In [1]:

import kagglehub # Download latest version path = kagglehub.dataset\_download("elmadafri/the-wildfire-dataset") print("Path to dataset files:", path)

Path to dataset files: /kaggle/input/the-wildfire-dataset

In [2]:

# Importing necessary libraries import os import numpy as np import matplotlib.pyplot as plt import tensorflow as tf from tensorflow.keras.preprocessing.image import ImageDataGenerator from tensorflow.keras.models import Sequential from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout, Input

In [3]:

# TO ensure the GPU is present physical\_devices = tf.config.experimental.list\_physical\_devices('GPU') if len(physical\_devices) > 0: tf.config.experimental.set\_memory\_growth(physical\_devices[0], True) print('GPU is Available!') else: print('GPU is Unavailable!')

GPU is Available!

In [4]:

# Load and explore the dataset train\_dir = '/kaggle/input/the-wildfire-dataset/the\_wildfire\_dataset\_2n\_version/train' val\_dir = '/kaggle/input/the-wildfire-dataset/the\_wildfire\_dataset\_2n\_version/val' test\_dir = '/kaggle/input/the-wildfire-dataset/the\_wildfire\_dataset\_2n\_version/test'

In [5]:

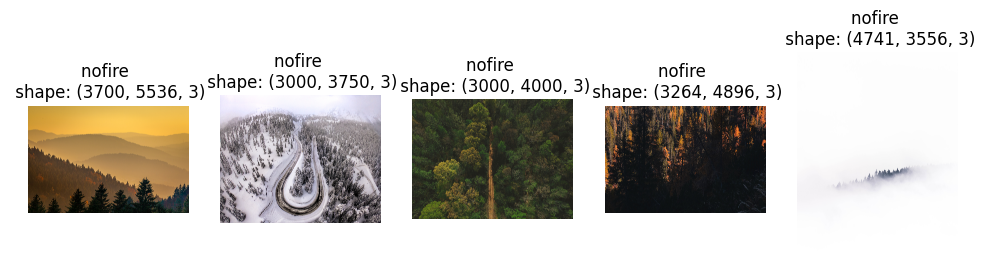
# List all the classes classes = os.listdir(train\_dir) num\_classes = len(classes) # Display the class names print(f'Number of Classes: {num\_classes}') print(f'Classes: {classes}')

Number of Classes: 2

Classes: ['nofire', 'fire']

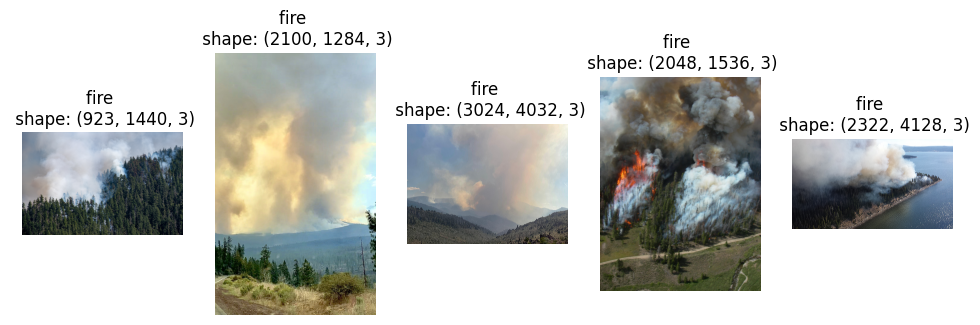
In [6]:

# Let's visualize the images in the dataset plt.figure(figsize=(12, 10)) for i in range(5): class\_path = os.path.join(train\_dir, classes[0]) img\_name = os.listdir(class\_path)[i] img\_path = os.path.join(class\_path, img\_name) img = plt.imread(img\_path) plt.subplot(1, 5, i+1) plt.imshow(img) plt.title(f'{classes[0]} \n shape: {img.shape}') plt.axis('off') plt.show()



In [7]:

plt.figure(figsize=(12, 10)) for i in range(5): class\_path = os.path.join(train\_dir, classes[1]) img\_name = os.listdir(class\_path)[i] img\_path = os.path.join(class\_path, img\_name) img = plt.imread(img\_path) plt.subplot(1, 5, i+1) plt.imshow(img) plt.title(f'{classes[1]} \n shape: {img.shape}') plt.axis('off') plt.show()



In [8]:

