*F*lamePerceptionAPP: Enabling Fire Engineers and Researchers to Understand and Analyze Flame Data

Meijie Wang*
Worcester Polytechnic Institute

ABSTRACT

Existing research on flame detection mainly focuses on the improvement of algorithms, from the traditional image processing method to the combination of image processing method and machine learning method. However, there's a huge gap between those algorithm improvements and the users' needs from those who care about fire information.

In this project, we will bridge the gap and explore applying most recent techniques which combine computer vision and deep learning techniques to help fire researchers from Fire Protection Department in WPI to access flame data without previous knowledge in computer science. We will build tools to help them percept, visualize and analyze flame data and better conduct their research.

Index Terms: Flame Detection,Computer Vision,Deep Learning,Web Application,Information Visualization

1 INTRODUCTION

Fire is one of the common happening natural hazards and detection at an early stage of fire is extremely important to minimize loss of lives and property damage. According to the Annual Disaster Statistical Review, Across the world, in the year 2015, wildfire disasters caused 494,000 victims with damage worth USD 3.1 billion, and the devastating wildfires continued from 2017 into 2018. In 2018, the Attica Fires in Greece becomes the deadliest wildfire in Europe in this century and has killed nearly 126 people. In the United States, the California Camp Fire killed 88 people and caused damage worth nearly 16.5 billion USD. Fire engineering is an application field of science and engineering principles to protect people, property, and the environment from the harmful fire hazards. Fire protection district is widely distributed across the world and provides firefighting, fire protection, fire investigation, and technical rescue services. In addition to fire protection departments in the industry, some universities also have fire engineering majors and laboratories. Worcester Polytechnic Institute's fire engineering department is one of them who promote significant fire research, develop practical solutions through fire laboratory.

This project is a cooperative work with the Combustion Lab of Fire Protection Department in Worcester Polytechnic Institute. They conduct research which will enable fire safety professionals to predict fire hazards and they work on developing novel techniques to measure fire-induced flows and help understand complex fire problem. Computer vision-based fire detection is one of the tasks. In recent years, the convolutional neural network (CNN) of deep learning has become a widely adopted technique because of its high accuracy recognition rate in industry applications. There are some researchers from computer science field exploring topics like Flame Detection Using Deep Learning [8] and A convolutional neural network-based flame detection method in video sequence [10]. They are trying to explore flame detection with reliable and effective techniques to help solve problems in practice.

*e-mail: mwang3@wpi.edu

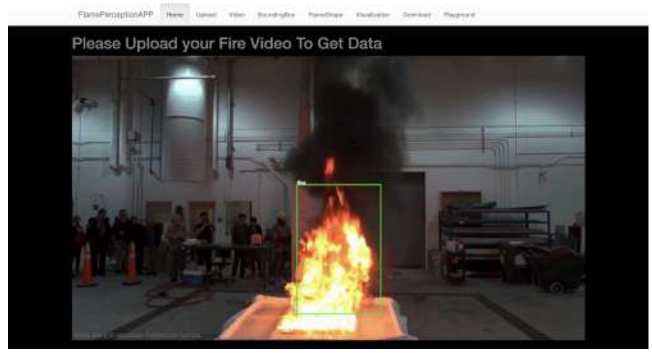

Figure 1: Homepage Interface of the application.

However, the academic exploration of fire detection is more focused on the improvement of algorithm and performance. Those most advanced techniques have rarely been reached by researchers and engineers in Fire Protection filed to put into practice. This project aims to bridge the gap between those techniques and real user needs of fire protection experts and help people access flame data without previous knowledge in computer science or programming. As we are based in Worcester Polytechnic Institute, We first conducted behavior observation and user study with fire researchers in the Combustion Lab in the school. we defined the functions of our application based on their research behavior. Then we dig into current techniques to help satisfy user needs from fire experts. We conducted experiments in both traditional computer vision methods and a combination of computer vision and deep learning models. Specifically, we implemented image segmentation and image enhancement methods to extract flame shape. We also experimented with Convolutional Neural Network to detect flame motion. For the software pipeline, We first agreed with experts in Combustion Lab on building a web application to implement the flame detection algorithms for use. To better assist the fire experts to analyze flame data, we utilized flask as a backend framework connected with D3.js library to help visualize the flame data on the front end. The user could upload the fire video on a phone or a laptop via the webpage link, the backend will process the video with the flame detection algorithms, and the extracted flame data will be given back to the front end and turned into visualization.

