Journal of
Low Power Electronics and Applications

MDPI

Article

FTFNet: Multispectral Image Segmentation

Justin Edwards and Mohamed El-Sharkawy *

Purdue School of Engineering and Technology, Indianapolis, IN 46202, USA; justedwa@iupui.edu
* Correspondence: tennis1962indy@aol.com

Abstract: Semantic segmentation is a machine learning task that is seeing increased utilization in multiple fields, from medical imagery to land demarcation and autonomous vehicles. A real-time autonomous system must be lightweight while maintaining reasonable accuracy. This research focuses on leveraging the fusion of long-wave infrared (LWIR) imagery with visual spectrum imagery to fill in the inherent performance gaps when using visual imagery alone. This approach culminated in the Fast Thermal Fusion Network (FTFNet), which shows marked improvement over the baseline architecture of the Multispectral Fusion Network (MFNet) while maintaining a low footprint.

Keywords: convolutional neural networks; semantic segmentation; sensor fusion; autonomous systems

1. Introduction

Semantic image segmentation is a classification task involving transforming an input image into a more meaningful representation upon which the system can make decisions. The goal of a system tasked to perform semantic image segmentation is to assign a class label to all pixels in an image that share similar features/properties. Typically, this would involve assigning a class label to all pixels of the detected objects of the same type in an image. The new representation distinguishes objects within the image from the background and delineates the boundaries between these objects. This process has found usefulness in aerial imagery boundary delineation, medical imagery, and autonomous vehicle applications.

Early methods of semantic image segmentation utilized techniques, such as clustering, support vector machines (SVMs), and Markov random fields. With the explosion of research within the deep learning field, researchers focusing on the semantic image segmentation task pivoted to deep learning techniques with great success [1]. They were often setting new records on popular segmentation benchmarks. Of these new deep learning techniques, CNNs stood out as being particularly successful. The fully convolutional strategy was thoroughly studied in 2015 by Long et al. culminating in the FCN architecture [2] and has since become the most widely adopted strategy for solving semantic segmentation tasks. This drove the development of many state-of-the-art architectures utilizing fully convolutional approaches, such as Segnet [3], ENet [4], UNet [5], MFNet [6], and others.

Semantic image segmentation has been utilized in the field of autonomous vehicles to make driving decisions based on environmental data typically captured via cameras mounted on the vehicle. Sensory fusion-based approaches aim to improve accuracy by utilizing multiple sensory inputs, primarily radar, cameras, and lidar. Long-wave infrared spectrum (LWIR) cameras have yet to see much use in modern sensory fusion approaches, likely due to the high cost of high-resolution data acquisition systems. However, companies such as FLIR have begun introducing low-resolution LWIR cameras onto the market. LWIR cameras used in a sensory fusion approach can compensate for some of the limitations of a purely visual spectrum camera. Such limitations can be mitigated, including low-light conditions, objects obscured by smoke/fog/precipitation, and lighting conditions, such as low sun angle.

A new type of sensory fusion utilizing LWIR and visual spectrum data can lead to higher performance in accuracy metrics. The extra data in LWIR images can compensate

check for updates

Citation: Edwards, J.; El-Sharkawy, M. FTFNet: Multispectral Image Segmentation. *J. Low Power Electron. Appl.* **2023**, *13*, 42. https://doi.org/10.3390/jlpea13030042

Received: 23 April 2023
Revised: 8 June 2023
Accepted: 20 June 2023
Published: 30 June 2023

Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

for gaps in visual spectrum data, particularly in less ideal road conditions. This extra data can be leveraged in CNN-based semantic image segmentation models, leading to high accuracy metrics and better decision-making capabilities for autonomous driving systems.

This research proposes a new architecture, the Fast Thermal Fusion Network (FTFNet), that aims to achieve real-time performance with a small footprint. In addition, FTFNet improves upon the accuracy of other high-speed state-of-the-art segmentation models, particularly MFNet. Finally, FTFNet leverages multispectral data using sensory fusion to allow FTFNet to perform in various environmental conditions. Additionally, the proposed loss function, the categorical cross-entropy dice loss, used in FTFNet is tested and shown to increase performance over the other tested loss functions.

2. Background

In the field of machine learning, researchers are always pushing for higher accuracy with a lower footprint and faster inference times. As progress is achieved, technology moves from the realm of theory into the realm of application. This is where machine learning research intersects with autonomous vehicle applications.

It has been found that fusing multiple sensory modalities has the potential to achieve greater accuracy than any one sensor modality alone [6]. This fusion technique was explored in the FuseNet architecture, which leveraged multiple encoders to extract information from different sensory modalities [7]. In this vein, the fusion of visual and infrared spectrum images into a multispectral image can be leveraged to fill in the gaps of any component sensory modalities. Multispectral Fusion Network (MFNet), proposed by [6], leveraged this sensory fusion approach to deliver a real-time, accurate semantic segmentation model.

Visual spectrum cameras have issues in conditions where precipitation, smoke, and fog exist. These conditions cause visible light to scatter when traveling through the obstructions, resulting in a loss of information. However, these obstructions do not scatter IR light as easily. Thus, one can utilize IR imagery to fill in the gaps in the visual spectrum information [8].

Similarly, visual spectrum cameras particularly struggle in low-light conditions. Due to the properties of IR, any object above absolute zero will emit IR wavelengths. This allows cameras in the MWIR and LWIR spectrums to detect these objects in absolute darkness. IR cameras are also unaffected by sun angle, which could obscure a visual spectrum camera, as they are unaffected by visual spectrum wavelengths emitted by the sun.

On the other hand, IR imagery alone does not lead to a very robust solution. IR imagery lacks the textural information that is captured in the visual spectrum. IR imagery also lacks the color information that is present in the visual spectrum. A fusion of the two sensor modalities captures the best of both sensor types and allows reliable operation in a wide variety of environmental conditions.

3. Baseline Architecture

The Multispectral Fusion Network (MFNet) is selected as the baseline architecture utilized in this research. MFNet is a network architecture designed to leverage sensory fusion in the form of multispectral data, specifically thermal imagery. One of the critical features of MFNet is its ability to perform segmentation tasks in real time. In addition to the above, MFNet has been chosen based on its ability to provide a real-time segmentation solution while maintaining a low footprint and reasonable accuracy. This is accomplished by an encoder–decoder-style neural network, with separate encoder stages for each sensory modality and then a shared decoder where features from the thermal and visual spectrum are concatenated and passed through the decoder [6]. Figures 1 and 2 detail the overall architecture and fusion method respectively.