# Preprocessing # Image dimensions and batch size img\_width, img\_height = 150, 150 batch\_size = 32 # there will be 32 images in a batch! # Data generators train\_datagen = ImageDataGenerator(rescale=1./255) val\_datagen = ImageDataGenerator(rescale=1./255) test\_datagen = ImageDataGenerator(rescale=1./255) train\_generator = train\_datagen.flow\_from\_directory( train\_dir, target\_size = (img\_width, img\_height), batch\_size = batch\_size, class\_mode = 'binary', shuffle = True ) val\_generator = val\_datagen.flow\_from\_directory( val\_dir, target\_size = (img\_width, img\_height), batch\_size = batch\_size, class\_mode = 'binary', shuffle = True ) test\_generator = test\_datagen.flow\_from\_directory( test\_dir, target\_size = (img\_width, img\_height), batch\_size = batch\_size, class\_mode = 'binary', shuffle = True )

Found 1887 images belonging to 2 classes.

Found 402 images belonging to 2 classes.

Found 410 images belonging to 2 classes.

In [9]:

# map the indices class\_mapping = train\_generator.class\_indices # print(class\_mapping) # Extract the class names class\_names = list(class\_mapping.keys()) print("Class Names:", class\_names)

Class Names: ['fire', 'nofire']

In [10]:

# Let's build the CNN Model model = Sequential([ Input(shape=(img\_width, img\_height, 3)), Conv2D(32, (3,3), activation='relu'), MaxPooling2D(pool\_size=(2,2)), Conv2D(64, (3, 3), activation='relu'), MaxPooling2D(pool\_size=(2, 2)), Conv2D(128, (3, 3), activation='relu'), MaxPooling2D(pool\_size=(2, 2)), Flatten(), Dense(512, activation='relu'), Dropout(0.5), Dense(1, activation='sigmoid') ] )

In [12]:

model.compile(optimizer='adam', loss='binary\_crossentropy', metrics=['accuracy']) model.summary()

**Model: "sequential"**

┏━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━┓

┃ **Layer (type)** ┃ **Output Shape** ┃ **Param #** ┃

┡━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━┩

│ conv2d (Conv2D) │ (None, 148, 148, 32) │ 896 │

├─────────────────────────────────┼────────────────────────┼───────────────┤

│ max\_pooling2d (MaxPooling2D) │ (None, 74, 74, 32) │ 0 │

├─────────────────────────────────┼────────────────────────┼───────────────┤

│ conv2d\_1 (Conv2D) │ (None, 72, 72, 64) │ 18,496 │

├─────────────────────────────────┼────────────────────────┼───────────────┤

│ max\_pooling2d\_1 (MaxPooling2D) │ (None, 36, 36, 64) │ 0 │

├─────────────────────────────────┼────────────────────────┼───────────────┤

│ conv2d\_2 (Conv2D) │ (None, 34, 34, 128) │ 73,856 │

├─────────────────────────────────┼────────────────────────┼───────────────┤

│ max\_pooling2d\_2 (MaxPooling2D) │ (None, 17, 17, 128) │ 0 │

├─────────────────────────────────┼────────────────────────┼───────────────┤

│ flatten (Flatten) │ (None, 36992) │ 0 │

├─────────────────────────────────┼────────────────────────┼───────────────┤

│ dense (Dense) │ (None, 512) │ 18,940,416 │

├─────────────────────────────────┼────────────────────────┼───────────────┤

│ dropout (Dropout) │ (None, 512) │ 0 │

├─────────────────────────────────┼────────────────────────┼───────────────┤

│ dense\_1 (Dense) │ (None, 1) │ 513 │

└─────────────────────────────────┴────────────────────────┴───────────────┘

**Total params:** 19,034,177 (72.61 MB)

**Trainable params:** 19,034,177 (72.61 MB)

**Non-trainable params:** 0 (0.00 B)

In [13]:

# train the model history = model.fit( train\_generator, steps\_per\_epoch = train\_generator.samples // batch\_size, epochs = 12, validation\_data = val\_generator, validation\_steps = val\_generator.samples // batch\_size )

/usr/local/lib/python3.11/dist-packages/keras/src/trainers/data\_adapters/py\_dataset\_adapter.py:121: UserWarning: Your `PyDataset` class should call `super().\_\_init\_\_(\*\*kwargs)` in its constructor. `\*\*kwargs` can include `workers`, `use\_multiprocessing`, `max\_queue\_size`. Do not pass these arguments to `fit()`, as they will be ignored.

self.\_warn\_if\_super\_not\_called()