2 RELATED WORK

We first dug into previous works of exploration in fire detection algorithms. Visual features like color-based model, shape-based model, motion-based model, fuzzy-based model could be used for flame detection. Among them, Color-based model alone conduct a good performance [1].

Shallow learning and deep learning are both methods to detect flame from images. Deep learning has been used in object detection, recognition and tracking [2] and turned out to be an efficient way to resolve visual problems [9]. Compared with shallow learning, the performance of deep learning is not affected by using flame colors or not when training with a large dataset, but gas fire detection accuracy

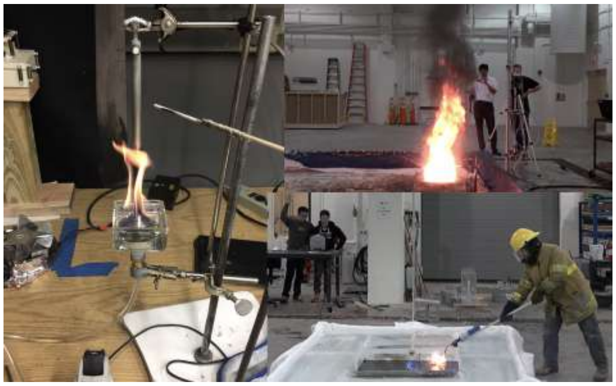

Figure 2: How fire researchers in the Combustion Lab in WPI conducted in lab studies of fire.

will be affected with a color-based model alone [7] .

A color fire feature detection and recognition scheme based on convolutional neural network method was proposed in 2018 [10]. In this method, an RGB-based model is first used to extract the color feature of fire image and obtain the candidate area. The neural network classifies the normalized feature maps after that. Finally, the alarm signal is obtained by the classification result of CNN and test the performance of the proposed neural network. This algorithm proved to have achieved superior performance in improving the fire color feature recognition based on RGB model accuracy.

3 User Study

In order to understand the research methods and processes of fire engineers, and to mine user needs, we reached a cooperation with the Combustion lab of Fire Protection Department in Worcester Polytechnic Institute. In the summer of 2019, we participated in the weekly meeting of the combustion lab and listened to the presentations made by the lab researchers. There were a total of 10 researchers in the combustion lab in the summer. Their research topics were quite different, some were to study the airflow with flame, some focused on studying the bubbles, some were to study the factors affecting the flame propagation rate, and some focused on the relationship between the flame and the air. The types of flames they studied were also very different, from explosive flames as large as a dozen meters high to small candle flames. But these studies will help them better understand flames and apply flame control research results to fire protection practice. Although their research directions and types of flames are diverse, a common method they all use is to control and simulate different types of flame environments in-lab study and record flame videos with cameras. The Fire researchers will observe their recorded flame videos, compare the effect, and draw experiments conclusions based on video data analysis.

We found that the method fire researchers utilized to analyze flame video is very primitive, mainly by observing and comparing video images and using Matlab for image processing to assist in the analysis of experimental results. Through interviews and communications, we found that fire researchers do not understand the principles and functions of these Matlab codes, nor do they know how to process and analyze the data after image processing. The interview turned out that the advanced flame detection algorithms proposed in the field of computer vision have never been reached by fire engineers and researchers who really care about flame detection and flame data acquisition. In order to better help fire researchers to obtain flame data in research experiments without previous knowledge in programming, we went through and summarized their fire research presentations. We found that they paid particular attention to the height of the flame, the shape of the flame, the area of the