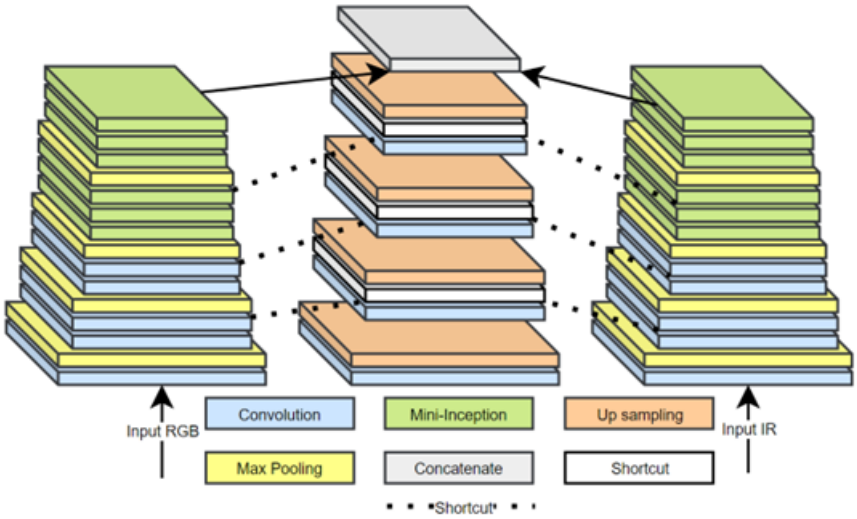


Figure 1. MFNet Architecture [6].

The encoder is comprised of five stages. The first three stages utilize convolutional layers with downsampling via max pooling to perform the initial feature extraction. Leaky ReLU is utilized as the activation function for all convolution operations. In addition, batch normalization is performed after each convolution operation. The final two stages replace the plain convolutional blocks with a mini-inception block to increase the receptive field. The mini-inception block accomplishes its increase in the receptive field by utilizing dilated convolutions with a dilation rate equal to two. The input is split into two branches, one undergoing a regular convolution operation and the other undergoing dilated convolution before the two branches are concatenated. There are skip connections in stages two through four that are connected to the corresponding decoder stages to preserve features.

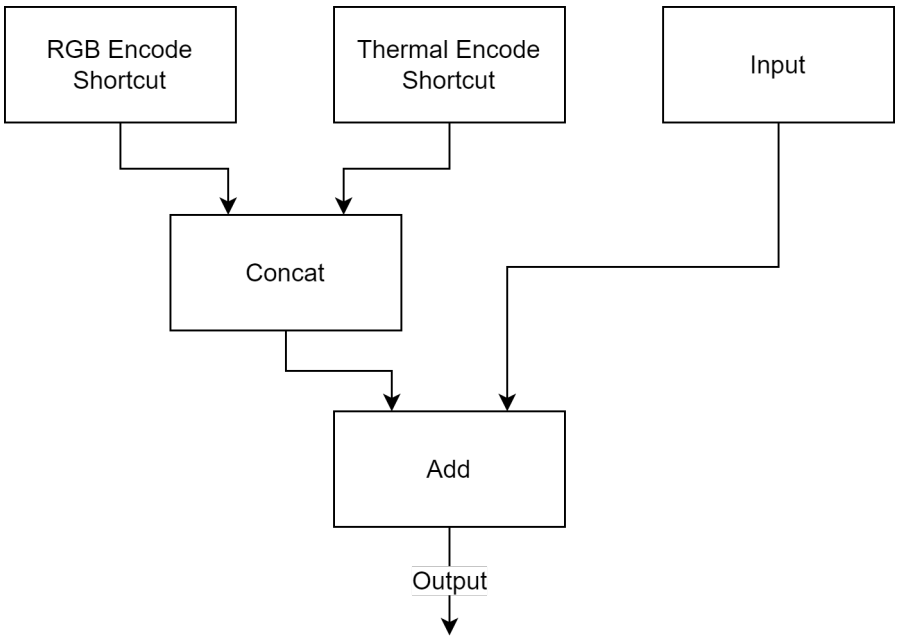


Figure 2. MFNet shortcut block.

After the encoder stage, the thermal and visual spectrum feature maps are concatenated and undergo a series of upsampling and convolution operations to restore the spatial resolution and fuse the thermal and visual information. Decoder stages two, three, and

four concatenate the skip connections from the corresponding encoder stages, and then perform an addition operation with the previous decoder stage before upsampling and convolution. Batch normalization and leaky ReLU are performed after each upsample and convolution. Once the feature maps are upsampled to the original spatial resolution of the input, the maps are passed to a final convolution layer with softmax activation to perform the final image segmentation. Details of the MFNet architecture used in this research can be found in Table 1.

Table 1. MFNet Architectural Details.

Encoder Design					
		RGB		Thermal	
Stage	Type	Input Size	Output Size	Input Size	Output Size
Stage 1	Conv 3 × 3	480 × 640 × 3	480 × 640 × 16	480 × 640 × 1	480 × 640 × 16
	Max Pooling 2 × 2	480 × 640 × 16	240 × 320 × 16	480 × 640 × 16	240 × 320 × 16
Stage 2	Conv 3 × 3	240 × 320 × 48	240 × 320 × 48	240 × 320 × 16	240 × 320 × 16
	Conv 3 × 3	240 × 320 × 48	240 × 320 × 48	240 × 320 × 16	240 × 320 × 16
	Max Pooling 2 × 2	240 × 320 × 48	120 × 160 × 48	240 × 320 × 16	120 × 160 × 16
Stage 3	Conv 3 × 3	120 × 160 × 48	120 × 160 × 48	120 × 160 × 16	120 × 160 × 16
	Conv 3 × 3	120 × 160 × 48	120 × 160 × 48	120 × 160 × 16	120 × 160 × 16
	Max Pooling 2 × 2	120 × 160 × 48	60 × 80 × 48	120 × 160 × 16	60 × 80 × 16
Stage 4	Mini-inception	60 × 80 × 48	60 × 80 × 96	60 × 80 × 16	60 × 80 × 32
	Mini-inception	60 × 80 × 96	60 × 80 × 96	60 × 80 × 32	60 × 80 × 32
	Mini-inception	60 × 80 × 96	60 × 80 × 96	60 × 80 × 32	60 × 80 × 32
	Max Pooling 2 × 2	60 × 80 × 96	30 × 40 × 96	60 × 80 × 32	30 × 40 × 32
Stage 5	Mini-inception	30 × 40 × 96	30 × 40 × 96	30 × 40 × 32	30 × 40 × 32
	Mini-inception	30 × 40 × 96	30 × 40 × 96	30 × 40 × 32	30 × 40 × 32
	Mini-inception	30 × 40 × 96	30 × 40 × 96	30 × 40 × 32	30 × 40 × 32
Decoder Design					
Stage	Type	Input Size	Output Size		
Stage 1	Concatenate	30 × 40 × 96 + 30 × 40 × 32	30 × 40 × 128		
Stage 2	Upsample	30 × 40 × 128	60 × 80 × 128		
	Shortcut	60 × 80 × 128	60 × 80 × 128		
	Conv 3 × 3	60 × 80 × 128	60 × 80 × 64		
Stage 3	Upsample	60 × 80 × 64	120 × 160 × 64		
	Shortcut	120 × 160 × 64	120 × 160 × 64		
	Conv 3 × 3	120 × 160 × 64	120 × 160 × 64		
Stage 4	Upsample	120 × 160 × 64	240 × 320 × 64		
	Shortcut	240 × 320 × 64	240 × 320 × 64		
	Conv 3 × 3	240 × 320 × 64	240 × 320 × 64		
Stage 5	Upsample	240 × 320 × 64	480 × 640 × 64		
	Conv 3 × 3	480 × 640 × 64	480 × 640 × 32		
	Conv 3 × 3	480 × 640 × 32	480 × 640 × N		

4. FTFNet Architecture

FTFNet leverages techniques proposed in MFNet with additional modifications to improve performance while maintaining a low footprint in terms of parameter count. Due to the selection of MFNet as a baseline, it will henceforth be referred to as the baseline architecture. Some of the modifications were inspired by architectures, such as MobileNet [9], PSPNet [10], and AMFuse [11]. Details of the modifications to the baseline architecture are outlined below. Like MFNet, FTFNet is comprised of separate encoders for thermal and visual spectrum inputs. The inputs undergo a series of convolutional operations before downsampling. After each downsample, a skip connection is placed between the encoder and the corresponding decoder blocks. After passing through the encoder portion, the branches are concatenated and undergo a series of upsample, fusion, and convolutional operations before being passed to a final convolutional layer with softmax activation. Like MFNet, FTFNet adopts a late fusion strategy via skip connections between the encoder and decoder portions of the network. The architectural elements are described in detail in the following sections; Figure 3 presents an overview of FTFNet's architecture.