Epoch 1/12

32/58 ━━━━━━━━━━━━━━━━━━━━ 3:13 7s/step - accuracy: 0.4874 - loss: 0.9499

/usr/local/lib/python3.11/dist-packages/PIL/Image.py:3402: DecompressionBombWarning: Image size (104688771 pixels) exceeds limit of 89478485 pixels, could be decompression bomb DOS attack.

warnings.warn(

51/58 ━━━━━━━━━━━━━━━━━━━━ 53s 8s/step - accuracy: 0.5278 - loss: 0.8587

/usr/local/lib/python3.11/dist-packages/PIL/Image.py:3402: DecompressionBombWarning: Image size (89747104 pixels) exceeds limit of 89478485 pixels, could be decompression bomb DOS attack.

warnings.warn(

58/58 ━━━━━━━━━━━━━━━━━━━━ 568s 10s/step - accuracy: 0.5410 - loss: 0.8325 - val\_accuracy: 0.6901 - val\_loss: 0.5447

Epoch 2/12

1/58 ━━━━━━━━━━━━━━━━━━━━ 1s 35ms/step - accuracy: 0.7500 - loss: 0.4796

/usr/local/lib/python3.11/dist-packages/keras/src/trainers/epoch\_iterator.py:107: UserWarning: Your input ran out of data; interrupting training. Make sure that your dataset or generator can generate at least `steps\_per\_epoch \* epochs` batches. You may need to use the `.repeat()` function when building your dataset.

self.\_interrupted\_warning()

58/58 ━━━━━━━━━━━━━━━━━━━━ 91s 2s/step - accuracy: 0.7500 - loss: 0.4796 - val\_accuracy: 0.7083 - val\_loss: 0.5231

Epoch 3/12

58/58 ━━━━━━━━━━━━━━━━━━━━ 556s 9s/step - accuracy: 0.7448 - loss: 0.4994 - val\_accuracy: 0.7708 - val\_loss: 0.4705

Epoch 4/12

58/58 ━━━━━━━━━━━━━━━━━━━━ 84s 1s/step - accuracy: 0.7500 - loss: 0.5213 - val\_accuracy: 0.7656 - val\_loss: 0.4855

Epoch 5/12

58/58 ━━━━━━━━━━━━━━━━━━━━ 465s 8s/step - accuracy: 0.7620 - loss: 0.4837 - val\_accuracy: 0.7500 - val\_loss: 0.4686

Epoch 6/12

58/58 ━━━━━━━━━━━━━━━━━━━━ 131s 2s/step - accuracy: 0.7188 - loss: 0.6087 - val\_accuracy: 0.7474 - val\_loss: 0.5179

Epoch 7/12

58/58 ━━━━━━━━━━━━━━━━━━━━ 491s 8s/step - accuracy: 0.7696 - loss: 0.4902 - val\_accuracy: 0.7630 - val\_loss: 0.4570

Epoch 8/12

58/58 ━━━━━━━━━━━━━━━━━━━━ 82s 1s/step - accuracy: 0.8438 - loss: 0.3137 - val\_accuracy: 0.7708 - val\_loss: 0.4474

Epoch 9/12

58/58 ━━━━━━━━━━━━━━━━━━━━ 462s 8s/step - accuracy: 0.8060 - loss: 0.4305 - val\_accuracy: 0.7995 - val\_loss: 0.4452

Epoch 10/12

58/58 ━━━━━━━━━━━━━━━━━━━━ 79s 1s/step - accuracy: 0.7812 - loss: 0.4174 - val\_accuracy: 0.7734 - val\_loss: 0.4424

Epoch 11/12

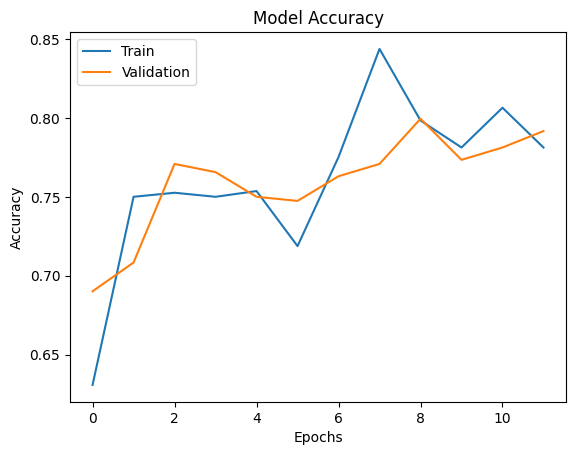
58/58 ━━━━━━━━━━━━━━━━━━━━ 487s 8s/step - accuracy: 0.8108 - loss: 0.4102 - val\_accuracy: 0.7812 - val\_loss: 0.4533