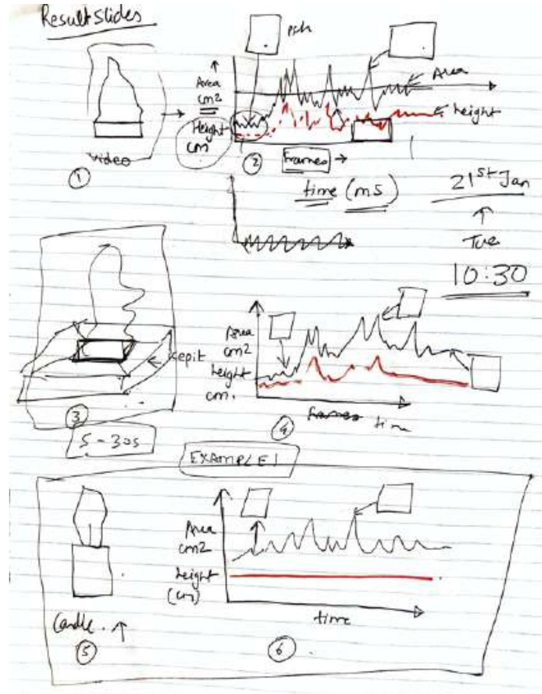

Figure 3: Discussion draft about visualization of flame data.

flame, and the frequency of the flame. These factors are important features that they always paid attention to in fire experiments in different research topics and experiment environments. As fire researchers are also not familiar with data representation forms, to better help flame researchers understand flame data, we also discussed with them about the visual form of flame data. We tested with line chart, bar chart and pie chart draft to visualize the flame information, and line chart turned out to be a more convincing visual chart for fire researchers. The line charts visualization system will be addressed in the design and development of the application to help fire researchers understand flame data more intuitively.

4 Experiments

To better build the application, we did some experiments on different techniques and compared the output. We tested with the computer vision method and utilized both image segmentation algorithms and image enhancement algorithms. We also utilized the Deepgaze library which is based on OpenCV and Tensorflow to detect the motion of flame. And at last, We focused on a deep learning model for object detection called MaskRCNN.

4.1 Computer Vision

Our first implementation is to use computer vision library. OpenCV is the leading open-source library for computer vision, image processing, and machine learning. OpenCV is being used for a very wide range of applications which include: Street view image stitching, Automated inspection and surveillance, Robot and driver-less car navigation and control, Medical image analysis, Video/image search and retrieval, Movies - 3D structure from motion, Interactive art installations. Here we tried both Image Segmentation technique and Image Enhancement technique and did a comparison of the results.

4.1.1 Image Segmentation

First, we used Image Segmentation with Distance Transform and Watershed Algorithm in OpenCV to segment flame shape in frames.

Figure 4: Effects of Image Segmentation with Distance Transform and Watershed Algorithm (mid) and Image Enhancement (right).

In this algorithm, we Use the OpenCV function cv::filter2D to perform some laplacian filtering for image sharpening, the OpenCV function cv::distanceTransform to obtain the derived representation of a binary image, where the value of each pixel is replaced by its distance to the nearest background pixel, and the OpenCV function cv::watershed to isolate objects in the image from the background. (see Fig. 4) As we can see from the comparison figures, the flame shape is nearly accurately segmented from background objects. However, due to the reflected light, it's hard to segment the flame shape from the desk with this segmentation algorithm.

4.1.2 Image Enhancement

Then we tried basic Image Enhancement using Basic Operator for brightness, contrast, and grayscale. The objective of image enhancement is to restore the image that is distorted during the transmission from one form to another form. The whole techniques centered upon the detailed formations and characteristics norms of image and according to the desire of applications and tools, the algorithm can be changed at any stage of the process. Adaption of all the algorithms in the image enhancement is easy. But with all the key features of the flame image, we need to care that flame features with lightness, brightness, and the color is hard to be simultaneously incorporated into the algorithm.

4.2 Motion Detection

Deepgaze [5] is a library for human-computer interaction, people detection and tracking which uses Convolutional Neural Networks (CNNs). Deepgaze contains useful packages for Head pose estimation (Perspective-n-Point, Convolutional Neural Networks), Face detection (Haar Cascade), Skin and color detection (Range detection, Backprojection), Histogram-based classification (Histogram Intersection), Motion detection (Frame differencing, MOG, MOG2), Motion tracking (Particle filter), Saliency map (FASA).