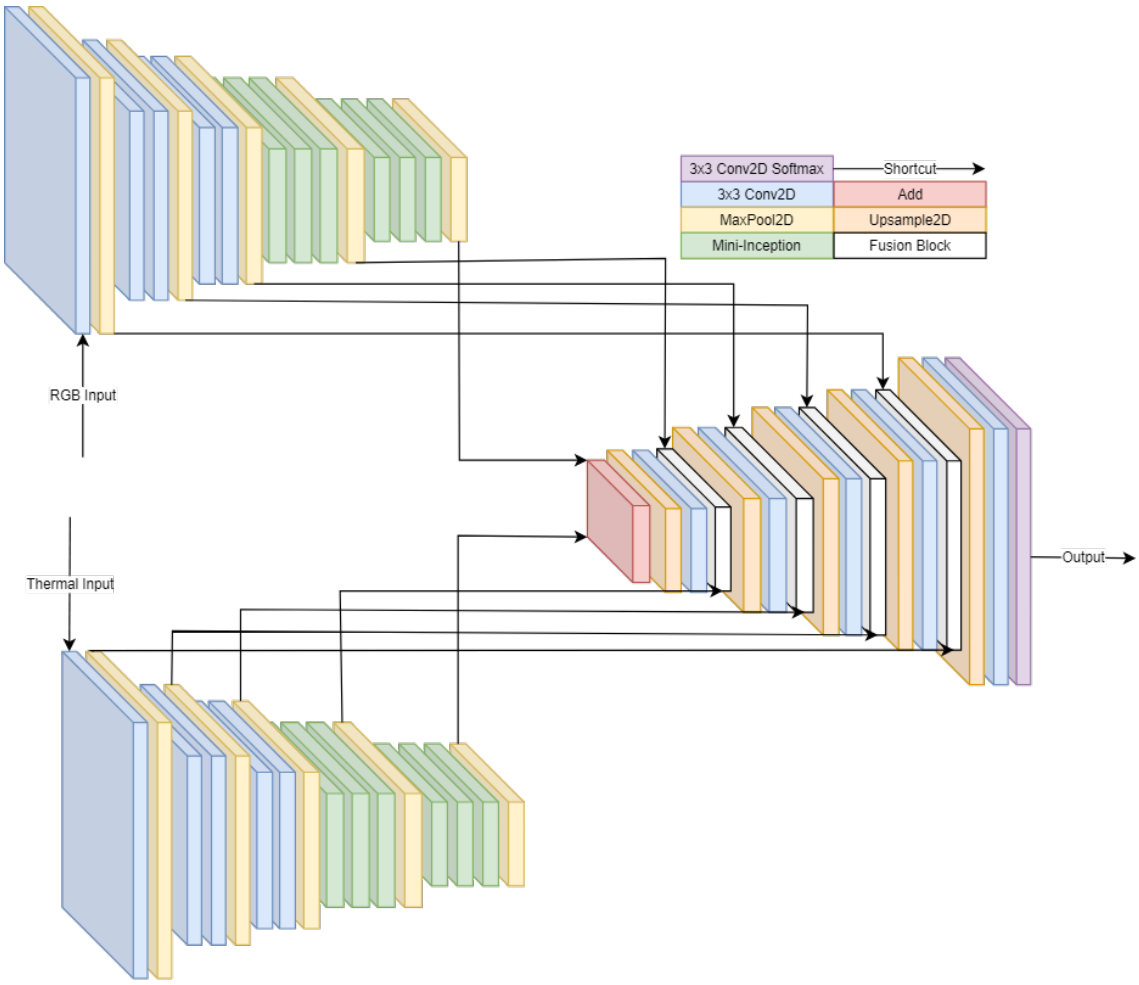


Figure 3. FTFNet Architecture.

4.1. Encoders

FTFNet consists of two encoders that perform feature extraction on each sensory modality. Unlike MFNet, FTFNet utilizes a symmetrical encoder architecture in which all sensory modalities share a common encoder architecture. The RGB encoder takes an input shape of $H \times W \times 3$, while the thermal encoder is designed for gray-scale thermal images with a shape of $H \times W \times 1$. Each input is passed through a normalization function before entering the first stage of the encoder. Encoding in FTFNet is broken down into five stages.

Each stage consists of convolutional operations used for feature extraction, followed by downsampling via max pooling. Unlike MFNet, downsampling occurs at each stage of the encoder.

The first stage consists of the basic convolutional block with max pooling used in FTFNet. This block performs a 3×3 convolutional operation, followed by batch normalization and leaky ReLU. Max pooling is then performed to reduce the spatial dimensions. After each stage, a skip connection will be inserted for feature fusion in the decoder stages. The following two stages have similar architecture but utilize one basic convolutional block and one basic convolutional block with max pooling. The details of the basic convolutional block and basic convolutional block with max pooling are shown in Figure 4.

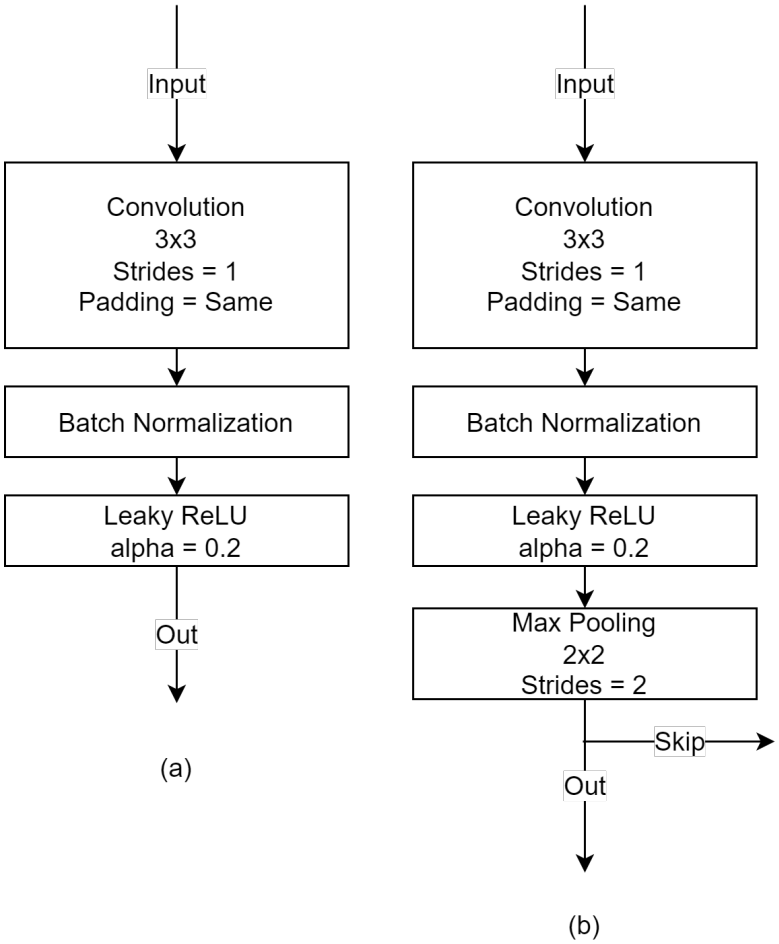


Figure 4. (**a**) Base convolution block. (**b**) Base convolution block with max pooling.

