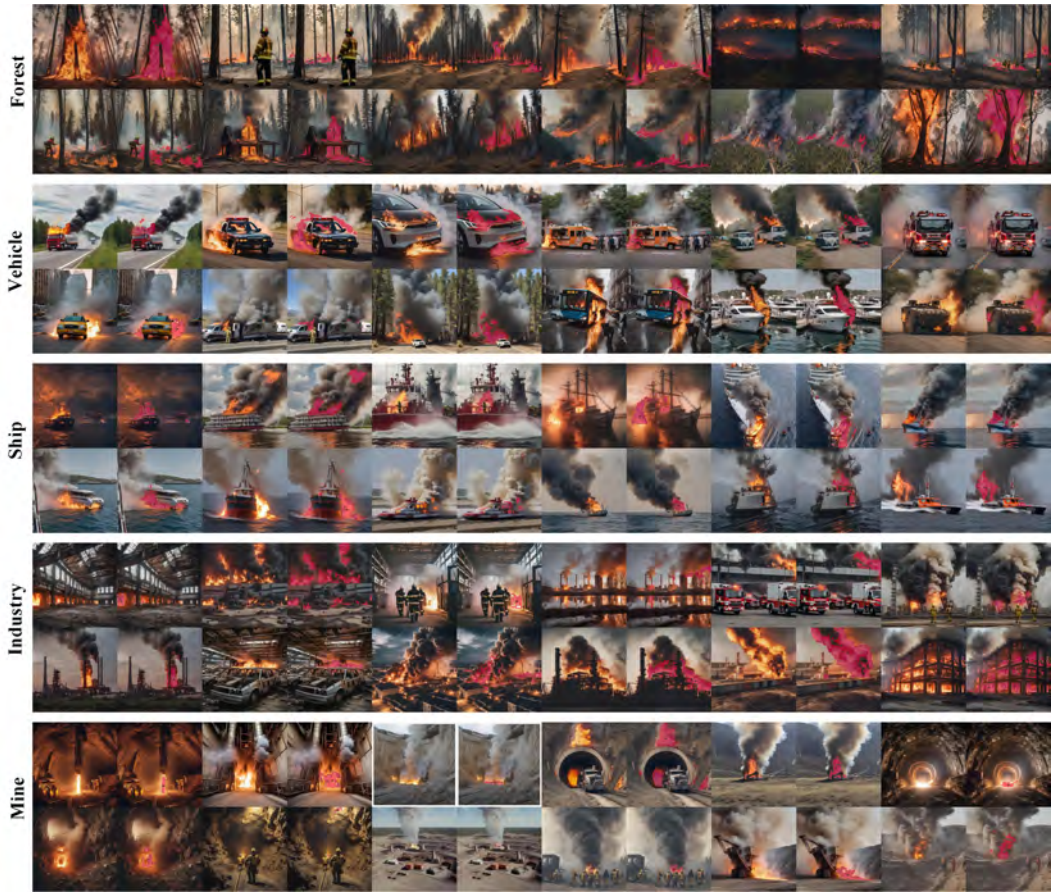

Figure B12: Visualisation results of the dataset generated by FireDM for five scenes with an image size of 1024×1024. The images are, from top to bottom: forest, vehicle, ship, industry, and mine.

Table C17
The segmentation results of the datasets generated by FireDM at three different image resolutions in response to severe obstruction in fire scenes.

Dataset	Real Image	Synthetic	Image Size	IoU (%)	Pixel Accuracy (%)	F1-Score (%)	Average Precision (%)
1. Baseline-BoWFire	119	0	-	43.24	45.21	40.52	39.47
2. FireDM-512-B	0	119	512×512	44.14 (↑ 0.90)	47.23 (↑ 2.02)	42.25 (↑ 1.73)	40.34 (↑ 0.87)
3. FireDM-768-B	0	119	768×768	44.37 (↑ 1.13)	47.67 (↑ 2.46)	42.58 (↑ 2.06)	40.68 (↑ 1.21)
4. FireDM-1024-B	0	119	1024×1024	44.68 (↑ 1.44)	47.89 (↑ 2.68)	42.76 (↑ 2.24)	40.78 (↑ 1.31)
5. Baseline-Corsican	1135	0	-	53.28	56.51	51.04	55.23
6. FireDM-512-C	0	1135	512×512	54.17 (↑ 0.89)	57.17 (↑ 0.66)	52.34 (↑ 1.30)	56.46 (↑ 1.23)
7. FireDM-768-C	0	1135	768×768	54.34 (↑ 1.06)	57.43 (↑ 0.92)	52.46 (↑ 1.42)	56.59 (↑ 1.36)
8. FireDM-1024-C	0	1135	1024×1024	55.46 (↑ 2.18)	57.53 (↑ 1.02)	52.59 (↑ 1.55)	56.73 (↑ 1.50)
9. Baseline-FLAME	2003	0	3480×2160	62.74	60.34	64.24	61.34
10. FireDM-512-F	0	2003	512×512	63.93 (↑ 1.19)	61.49 (↑ 1.15)	65.27 (↑ 1.03)	62.56 (↑ 1.22)
11. FireDM-768-F	0	2003	768×768	64.17 (↑ 1.43)	61.59 (↑ 1.25)	65.31 (↑ 1.07)	62.76 (↑ 1.42)
12. FireDM-1024-F	0	2003	1024×1024	64.32 (↑ 1.58)	61.65 (↑ 1.31)	65.56 (↑ 1.32)	62.93 (↑ 1.59)
13. Baseline-BoWFire+Corsican+FLAME	3257	0	-	65.71	63.38	66.25	65.38
14. FireDM-512-BCF	0	3257	512 × 512	65.93 (↑ 0.22)	64.57 (↑ 1.19)	67.53 (↑ 1.28)	66.78 (↑ 1.40)
15. FireDM-768-BCF	0	3257	768 × 768	66.12 (↑ 0.41)	64.68 (↑ 1.30)	67.67 (↑ 1.42)	66.88 (↑ 1.50)
16. FireDM-1024-BCF	0	3257	1024 × 1024	66.32 (↑ 0.61)	64.78 (↑ 1.40)	67.75 (↑ 1.50)	66.97 (↑ 1.59)
17. FireDM-512	0	13950	512 × 512	70.21	69.87	68.43	71.47
18. FireDM-768	0	14926	768 × 768	71.38	70.96	69.56	72.63
19. FireDM-1024	0	14967	1024 × 1024	73.12	71.78	70.34	73.37
20. FireDM-512+768+1024	0	43843	-	75.23	72.67	71.54	74.12