Deepgaze is based on OpenCV and Tensorflow, some of the best libraries in computer vision and machine learning. We tried deepgaze library and implemented Color detection using the Histogram Backprojection algorithm and compared it with our previous computer vision application. Because the flame feature is combined with lightness, brightness, and the color of flame, Color detection implementation here is far less effective than our previous computer vision method. Then we tested Motion detection and tracking using frame differencing on a video streaming. This algorithm is effective in tracking flame motion. However, this algorithm could only detect flame height and the frequency of the flame's motion, the important flame area data could not be calculated with this algorithm and flame height information is not accurate because of possible scene object motion disturbance (see Fig. 5). Thus we decided to establish our flame dataset and train deep learning model for flame detection.

4.3 Mask RCNN

In deep learning areas, YOLO [6] and RCNN are generally used models for object detection. In this project, We used a model called Mask R-CNN which could efficiently detect objects in an image while simultaneously generating a high-quality segmentation mask

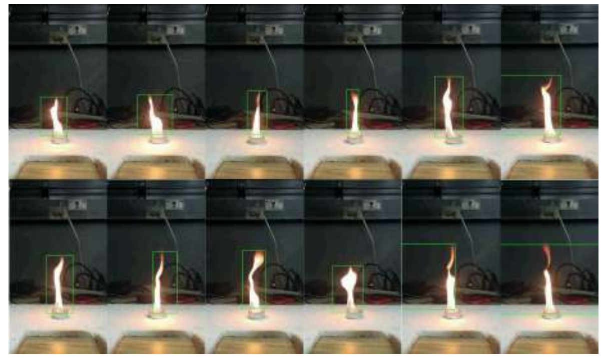

Figure 5: Flame motion detection with Deepgaze. My github repository: https://github.com/wmjpillow/FlameMotionDetection.

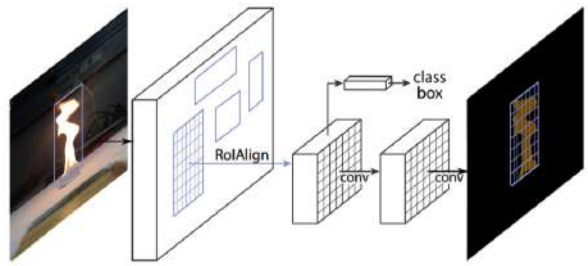

Figure 6: Mask RCNN Framework For Instance Segmentation.

for each instance. The method extends Faster R-CNN by adding a branch for predicting an object mask in parallel with the existing branch for bounding box recognition. Mask R-CNN is simple to train and adds only a small overhead to Faster R-CNN, running at 5 fps. Moreover, Mask R-CNN is easy to generalize to other tasks, e.g., allowing us to estimate human poses in the same framework. Mask R-CNN shows top results in all three tracks of the COCO suite of challenges, including instance segmentation, bounding-box object detection, and person keypoint detection. Without bells and whistles, Mask R-CNN outperforms all existing, single-model entries on every task, including the COCO 2016 challenge winners [3]. Thus in this project, the MaskRCNN model is considered as the best deep Neural Network for us to train a flame detection model(see Fig. 6). With Mask RCNN, We could output flame area with mask, flame height with bounding box.

4.3.1 Flame Dataset

To build the dataset for training the model, we extracted 1000 images(per image every 5 frames) from a previously recorded in lab study fire video. We labeled the images with PixelAnnotationTool (https://github.com/abreheret/PixelAnnotationTool). We devided the labelled images into two parts: 900 images as training dataset and 100 images as test dataset.

4.3.2 Model Modules

Mask R-CNN (see Fig. 6) consists of the following modules:

The backbone is a standard convolutional neural network that serves as a feature extractor. The early layers detect low level features (edges and corners), and later layers successively detect higher level features (car, person, sky). Passing through the backbone network, the image is converted from 1024x1024px x 3 (RGB) to a feature map of shape 32x32x2048. This feature map becomes the input for the following stages.