Instead of a basic convolutional block, stages four and five utilize the squeeze-and-excitation (SE) mini-inception module, see Figure 5. This module utilizes dilated convolutions to increase the receptive field in the deeper layers of the network; this mini-inception module was introduced in [6]. The module consists of two branches; the first undergoes a regular 3×3 convolution operation, while the second undergoes a 3×3 dilated convolution with a dilation rate equal to two. The convolutions in each branch are followed by batch normalization and leaky ReLU. The two branches are then concatenated. In FTFNet, this mini-inception module is followed by a channel attention block in the form of squeeze and excitation. Both stages four and five utilize three of the aforementioned mini-inception plus channel attention modules followed by max pooling. FTFNet encoder details are outlined in Table 2.

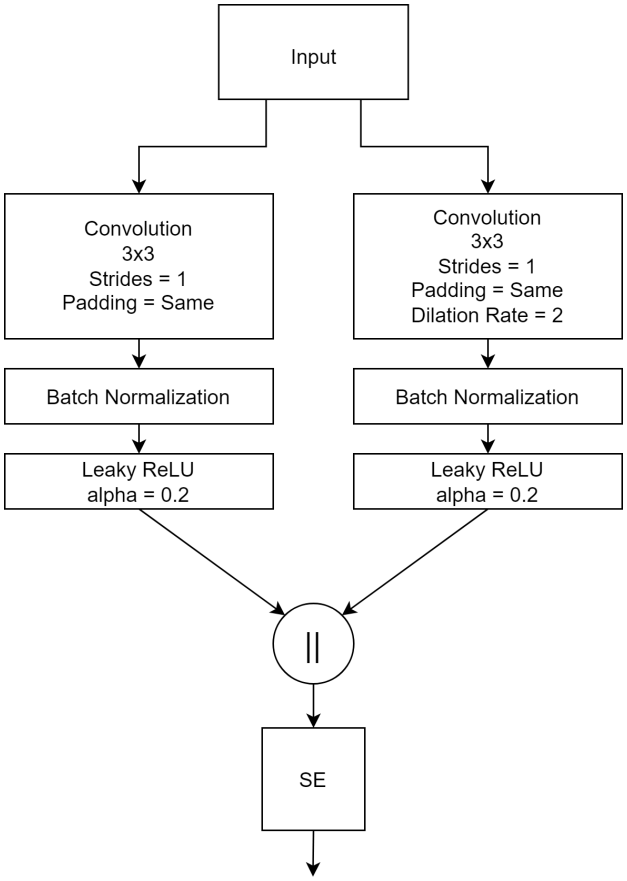


Figure 5. SE Mini-inception Module.

Table 2. FTFNet Encoder Details.

Encoder Design			
Stage	**Type**	**Input Size**	**Output Size**
Stage 1	Conv 3 × 3	480 × 640 × (1/3) *	480 × 640 × 16
	Max Pooling 2 × 2	480 × 640 × 16	240 × 320 × 16
Stage 2	Conv 3 × 3	240 × 320 × 48	240 × 320 × 48
	Conv 3 × 3	240 × 320 × 48	240 × 320 × 48
	Max Pooling 2 × 2	240 × 320 × 48	120 × 160 × 48
Stage 3	Conv 3 × 3	120 × 160 × 48	120 × 160 × 48
	Conv 3 × 3	120 × 160 × 48	120 × 160 × 48
	Max Pooling 2 × 2	120 × 160 × 48	60 × 80 × 48
Stage 4	Mini-inception	60 × 80 × 48	60 × 80 × 96
	Mini-inception	60 × 80 × 96	60 × 80 × 96
	Mini-inception	60 × 80 × 96	60 × 80 × 96
	Max Pooling 2 × 2	60 × 80 × 96	30 × 40 × 96
Stage 5	Mini-inception	30 × 40 × 96	30 × 40 × 96
	Mini-inception	30 × 40 × 96	30 × 40 × 96
	Mini-inception	30 × 40 × 96	30 × 40 × 96
	Max Pooling 2 × 2	30 × 40 × 96	15 × 20 × 96

* Channel dimension of 1 or 3 is utilized for thermal and RGB encoders respectively.

4.2. Decoder

Like the encoder, the decoder of FTFNet consists of five stages. The first stage computes the element-wise addition of the output of the two encoders, and then upsamples the result. The upsampled result then undergoes a 3×3 convolutional layer to produce denser features. As in the encoder, the convolutional operation is followed up by batch normalization and leaky ReLU. Stages two to five utilize the skip connections introduced in the encoder. The thermal and RGB skip connections pass through a fusion block, detailed in the next section. The output of the feature fusion block is then concatenated with the previous stage's output before undergoing the same upsampling and convolution operations described in the previous stage. As data flows through the decoder stages, the features become denser and more refined. The final layer of the decoder passes the output of stage five through a 3×3 convolutional layer with softmax activation to perform classification. FTFNet decoder details are outlined in Table 3.

Table 3. FTFNet Decoder Details.

Decoder Design			
Stage	**Type**	**Input Size**	**Output Size**
Stage 1	Add	$15 \times 20 \times 96$	$15 \times 20 \times 96$
	Upsample2D	$15 \times 20 \times 96$	$30 \times 40 \times 96$
	Conv 3×3	$30 \times 40 \times 96$	$30 \times 40 \times 64$
Stage 2	Fusion Block	$30 \times 40 \times 64$	$30 \times 40 \times 64$
	Upsample2D	$30 \times 40 \times 64$	$60 \times 80 \times 64$
	Conv 3×3	$60 \times 80 \times 64$	$60 \times 80 \times 64$
Stage 3	Fusion Block	$60 \times 80 \times 64$	$60 \times 80 \times 64$
	Upsample2D	$60 \times 80 \times 64$	$120 \times 160 \times 64$
	Conv 3×3	$120 \times 160 \times 64$	$120 \times 160 \times 64$
Stage 4	Fusion Block	$120 \times 160 \times 64$	$120 \times 160 \times 32$
	Upsample2D	$120 \times 160 \times 32$	$240 \times 320 \times 32$
	Conv 3×3	$240 \times 320 \times 32$	$240 \times 320 \times 32$
Stage 5	Fusion Block	$240 \times 320 \times 32$	$240 \times 320 \times 32$
	Upsample2D	$240 \times 320 \times 32$	$480 \times 640 \times 32$
	Conv 3×3	$480 \times 640 \times 32$	$480 \times 640 \times 32$
	Conv 3×3, Softmax	$480 \times 640 \times 32$	$480 \times 640 \times$ N