Epoch 12/12

58/58 ━━━━━━━━━━━━━━━━━━━━ 83s 1s/step - accuracy: 0.7812 - loss: 0.5917 - val\_accuracy: 0.7917 - val\_loss: 0.4406

In [14]:

# Evaluate the model - plotting training and validation acc plt.plot(history.history['accuracy']) plt.plot(history.history['val\_accuracy']) plt.title('Model Accuracy') plt.ylabel('Accuracy') plt.xlabel('Epochs') plt.legend(['Train', 'Validation'], loc='upper left') plt.show()



In [15]:

# Evaluate the model - plotting training and validation loss plt.plot(history.history['loss']) plt.plot(history.history['val\_loss']) plt.title('Model Loss') plt.ylabel('Loss') plt.xlabel('Epochs') plt.legend(['Train', 'Validation'], loc='upper left') plt.show()



In [16]:

# Evaluate the model using test data test\_loss, test\_acc = model.evaluate(test\_generator, steps=test\_generator.samples // batch\_size) print(f'Test Accuracy: {test\_acc:.4f}')

/usr/local/lib/python3.11/dist-packages/PIL/Image.py:3402: DecompressionBombWarning: Image size (96631920 pixels) exceeds limit of 89478485 pixels, could be decompression bomb DOS attack.

warnings.warn(

/usr/local/lib/python3.11/dist-packages/PIL/Image.py:3402: DecompressionBombWarning: Image size (101859328 pixels) exceeds limit of 89478485 pixels, could be decompression bomb DOS attack.

warnings.warn(

9/12 ━━━━━━━━━━━━━━━━━━━━ 23s 8s/step - accuracy: 0.8107 - loss: 0.3873

/usr/local/lib/python3.11/dist-packages/PIL/Image.py:3402: DecompressionBombWarning: Image size (94487082 pixels) exceeds limit of 89478485 pixels, could be decompression bomb DOS attack.

warnings.warn(

12/12 ━━━━━━━━━━━━━━━━━━━━ 97s 8s/step - accuracy: 0.8047 - loss: 0.3958

Test Accuracy: 0.7969

In [17]:

# Save the model model.save('FFD.keras')

In [20]:

from tensorflow.keras.models import load\_model from tensorflow.keras.preprocessing import image # Load the model model = load\_model('FFD.keras') # Predict on new image - create a function def predict\_fire(img\_path): img = image.load\_img(img\_path, target\_size=(150, 150)) img\_array = image.img\_to\_array(img) img\_array = np.expand\_dims(img\_array, axis=0) img\_array /= 255.0 prediction = model.predict(img\_array) if prediction[0] > 0.5: predicted\_class = class\_names[1] else: predicted\_class = class\_names[0] plt.imshow(img) plt.title(f'Predicted: {predicted\_class}') plt.axis('off') plt.show()

In [21]:

os.listdir('/kaggle/input/the-wildfire-dataset/the\_wildfire\_dataset\_2n\_version/test/nofire')

Out[21]:

['josh-sorenson-w\_QAjcf1ipU-unsplash.jpg',

'balazs-busznyak-AhMSy4GpU0w-unsplash.jpg',

'52358616216\_ec885ff145\_o.jpg',

'matthias-heil-T7lMR-syZAg-unsplash.jpg',

'mark-boss-KG8N1Rqf02w-unsplash.jpg',

'chris-barbalis-SIbpq4Lvzdk-unsplash.jpg',

'brendan-beale-h3qe4EZd4\_4-unsplash.jpg',

'dan-rogers-G807nWEuFIo-unsplash.jpg',

'hala-alghanim-yyE90v\_E8dA-unsplash.jpg',

'zetong-li-79sabz\_JrXg-unsplash.jpg',

'federico-bottos-JUFuI-kBtas-unsplash.jpg',

'single-earth-FP5tOXT6aMs-unsplash.jpg',

'sunnie-KJ\_G43MHDf0-unsplash.jpg',

'35581707273\_fed23248e8\_o.jpg',

'martin-sanchez-ML6kHR--Uys-unsplash.jpg',

'marek-piwnicki-DCGABYDIdT0-unsplash.jpg',

'austin-schmid-zQ-y4Gj8194-unsplash.jpg',

'rhema-kallianpur-t0GpJ2cdJuo-unsplash.jpg',

'victor-rodriguez-e5g9otcYzkU-unsplash.jpg',

'kristaps-ungurs-7aY6JOyFY8A-unsplash.jpg',

'52452040025\_916ddf3f9d\_o.jpg',

'daniil-silantev-VggwSJxYlAw-unsplash.jpg',

'josh-withers-UN\_W3z4D0pw-unsplash.jpg',

'dave-hoefler-swH\_IVJGLDA-unsplash.jpg',

'tobias-reich-SfhLrF4yxXI-unsplash.jpg',

'cristian-vieriu-lDC7L1cQCeY-unsplash.jpg',

'ozark-drones-iWMksWAU0so-unsplash.jpg',

'tom-allport-Yh8\_B8NlcFU-unsplash.jpg',

'kym-mackinnon-HCSWKjv\_cys-unsplash.jpg',

'carles-rabada-f7UprkNqi08-unsplash.jpg',

'nathan-anderson-PpZiXBY7\_Nw-unsplash.jpg',

'j-brouwer-43jy3ugj8pk-unsplash.jpg',

'matthew-brodeur-aIxDXa4T-u8-unsplash.jpg',

'nathan-anderson-ajFuxNTkRiE-unsplash.jpg',

'kym-mackinnon-DDyw4XRzojA-unsplash.jpg',

'marek-piwnicki-Z45VspLrGYk-unsplash.jpg',

'jakub-sejkora-utqJcneoFjo-unsplash.jpg',

'olivier-mesnage-lO6DOAIxDE0-unsplash.jpg',

'linford-miles-ECYv-bIJ3s4-unsplash.jpg',

'42376136734\_b4524237bf\_o.jpg',

'nathan-anderson-eelpgSLWrqo-unsplash.jpg',

'ozark-drones-XfnxPrd0-2g-unsplash.jpg',

'thought-catalog-bGiMXk8sMHw-unsplash.jpg',

'eberhard-grossgasteiger-NB3dTVw1T0o-unsplash.jpg',

'kym-mackinnon-zzlvHbtI1F8-unsplash.jpg',

'marek-piwnicki-H7oXHL19SlU-unsplash.jpg',

'adrian-mag-l9p-7JBmVFE-unsplash.jpg',

'dave-hoefler-UFnHt94r91w-unsplash.jpg',

'eberhard-grossgasteiger-Np1l2KmC0WA-unsplash.jpg',

'huper-by-joshua-earle-Hm68X4fFf6s-unsplash.jpg',

'clay-banks-EBW1p0vGqDc-unsplash.jpg',

'alan-hardman-pLq3oZDUEWk-unsplash.jpg',

'hans-isaacson-Ke1haOmqjdM-unsplash.jpg',

'arnold-dogelis-JN782tahBLI-unsplash.jpg',

'wolfgang-hasselmann-6yQbmbaucqg-unsplash.jpg',

'wil-stewart-WxXMapKQujw-unsplash.jpg',

'taras-chuiko-QiWzwOet4dQ-unsplash.jpg',

'kirill-tonkikh-gZIl698xiGc-unsplash.jpg',

'dave-hoefler-XfRglWxgJXk-unsplash.jpg',

'33696343816\_811bd98925\_o.jpg',

'christopher-burns-9wgdqT2VNSE-unsplash.jpg',

'dimitry-ljasuk-7q3xrpcBwuM-unsplash.jpg',

'julien-lanoy-EGqaqmSNc2I-unsplash.jpg',