Figure 7: Flame image before training (left), labeled flame data (mid), generated flame mask after first round of training (right).

The RPN is a lightweight neural network that scans the image in a sliding-window fashion and finds areas that contain objects. The regions that the RPN scans over are called anchors. Which are boxes distributed over the image area, as show on the left. This is a simplified view, though. In practice, there are about 200K anchors of different sizes and aspect ratios, and they overlap to cover as much of the image as possible. The RPN generates two outputs for each anchor: Anchor Class and Bounding Box Refinement.

ROI Classifier Bounding Box Regressor: This stage runs on the regions of interest (ROIs) proposed by the RPN. And just like the RPN, it generates two outputs for each ROI: 1 Class: The class of the object in the ROI. This network has the capacity to classify regions to specific classes (person, car, chair, . . . etc.). It can also generate a background class, which causes the ROI to be discarded. 2 Bounding Box Refinement: Very similar to how it's done in the RPN, and its purpose is to further refine the location and size of the bounding box to encapsulate the object.

Segmentation Masks: The mask branch is a convolutional network that takes the positive regions selected by the ROI classifier and generates masks for them. The generated masks are low resolution: 28x28 pixels. But they are soft masks, represented by float numbers, so they hold more details than binary masks. The small mask size helps keep the mask branch light. During training, we scale down the ground-truth masks to 28x28 to compute the loss, and during inferencing we scale up the predicted masks to the size of the ROI bounding box and that gives us the final masks, one per object.

4.3.3 Model Training

Mask R-CNN is a fairly large model. Especially that our implementation uses ResNet101 and FPN. Before writing the proposal, we just trained the model on the CPU, which might be a big hurt for training. It would be better if we could have a modern GPU with at least 12GB of memory. Our first training ran 2000 steps at a rate of 2.396 sec/step, and the lost arrived at 0.0004. Then we tried more steps which is nearly 20000 steps, at a rate of 4.049 sec/step, and the loss arrived 0.0000. But our testing dataset proved that this training caused model overfitting (see Fig. 7).

After that, we put forward many methods to improve the model, like enlarging the dataset, pre-processing the images before training and so on. Now we are working on semi-supervised learning to solve the problem. Semi-supervised learning is an approach to machine learning that combines a small amount of labeled data with a large amount of unlabeled data during training [11]. Semi-supervised learning falls between unsupervised learning (with no labeled training data) and supervised learning (with only labeled training data). Unlabeled data, when used in conjunction with a small amount of labeled data, can produce a considerable improvement in learning accuracy. The acquisition of labeled data for a learning problem often requires a skilled human agent (e.g. to transcribe an audio segment) or a physical experiment (e.g. determining the 3D structure of a protein or determining whether there is oil at a particular location). The cost associated with the labeling process thus may render large, fully labeled training sets infeasible, whereas the acquisition of unlabeled data is relatively inexpensive. In such situations, semi-supervised learning can be of great practical value.

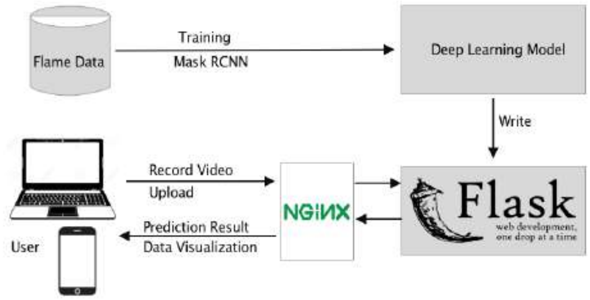

Figure 8: Diagram of the software system model

5 Infrastructure

Since the purpose of the project is to design and develop software for engineers and researchers in Fire Protection Department in WPI to help them easily conduct their research in flame, the design of the application flow and the infrastructure is based on our previous user study. First, we need to come to an agreement with fire researchers about the form of the application, a mobile APP or a web application. With a mobile app, the fire researchers could easily record the in lab fire experiments on the phone and get the flame data output. However, considering that researchers in Fire Protection Department use different types of phones with varied qualities of camera and recording flame video by hand may well bring video quality problems for accurate data processing, we finally decide to build a web application. With web application, researchers could upload the video they recorded whether by phone camera or by professional camera. Since the application is designed to be responsive on both computer and mobile, on the one hand, they could simply open the website on their phone and upload the video to get immediate in lab study result, on the other hand, they could access the website via computer and upload fire videos recorded with Canon camera(which is their current lab study recording tool).