4.3. Feature Fusion Block

MFNet proposes connections between the encoder and decoder after subsampling operations in the encoder portion of the network for each modality; these skip connections are then connected to the decoder via an add operation before upsampling. In order to retain the maximum amount of useful information between modalities, the feature fusion block utilized in FTFNet performs both element-wise addition and element-wise multiplication between the two modalities, then concatenating the results. The reasoning behind this is the retention of complementary information (element-wise multiplication) and enhancement of common information between modalities (element-wise addition) [11]. FTFNet proposes a new fusion block with squeeze-and-excitation blocks applied to both inputs to perform channel attention by enhancing important features and attenuating redundant information before being fused [12]. Additionally, pyramid pooling is added to the fusion block after concatenation of the channels to extract additional contextual information between modalities at different scales [10]. Due to the element-wise add/multiply operations requiring the same tensor shape between modalities, the encoder is modified to be symmetrical between

the thermal and visual encoders. See Figure 6 for the details of the FTFNet feature fusion block.

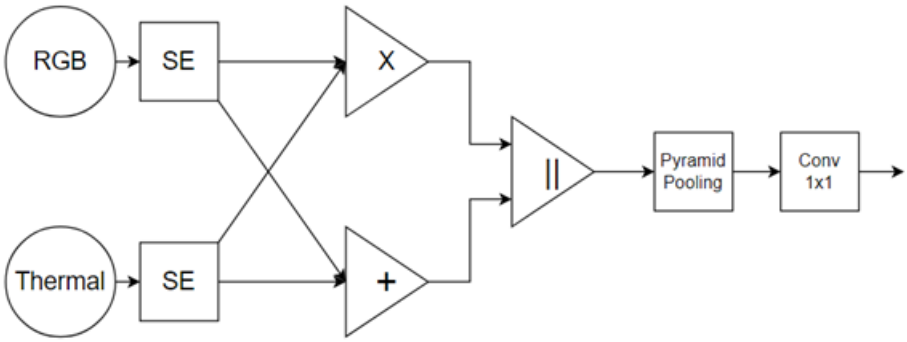


Figure 6. FTFNet Feature Fusion Block.

4.4. FTFNet Lite

Two additional FTFNet derivatives are created by utilizing the depthwise separable convolution layer first described in [9] for use in MobileNet. These derivatives see a significant reduction in parameter count due to using the depthwise separable convolution block. FTFNet lite 1 utilizes depthwise separable convolution instead of regular 3 × 3 convolutional operations in stages 2–5 of the encoder. This includes replacing the convolutional operations in the mini-inception block with depthwise separable convolution. FTFNet lite 2 only replaces the convolutional layers within the mini-inception module (stages 4 and 5) with depthwise separable convolutional layers. The details of the implementation of the depthwise separable convolution block and the FTFNet lite encoder architectures are outlined in Figure 7 and Table 4.

Table 4. FTFNet Lite Encoder Details.

Encoder Design			
Stage	**Type**	**Input Size**	**Output Size**
Stage 1	Conv 3 × 3	480 × 640 × (1/3) *	480 × 640 × 16
	Max Pooling 2 × 2	480 × 640 × 16	240 × 320 × 16
Stage 2	DW Conv 3 × 3	240 × 320 × 48	240 × 320 × 48
	DW Conv 3 × 3	240 × 320 × 48	240 × 320 × 48
	Max Pooling 2 × 2	240 × 320 × 48	120 × 160 × 48
Stage 3	DW Conv 3 × 3	120 × 160 × 48	120 × 160 × 48
	DW Conv 3 × 3	120 × 160 × 48	120 × 160 × 48
	Max Pooling 2 × 2	120 × 160 × 48	60 × 80 × 48
Stage 4	Mini-inception DW	60 × 80 × 48	60 × 80 × 96
	Mini-inception DW	60 × 80 × 96	60 × 80 × 96
	Mini-inception DW	60 × 80 × 96	60 × 80 × 96
	Max Pooling 2 × 2	60 × 80 × 96	30 × 40 × 96
Stage 5	Mini-inception DW	30 × 40 × 96	30 × 40 × 96
	Mini-inception DW	30 × 40 × 96	30 × 40 × 96
	Mini-inception DW	30 × 40 × 96	30 × 40 × 96
	Max Pooling 2 × 2	30 × 40 × 96	15 × 20 × 96

* Channel dimension of 1 or 3 is utilized for thermal and RGB encoders respectively.

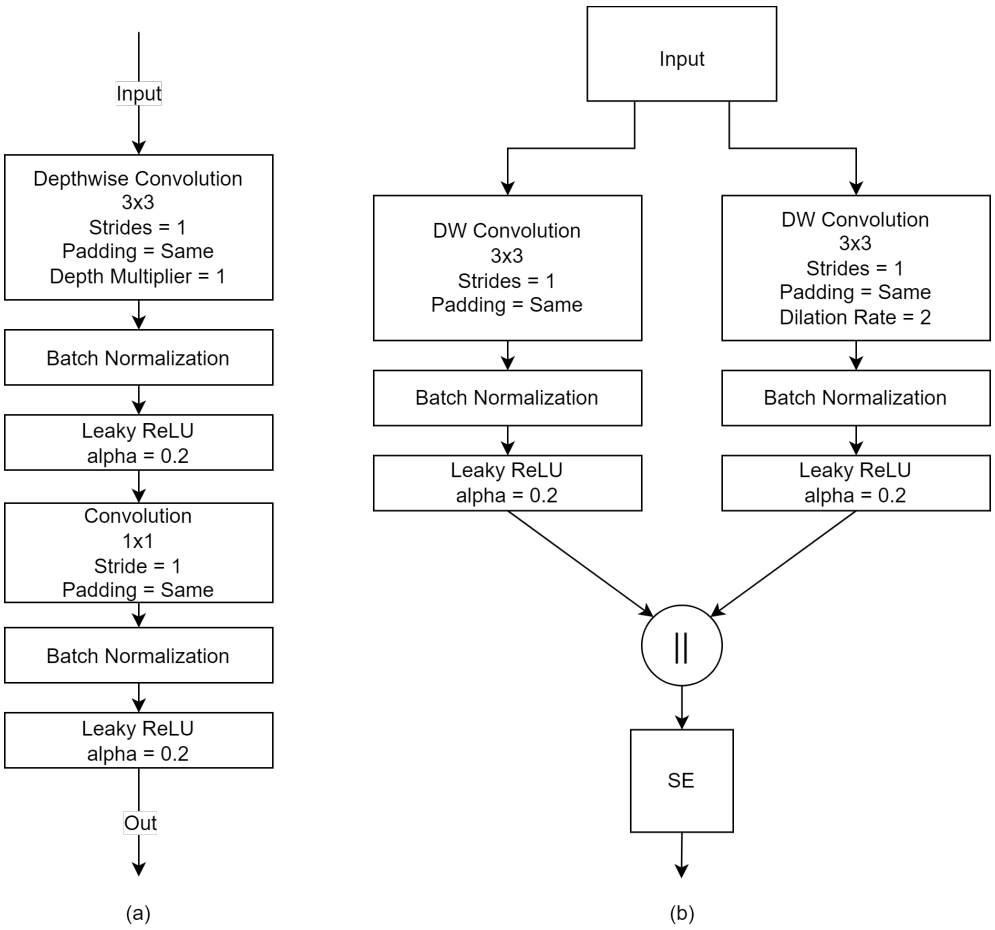


Figure 7. (**a**) Depthwise Separable (DW) Basic Convolution Block. (**b**) Depthwise Separable SE Mini-Inception Block.