'sander-lenaerts-X4yQIgcSYcQ-unsplash.jpg',

'caleb-russell-gYUK48mFopg-unsplash.jpg',

'eric-muhr-C6AY64q2l84-unsplash.jpg',

'zongnan-bao-deagK9-Dmt0-unsplash.jpg',

'kristaps-ungurs-hBWPG68Cct4-unsplash.jpg',

'tomasz-smal-v3N-gJf62nE-unsplash.jpg',

'lukas-mann-d-CvnGwwJts-unsplash.jpg',

'steven-van-elk-Vu23WPzGBr8-unsplash.jpg',

'kristaps-ungurs-s-4CYrJJXCU-unsplash.jpg',

'marek-piwnicki-mTmKpI53NcY-unsplash.jpg',

'jeffrey-blum-oFd4y82w9VY-unsplash.jpg',

'tobias-reich-zwoFKAgEvT0-unsplash.jpg',

'kristaps-ungurs-jywLBT6YHiM-unsplash.jpg',

'john-towner-CakC6u4d95g-unsplash.jpg',

'paul-pastourmatzis-KT3WlrL\_bsg-unsplash.jpg',

'36644001004\_f3e3a56edc\_o.jpg',

'peter-robbins-g1HdaJxDFrk-unsplash.jpg',

'marek-piwnicki-z14zNu59bLo-unsplash.jpg',

'toby-elliott-zeuK1e-c0wI-unsplash.jpg',

'anthony-intraversato-YG5l5XIZ76w-unsplash.jpg',

'51099217941\_143488e656\_o.jpg',

'cosmic-timetraveler-pIfF5Q8AYaE-unsplash.jpg',

'julien-lanoy-EjMMJPrfb5E-unsplash.jpg',

'corey-agopian-XGOzlCNeP1I-unsplash.jpg',

'arnold-dogelis-lsjRjuJlg5M-unsplash.jpg',

'kym-mackinnon-SMb0XdJxxfI-unsplash.jpg',

'neven-krcmarek-ZM7PgZ1zU2k-unsplash.jpg',

'tom-podmore-BdFn6XZzPOY-unsplash.jpg',

'41058079202\_08ce34a338\_o.jpg',

'kym-mackinnon-imEpwtWEPPQ-unsplash.jpg',

'malik-skydsgaard-GDDe4yAW5W4-unsplash.jpg',

'jan-szwagrzyk-TGOorJ1Fkfo-unsplash.jpg',

'pat-whelen-H5RYQLQMR38-unsplash.jpg',

'waranont-joe-CJ1oBPZJFPg-unsplash.jpg',

'39728540251\_888dc24afa\_o.jpg',

'jp-valery-W9iWetsS4EE-unsplash.jpg',

'eberhard-grossgasteiger-l\_oWTzCrgoA-unsplash.jpg',

'ales-krivec-7-Og8aZTRyM-unsplash.jpg',

'evan-leith-7fuZmBjjXjE-unsplash.jpg',

'trey-gibson-MGtUhF7oN70-unsplash.jpg',

'marek-piwnicki-2W7GMDQGuzI-unsplash.jpg',

'spencer-watson-G5QSfcEUQ-M-unsplash.jpg',

'eberhard-grossgasteiger-m9dWtVvoEww-unsplash.jpg',

'dominik-fischer-90HyQxrg7ZE-unsplash.jpg',

'kyle-mesdag-xrLn6YsmjI4-unsplash.jpg',

'casey-horner-TI2iMeyOo-o-unsplash.jpg',

'venti-views-Clwh1mksdAM-unsplash.jpg',

'clay-banks-yBAgX-oGSeo-unsplash.jpg',

'nathan-queloz-rj0KbZGsOP4-unsplash.jpg',

'mael-seigneurie-UAFp3u8Rcvk-unsplash.jpg',

'patrick-langwallner-zzRNMpEjppI-unsplash.jpg',

'leslie-cross-v7nyWOBSnLw-unsplash.jpg',

'nathan-anderson-YWWMGwXX2JY-unsplash.jpg',

'marek-piwnicki-C8pKTvqtug0-unsplash.jpg',

'nathan-queloz-gce72ixcGyU-unsplash.jpg',

'nathan-anderson-EZ9kyweogsI-unsplash.jpg',

'saketh-garuda-9nkoCtI1HnI-unsplash.jpg',

'jonathan-arbely-5D9k7RnOgLQ-unsplash.jpg',

'rc-victorino--o3x\_dhZijU-unsplash.jpg',

'52359041065\_1566fd6ec1\_o.jpg',

'cole-freeman-x2Gxd6ohasU-unsplash.jpg',

'tolga-ahmetler-\_b\_80WiUkzY-unsplash.jpg',

'kym-mackinnon-mT8BWsYyeU0-unsplash.jpg',

'gwyn-DlTcL3W6fOs-unsplash.jpg',