Table C18
The segmentation results of the datasets generated by FireDM at three different image resolutions in response to severe fog or smoke conditions in fire scenes.

Dataset	Real Image	Synthetic	Image Size	IoU (%)	Pixel Accuracy (%)	F1-Score (%)	Average Precision (%)
1. Baseline-BoWFire	119	0	-	38.24	35.21	37.52	42.47
2. FireDM-512-B	0	119	512×512	39.15 (↑ 0.91)	36.34 (↑ 1.13)	38.54 (↑ 1.02)	43.55 (↑ 1.08)
3. FireDM-768-B	0	119	768×768	40.22 (↑ 1.98)	37.00 (↑ 1.79)	38.92 (↑ 1.40)	43.87 (↑ 1.40)
4. FireDM-1024-B	0	119	1024×1024	40.38 (↑ 2.14)	37.29 (↑ 2.08)	39.17 (↑ 1.65)	43.99 (↑ 1.52)
5. Baseline-Corsican	1135	0	-	42.73	43.81	40.54	44.37
6. FireDM-512-C	0	1135	512×512	43.84 (↑ 1.11)	44.65 (↑ 0.84)	41.31 (↑ 0.77)	45.25 (↑ 0.88)
7. FireDM-768-C	0	1135	768×768	43.95 (↑ 1.22)	44.74 (↑ 0.93)	41.46 (↑ 0.92)	45.37 (↑ 1.00)
8. FireDM-1024-C	0	1135	1024×1024	44.51 (↑ 1.78)	45.20 (↑ 1.39)	42.20 (↑ 1.66)	45.57 (↑ 1.20)
9. Baseline-FLAME	2003	0	3480×2160	54.29	55.12	51.31	55.07
10. FireDM-512-F	0	2003	512×512	54.59 (↑ 0.30)	55.48 (↑ 0.36)	52.15 (↑ 0.84)	55.68 (↑ 0.61)
11. FireDM-768-F	0	2003	768×768	54.78 (↑ 0.49)	55.67 (↑ 0.55)	52.27 (↑ 0.96)	55.81 (↑ 0.74)
12. FireDM-1024-F	0	2003	1024×1024	55.21 (↑ 0.92)	55.93 (↑ 0.81)	52.65 (↑ 1.34)	55.97 (↑ 0.90)
13. Baseline-BoWFire+Corsican+FLAME	3257	0	-	62.23	64.53	65.13	66.01
14. FireDM-512-BCF	0	3257	512 × 512	62.53 (↑ 0.30)	64.98 (↑ 0.45)	65.89 (↑ 0.76)	66.23 (↑ 0.22)
15. FireDM-768-BCF	0	3257	768 × 768	62.81 (↑ 0.58)	65.21 (↑ 0.68)	66.21 (↑ 1.08)	66.53 (↑ 0.52)
16. FireDM-1024-BCF	0	3257	1024 × 1024	62.95 (↑ 0.72)	65.38 (↑ 0.85)	66.65 (↑ 1.52)	67.01 (↑ 1.00)
17. FireDM-512	0	13950	512 × 512	71.10	72.32	71.84	72.95
18. FireDM-768	0	14926	768 × 768	72.03	72.57	72.11	73.16
19. FireDM-1024	0	14967	1024 × 1024	73.40	73.11	72.55	73.52
20. FireDM-512+768+1024	0	43843	-	75.31	75.65	76.13	74.92

Graphical Abstracts