4.5. Optimizer Selection

Two additional optimization methods are tested against the baseline optimization method proposed in [6] utilizing the MSRS dataset introduced in [13]. The following optimization methods are tested: SGD, RMSProp, and ADAM. Each optimizer is tested using the built-in TensorFlow optimizer library. SGD, with learning rate 0.01, is the baseline optimizer used in the training of the MFNet baseline. The training setup for this experiment can be found in Table 5.

Table 5. Optimizer selection experiment training setup.

Model	FTFNet
Dataset	MSRS Dataset
Batch Size	6
Optimizer	ADAM, SGD, RMSProp
Loss Function	Categorical Cross-Entropy
Metrics	mIoU, Categorical Accuracy

After training for 80 epochs, the models are tested using an evaluation dataset. The results are outlined in Table 6 below.

SGD, a non-adaptive algorithm, performs the worst out of the optimizers selected. ADAM and RMSProp have similar performance in terms of categorical accuracy at 96.64% and 96.75%, respectively. Regarding the mIoU score, ADAM and RMSProp also perform similarly, with ADAM having a slightly higher score at 42.79%.

Table 6. Optimizer selection results; bold indicates best result.

Optimizer	mIoU	Categorical Accuracy
ADAM	**42.79%**	96.64%
RMSProp	41.14%	**96.75%**
SGD	27.39%	94.90%

4.6. Loss Function Selection

Three different loss functions are tested when training FTFNet. Dice loss, categorical cross-entropy, and a combo loss consisting of categorical cross-entropy and dice loss are tested. The combo loss function is based on the binary cross-entropy dice loss function proposed in [14]. The proposed combo loss function tested replaces binary cross-entropy with categorical cross-entropy in the following equation:

$$CCEDL(y,(\hat{y})) = \frac{CCE(y,\hat{y})}{2 \times (1 - DC(y,\hat{y}))} \quad (1)$$

where *CCEDL* is the categorical cross-entropy dice loss, *CCE* is the categorical cross-entropy function, *DC* is the dice coefficient function, and y and $\hat{y}$ are the ground truth and prediction, respectively. Both the dice loss and combo loss functions have a custom implementation, while categorical cross-entropy loss utilizes built-in TensorFlow loss functions. The training setup and results are outlined in Tables 7 and 8 below.

Table 7. Combo loss function experiment training setup.

Model	FTFNet
Dataset	MIL-Coaxial Dataset
Batch Size	6
Optimizer	ADAM
Loss Function	Categorical Cross-Entropy, Dice Loss, Combo Loss
Metrics	mIoU, Categorical Accuracy

Table 8. Combo loss experiment results; bold indicates best result.

Loss Function	mIoU	Categorical Accuracy
Categorical Cross-entropy	50.68%	52.35%
Dice Loss	14.31%	71.63%
Combo Loss	**52.32%**	**84.11%**

The experimental results show that the dice loss outperforms the categorical cross-entropy in categorical accuracy, while the categorical cross-entropy outperforms in the mIoU score. However, the combo loss function leverages the strengths of both dice and categorical cross-entropy loss, significantly improving the mIoU and categorical accuracy compared to its constituent's loss functions. Compared to dice loss, there is an improvement of 27.4% and 14.84% in the mIoU and categorical accuracy, respectively. At the same time, it outperforms categorical cross-entropy by 3.13% and 37.76% in the mIoU and categorical accuracy.

5. Training Setup

In order to provide a solid basis for comparison, FTFNet and FTFNet lite were tested against the baseline architecture, MFNet, and UNet. UNet is a commonly utilized encoder–decoder semantic segmentation network architecture. The UNet architecture adopted an

early fusion approach where the thermal and visual spectrum images were concatenated at the input layer.

5.1. Dataset

FTFNet and the baseline MFNet were trained on the MIL-Coaxials dataset [15]. It comprises 17,023 image and segmentation mask pairs captured using an FIRplus coaxial camera. The images are captured in a plethora of environmental conditions, including clear sky, cloudy, rainy, snowy, indoor, evening, and night. The resolutions of the visual and thermal images are 1280 × 1024 and 320 × 256, respectively. When training with the MIL-Coaxials dataset, the dataset was split into the following segments outlined in Table 9 below.

Table 9. MIL-Coaxial Dataset Train/Test/Validate Splits.

Split	Train	Validation	Test
Dataset Size	3000	500	500
Percentage	75%	12.50%	12.50%

In addition, the smaller Multispectral Road Scenarios (MSRS) dataset [13] was utilized to test and select an optimization function for FTFNet. It consists of 820 daytime and 749 nighttime images recorded with an InfRec R500 coaxial camera with a resolution of 480 × 640 pixels. Details of the MSRS dataset split can be found in Table 10.

Table 10. Multispectral Road Scenarios Semantic Segmentation Dataset Train/Test/Validate Splits.

Split	Train	Validation	Test
Dataset Size	1568	392	393
Percentage	66.7%	16.65%	16.65%

5.2. Hyperparameters

The input resolutions for all the models were selected as 480 × 640 as this is the resolution proposed in [6], which outlines the baseline architecture used for this research. For datasets with resolutions higher/lower than the selected input resolution, the images were scaled during image preprocessing. A batch size of six was selected as it did not significantly affect the training of the models other than reducing the training time. It was the maximum batch size allowable due to the GPU memory constraints. The training was performed for 80 epochs as the validation accuracy and mean intersection over union metrics converged and did not improve with additional training over 80 epochs. Where utilized, the batch normalization parameters momentum and epsilon were set to 0.1 and 0.00001, respectively. In addition, leaky ReLU utilized an alpha value equivalent to 0.2. ADAM was selected as the optimization function. All the training runs utilized categorical cross-entropy loss unless combo loss is indicated. The combo loss function, in this case, is the proposed categorical cross-entropy dice loss function. The evaluation of the different architectures was performed utilizing a 500-image test set.

- NVIDIA Tesla T4.
- 55 GB RAM.
- Intel Xeon 2.00 GHz, 4 Cores, 8 Threads.
- Google Colaboratory Virtual Environment.

5.3. Software Packages

- Ubuntu 18.04 LTS.
- CUDA 11.2.
- TensorFlow 2.9.2.
- Python 3.8.

- TensorFlow Datasets.
- TensorFlow Addons.

6. Results

FTFNet and the FTFNet lite variants were tested against MFNet and UNet. This provides testing exposure to two different sensory fusion methods in state-of-the-art architectures. Parametric results can be found in Figure 8 and Table 11, while visual results can be found in Figure 9. MFNet utilizes a late fusion approach where information from the two sensory modalities is extracted separately and fused later in the architecture. In this case, the fusion begins at the first stage of the decoder. UNet, on the other hand, adopts an early fusion approach in which the two modalities are immediately fused at the beginning stages of the architecture.