'casey-horner-Tmev8bpBSvw-unsplash.jpg',

'dawson-mccormick-dZrXqr0IA8A-unsplash.jpg',

'olena-sergienko-ZYZeA\_rwW9I-unsplash.jpg',

'kym-mackinnon-bYMkg-r-E0A-unsplash.jpg',

'50380847112\_db217aea45\_o.jpg',

'yevhenii-dubrovskyi-7Ya\_HBl5dpk-unsplash.jpg',

'ethan-dow-l7Wb6FXHIOQ-unsplash.jpg',

'eberhard-grossgasteiger--n-2Xhr3dOA-unsplash.jpg',

'max-voxberg-hY5OUxg\_Zgc-unsplash.jpg',

'single-earth-pfXTECJcppk-unsplash.jpg',

'52358931719\_b6fbcc77f2\_o.jpg',

'ben-den-engelsen-htcQ7uAWzAo-unsplash.jpg',

'sergio-melo-YYXh3QHhk40-unsplash.jpg',

'49200563427\_e5295b4455\_o.jpg',

't-selin-erkan--2xRZIGu3EU-unsplash.jpg',

'iewek-gnos-W6y6xCk2s6c-unsplash.jpg',

'john-vowles-lYbT6HBkimk-unsplash.jpg',

'annie-spratt-FdlsmCxwHGU-unsplash.jpg',

'bernd-dittrich-l8UiD8xY3IA-unsplash.jpg',

'kym-mackinnon-ZaannBOrvY8-unsplash.jpg',

'john-o-nolan-uMWPrcRsrto-unsplash.jpg',

'52357656897\_dde5e5a66a\_o.jpg',

'53078944067\_43c7e3552a\_o.jpg',

'mark-basarab-okYSDKshIes-unsplash.jpg',

'marek-piwnicki-bxsFTaNFFRY-unsplash.jpg',

'malik-skydsgaard-OSQyEh1siCo-unsplash.jpg',

'tristan-mckenzie-JZ7D9NsEO5I-unsplash.jpg',

'eberhard-grossgasteiger-LmqySFs3TQQ-unsplash.jpg',

'florian-van-duyn-grCP1PYmw60-unsplash.jpg',

'cathal-mac-an-bheatha-6OrHxKx\_tMc-unsplash.jpg',

'52452051055\_4e2a4e4de0\_o.jpg',

'olena-sergienko-3BlVILvh9hM-unsplash.jpg',

'eberhard-grossgasteiger-QoXYAE2hFJQ-unsplash.jpg',

'gwyn-jqdZhPMqsg4-unsplash.jpg',

'daniil-silantev-JfAzBxcRbv4-unsplash.jpg',

'nathan-anderson-n5RPHzjxUWM-unsplash.jpg',

'scott-taylor-rOrGGkyllm4-unsplash.jpg',

'nathan-anderson-MEIBNPI1RC8-unsplash.jpg',

'tom-podmore-eNWJIJeLnbk-unsplash.jpg',

'nitish-meena-ANo5\_iE9dcU-unsplash.jpg',

'raphael-biscaldi-Ux\_6dqF8WiQ-unsplash.jpg',

'zetong-li-UVv2vL3ih7o-unsplash.jpg',

'john-gibbons-fB\_aqZDhKFw-unsplash.jpg',

'john-towner-89PFnHKg8HE-unsplash.jpg',

'52359026765\_4a4ba189c8\_o.jpg',

'felix-besombes-fAseBsd1r0U-unsplash.jpg',

'dave-hoefler-U-Dhokv9iXk-unsplash.jpg',

'levi-bare-5\_\_JKMn5H\_c-unsplash.jpg',

'yann-allegre-ZDwp-QdzoH8-unsplash.jpg',

'zhu-yunxiao-m6ao043bsrI-unsplash.jpg',

'will-suddreth-hneAIGb01xQ-unsplash.jpg',

'jean-wimmerlin-Dmsiauit2z0-unsplash.jpg',

'nathan-queloz-1e\_HHIbqk7s-unsplash.jpg',

'ales-krivec-sTEg\_SLAyUs-unsplash.jpg',

'hunter-bryant-C0bPTsZnoH8-unsplash.jpg',

'50010867961\_903b6432f7\_o.jpg',

'23289748533\_1a14a0f249\_o.jpg',

'dave-hoefler-H2fNMhs7lVo-unsplash.jpg',

'stories-tNhGCix\_HvA-unsplash.jpg',

'yulian-alexeyev-R2BuLIE-i3M-unsplash.jpg',

'24498907094\_491ca433dd\_o.jpg',

'kristaps-ungurs-gnrKdXxeLrQ-unsplash.jpg',

'eberhard-grossgasteiger-HU\_Ahk35FK8-unsplash.jpg',

'marek-piwnicki-Yg7PWiIFqWM-unsplash.jpg',