FTFNet shows a 7.05% increase in the mIoU score but an 8.13% decrease in categorical accuracy compared to the baseline architecture. The loss in categorical accuracy does not necessarily point to a decrease in the segmentation quality due to the data's highly imbalanced nature. The MIL-Coaxial dataset images are primarily comprised of background pixels which can cause misleading results in terms of categorical accuracy, a measure of the total correctly classified pixels. MFNet, FTFNet, and the FTFNet lite variants outperform the UNet model with early fusion. FTFNet trained with combo loss shows the most significant improvement over the baseline with an improvement of an 8.69% mIoU score and a 23.63% increase in categorical accuracy. This is achieved with a minimal increase in the parameter count of 1 M parameters. The FTFNet lite 1 and 2 variants also outperform the MFNet baseline architecture with an increase in the mIoU of 4.4% and 5.02%, respectively, while maintaining a similar parameter count of 0.8 M and 0.92 M. The larger parameter count between FTFNet and MFNet did increase the latency when processing images. MFNet demonstrates a latency of 21 ms compared to 34 ms with FTFNet. FTFNet and its variants all perform similarly in latency, with both FTFNet lite variants showing an improvement of 3 ms. UNet performs the worst out of the architectures tested with a latency of 104 ms; this is to be expected as UNet has a significantly larger parameter count compared to FTFNet and MFNet.

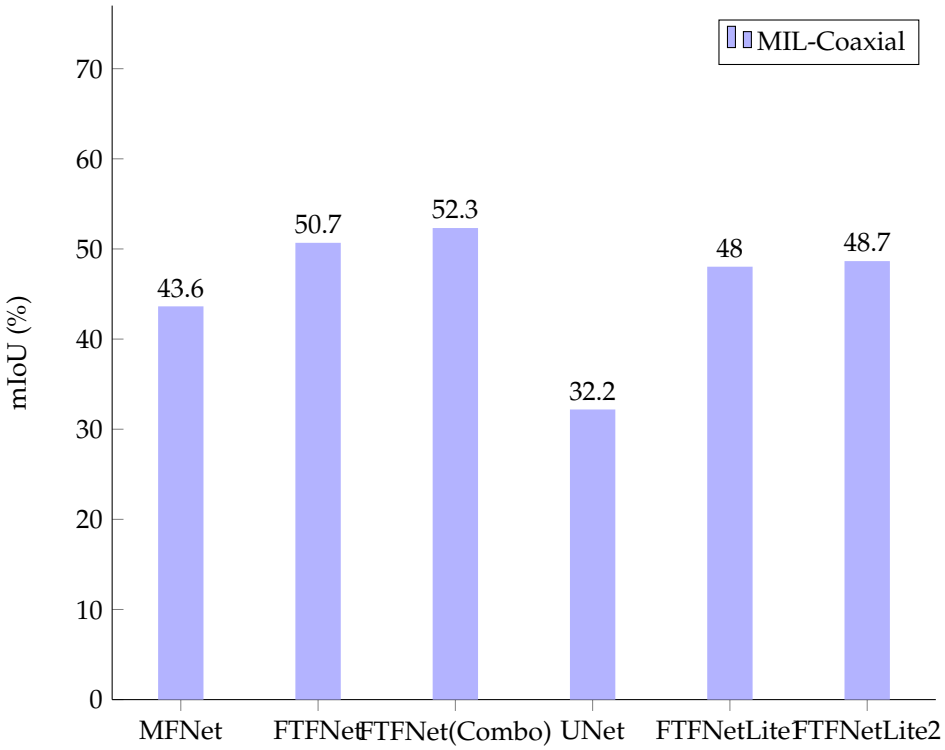


Figure 8. mIoU Performance of Tested Architectures, Tested on MIL-Coaxial Datasets.

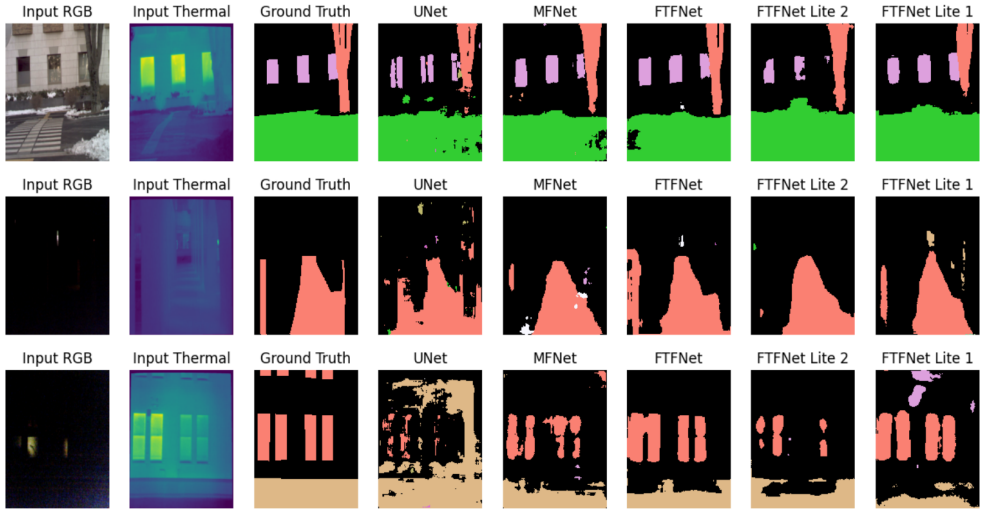


Figure 9. Segmentation results of all models tested.

Table 11. FTFNet and FTFNet lite results when trained on MIL-Coaxial dataset; bold indicates best results.

Model	mIoU	Categorical Accuracy	Parameters	Latency
MFNet (Baseline)	43.63%	60.48%	**0.74 M**	**21 ms**
FTFNet	50.68%	52.35%	1.70 M	34 ms
FTFNet (Combo Loss)	**52.32%**	**84.11%**	1.70 M	34 ms
UNet	32.19%	72.65%	34.5 M	104 ms
FTFNet Lite 1	48.03%	53.17%	0.80 M	31 ms
FTFNet Lite 2	48.65%	54.63%	0.92 M	31 ms

7. Conclusions

The objective of this research was to investigate the viability of lightweight multispectral fusion networks for semantic segmentation and introduce improvements upon existing architectures to increase segmentation quality without significant increases to the parameter count. This was achieved by introducing a symmetrical encoder, the SE block, pyramid pooling, and a new late fusion module to maintain model depth while maintaining a low footprint in terms of parameter count. The fusion block leveraged the SE block to repress redundant features and highlight essential features in their respective encoder branches. The add–multiply–concatenate operations were utilized to retain important information in each sensor modality. Finally, the fusion block used pyramid pooling to extract features at different scales. These improvements allowed FTFNet to extract more pertinent information from the multispectral data presented to the architecture, leading to increased performance. The SE block was also utilized with the mini-inception block to enhance essential features in each encoder, providing a bump to performance on single modality data. Additional improvements were also realized through the proper selection of loss and optimization functions. The proposed categorical cross-entropy dice loss was shown to be a more effective loss function than the baseline categorical cross-entropy function with an improvement of 3.13% and 37.76% in the mIoU and categorical accuracy, respectively. Optimization and loss function selection were more generalized improvements whose improvement does not rely on the multispectral nature of FTFNet.