'kyson-dana-uTRC5HzYyXE-unsplash.jpg',

'49422069676\_3e89abc052\_o.jpg',

'alina-fedorchenko-p2B1b56FjvQ-unsplash.jpg',

'daniel-mirlea-sS7GkbDdpMY-unsplash.jpg',

'augustine-wong-pdjFlhFqznc-unsplash.jpg',

'sander-weeteling-rLvCmdZrS4Y-unsplash.jpg',

'calvin-craig-MLf4-0\_k-P8-unsplash.jpg',

'andrew-svk-X4b2DK80F1k-unsplash.jpg',

'andrew-svk-ob1NS03tWSE-unsplash.jpg',

'mariusz-slonski-MxkcVyolp-4-unsplash.jpg',

'kym-mackinnon-4wlQBd2xXDI-unsplash.jpg',

'mathilda-khoo-00qUCwYToAY-unsplash.jpg',

'spencer-watson-bYLHitxMHmU-unsplash.jpg',

'drew-dau-w8uYD-4zbEo-unsplash.jpg',

'neven-krcmarek-50HS3AzJsvQ-unsplash.jpg',

'enrico-naletto-W0yZbLyshzc-unsplash.jpg',

'nigel-tadyanehondo-RonhCA39VRw-unsplash.jpg',

'troy-williams-7M6ZmfnKI6g-unsplash.jpg',

'eberhard-grossgasteiger-wzuySoHrtX0-unsplash.jpg',

'pat-whelen-5PGbTcaX9rs-unsplash.jpg',

'kym-mackinnon-y70s9rNiuvY-unsplash.jpg',

'malik-skydsgaard-GxEOyQLP7xY-unsplash.jpg',

'filip-varga-AsgsGO6U78g-unsplash.jpg',

'harrison-steen-lUDMZUWFUXE-unsplash.jpg',

'inggrid-koe-kbKEuU-YEIw-unsplash.jpg',

'nikola-knezevic-lPzO6ZGRGYI-unsplash.jpg',

'nathan-queloz-gkMuexmFMR4-unsplash.jpg',

'bahadir-L2G0g3mSliU-unsplash.jpg',

'arnold-dogelis-04JZFPD5eBA-unsplash.jpg',

'eberhard-grossgasteiger-gptRZytFzFQ-unsplash.jpg',

'levi-bare-tI1Kc3LuRMI-unsplash.jpg',

'lasma-artmane-II0WWqBorw0-unsplash.jpg',

'ivars-utinans-r\_LT\_Ne3pVo-unsplash.jpg',

'ryan-searle-bMQEHzEm09w-unsplash.jpg',

'45636031272\_8408d32834\_o.jpg',

'carles-rabada-c9RYAY5f4Ac-unsplash.jpg',

'silas-baisch-IRk4gKJQLBI-unsplash.jpg',

'anton-repponen-zNxlvCZhHNA-unsplash.jpg',

'43078090081\_07d1f9ec0f\_o.jpg',

'jeremy-bishop-1GJnLJjQ9sg-unsplash.jpg',

'guy-tsror-ibpr-RbpRG8-unsplash.jpg',

'smaran-alva-yOj9j2Ho-kE-unsplash.jpg',

'ales-krivec-VfMgiZscSXE-unsplash.jpg',

'mj-lDXV1dVQrTA-unsplash.jpg',

'andrei-p-Sc\_jhFDkqQc-unsplash.jpg',

'marek-piwnicki-U22rbeq3m5E-unsplash.jpg',

'ryo-yoshitake-c\_rWbq-Bcb4-unsplash.jpg',

'nicolas-cool-kG2uTD1KxYI-unsplash.jpg',

'komang-gita-krishna-murti-bVeodNsZQZI-unsplash.jpg',

'kym-mackinnon-GOAbTarWc0Q-unsplash.jpg',

'alina-fedorchenko-UyQU4SlZTaA-unsplash.jpg',

'erik-mclean-0aYkdo3XfYo-unsplash.jpg',

'casey-horner--ulHK\_2i1aY-unsplash.jpg',

'nick-scheerbart-xFjAftU8lMY-unsplash.jpg',

'artem-sapegin-8c6eS43iq1o-unsplash.jpg',

'darpan-WwegI5jjJ4c-unsplash.jpg',

'nathan-anderson-d1gPSEN9dOE-unsplash.jpg',

'christine-kozak-HuwH6i0\_pwk-unsplash.jpg',

'chris-linnett-FatkDF8j3pk-unsplash.jpg',

'35554264024\_b33afa21ac\_o.jpg']

In [22]:

predict\_fire('/kaggle/input/the-wildfire-dataset/the\_wildfire\_dataset\_2n\_version/test/nofire/35554264024\_b33afa21ac\_o.jpg')

1/1 ━━━━━━━━━━━━━━━━━━━━ 2s 2s/step