This culminated in an architecture, FTFNet, that outperformed the baseline MFNet when trained on the MIL-Coaxial dataset. When trained on the MIL-Coaxial dataset, FTFNet improved the mIoU by 8.69% and categorical accuracy by 23.63% over MFNet. These results were accomplished by only increasing the parameter count from 0.7 M to 1.7 M parameters. With the minor increase in parameter count, the latency increased to 34 ms versus the 21 ms latency achieved with MFNet.

With FTFNet outperforming the baseline model on the MIL-Coaxial dataset, parameter reduction techniques were utilized, resulting in the FTFNet lite variants. FTFNet lite 1 and 2 achieved a parameter reduction of 47% and 54%, respectively, when compared to FTFNet. FTFNet lite 1 and 2 showed improvements of 4.4% and 5.0% in the mIoU score when trained on the MIL-Coaxial dataset. FTFNet lite 2 showed the most significant promise in terms of the trade-off between the parameter count and increased mIoU score.

Based on the above data and observations, there is an ideal configuration for the architecture, loss function, and optimizer selection. In all cases, the categorical cross-entropy dice loss outperformed the alternatives explored and should be utilized. Similarly, ADAM was shown to be the best optimizer selection out of those tested. However, other adaptive optimization algorithms should perform similarly, or in some cases better, though this needs to be confirmed.

8. Future Scope

Though the proposed architecture has significantly improved upon the baseline, additional techniques could be explored to improve the quality of segmentation further. The following list highlights some potential avenues for additional improvement:

- Model backbone: The MFNet backbone proposed in [6] was utilized and modified in this research. However, different backbones could be selected that might offer additional improvement.
- Additional sensor modalities: In this research, only RGB and thermal data were utilized; additional sensory modalities could be fused using the FTFNet framework by adding additional encoder branches for other modalities and then modifying the decoder accordingly. Lidar or radar data could be targeted as additional sensory modalities.
- Hardware deployment: As mentioned in the introduction, this research aimed to create lightweight segmentation models that can be deployed in ADAS systems. Model quantization and deployment on embedded hardware would be necessary for evaluating FTFNet's real-world performance.
- Model size reduction: FTFNet lite variants were proposed by implementing depthwise separable convolutional layers to reduce the parameter count and computational complexity. Additional techniques could be used to further reduce the footprint of FTFNet while maintaining the improvements in the mIoU and accuracy.

Author Contributions: Writing—original draft, J.E.; Writing—review & editing, M.E.-S. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The MIL-Coaxial and Multispectral Road Scenarios datasets used in this research are publicly available and can be found at the following links: MIL-Coaxial https://github.com/mil-tokyo/coaxials & Multispectral Road Scenarios https://www.mi.t.u-tokyo.ac.jp/static/projects/mil_multispectral/.

Acknowledgments: I would like to thank my colleagues at Indiana University Purdue University Indianapolis for their helpful feedback and support. In particular, I would like to thank Professor Mohamed El-Sharkawy for his valuable contributions to my research.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

LWIR	Long-Wave Infrared
SVM	Support Vector Machine
ADAS	Autonomous Driving Assistance System
SGD	Stochastic Gradient Descent
mIoU	Mean Intersection over Union
FCN	Fully Convolutional Network
ResNet	Residual Network
ICNet	Image Cascade Network
AMFuse	Add Multiply Fusion
ANN	Artificial Neural Network
CNN	Convolutional Neural Network
MFNet	Multispectral Fusion Network
FTFNet	Fast Thermal Fusion Network
PSPNet	Pyramid Scene Parsing Network
ENet	Efficient Network

References

1. Matcha, A.C.N. A 2021 Guide to Semantic Segmentation. Available online: https://nanonets.com/blog/semantic-image-segmentation-2020/ (accessed on 15 February 2023).
2. Long, J.; Shelhamer, E.; Darrell, T. Fully Convolutional Networks for Semantic Segmentation. *arXiv* **2015**, arXiv:1411.4038.
3. Badrinarayanan, V.; Kendall, A.; Cipolla, R. SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation. *arXiv* **2016**, arXiv:1511.00561.
4. Paszke, A.; Chaurasia, A.; Kim, S.; Culurciello, E. ENet: A Deep Neural Network Architecture for Real-Time Semantic Segmentation. *arXiv* **2016**, arXiv:1606.02147.
5. Ronneberger, O.; Fischer, P.; Brox, T. U-Net: Convolutional Networks for Biomedical Image Segmentation. *arXiv* **2015**, arXiv:1505.04597.
6. Ha, Q.; Watanabe, K.; Karasawa, T.; Ushiku, Y.; Harada, T. MFNet: Towards real-time semantic segmentation for autonomous vehicles with multi-spectral scenes. In Proceedings of the 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vancouver, BC, Canada, 24–28 September 2017; pp. 5108–5115, ISSN 2153-0866. [CrossRef]
7. Hazirbas, C.; Ma, L.; Domokos, C.; Cremers, D. FuseNet: Incorporating Depth into Semantic Segmentation via Fusion-Based CNN Architecture. In Proceedings of the Asian Conference on Computer Vision, Taipei, Taiwan, 20–24 November 2016.
8. Teledyne. Can Thermal Imaging See through Fog and Rain? Available online: https://www.flir.com/discover/rd-science/can-thermal-imaging-see-through-fog-and-rain/ (accessed on 16 February 2023).
9. Howard, A.G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; Adam, H. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *arXiv* **2017**, arXiv:1704.04861.
10. Zhao, H.; Shi, J.; Qi, X.; Wang, X.; Jia, J. Pyramid Scene Parsing Network. Available online: http://xxx.lanl.gov/abs/1612.01105 (accessed on 23 January 2023).
11. Liu, H.; Chen, F.; Zeng, Z.; Tan, X. AMFuse: Add–Multiply-Based Cross-Modal Fusion Network for Multi-Spectral Semantic Segmentation. *Remote Sens.* **2022**, *14*, 3368. [CrossRef]
12. Hu, J.; Shen, L.; Albanie, S.; Sun, G.; Wu, E. Squeeze-and-Excitation Networks. Available online: http://xxx.lanl.gov/abs/1709.01507 (accessed on 30 January 2023).
13. Tang, L.; Yuan, J.; Zhang, H.; Jiang, X.; Ma, J. PIAFusion: A progressive infrared and visible image fusion network based on illumination aware. *Inf. Fusion* **2022**, *83*, 79–92. [CrossRef]
14. Jadon, S. A survey of loss functions for semantic segmentation. In Proceedings of the 2020 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB), Vina del Mar, Chile, 27–29 October 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 1–7. [CrossRef]
15. Okazawa, A.; Takahata, T.; Harada, T. Simultaneous Transparent and Non-Transparent Object Segmentation with Multispectral Scenes. In Proceedings of the 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Macau, China, 3–8 November 2019; pp. 4977–4984, ISSN 2153-0858. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.