5.1 System Model

Since the function of the application is very clear based on our previous user study, the proposed system would be pretty simple (see Fig. 8). The user records the fire video and uploads it whether on phone or on laptop via the website, the backend with the trained model will process the flame video data, output prediction result and give it back to the front end, the front end will give the data file for downloading and also visualize the data for analyzing.

5.2 Web Application

We chose Flask [4] as a backend framework to build the application. Flask is a micro web framework written in Python. It is classified as a microframework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common framework related tools. Flask is very flexible and more lightweight than Django, great for smaller projects and would be a perfect choice for building this type of light application.

The front end of the web application would mainly be consisted of two parts. One part is the uploading page where the user uploads the fire video. The other part is the data output page, where the user could download the processed flame data of the video and interact with the data visualization.

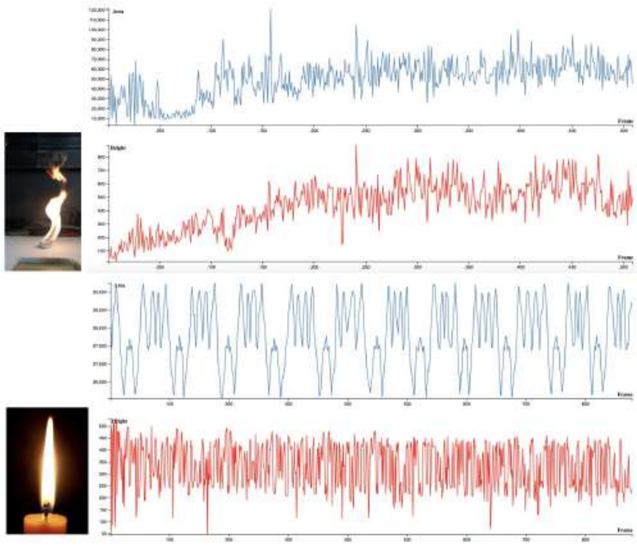

Figure 9: Line chart data visualization of different flame patterns

5.3 Visualization

Since this is a web application, d3.js is the best library to build data visualization to help fire researchers analyze it. D3.js (also known as D3, short for Data-Driven Documents) is a JavaScript library for producing dynamic, interactive data visualizations in web browsers. It makes use of Scalable Vector Graphics (SVG), HTML5, and Cascading Style Sheets (CSS) standards. Line chart with time sequence as x-axis and flame height/ flame area value as y-axis is the best chart type to visualize the data (see Fig. 9). We will also add other types of chart like bar chart or pie chart to help visualize the flame data from a more comprehensive perspective.

6 Future Work

Now we have built a prototype of the web application, the elementary function is implemented. We tested the system with 6 fire videos. one is a candle flame video, two of them are in lab study of big fire and the other three are the small fire with 4cm, 6cm, 10cm containers.

our current algorithm calculates flame shape as pixels in the image instead of the centimeter, it will be hard for fire researchers to compare the flame data in different videos. To solve this problem, we will need to add input of reference substance size to help scale flame shape data. During the testings, we also found that Some of the flames in recording videos are out of camera range and its height/area information in the recorded videos is not correct. As a result, we need to write a program to filter the raw data in the data processing part. And we also need to add interactive data visualization on the website front end to make it more user-friendly to fire researchers.

7 Timeline

Time/Status	Task	Description
Completed	Background	Related Works Collection
Completed	User Study	Define Application Function
Completed	Experiments	Algorithm Exploration
Completed	Establish Dataset	Label Flame Data
In Progress	Model Training	Get the best model
Completed	Infrastructure	Web App Backend
In Progress	Infrastructure	Visualization Pipeline
May 2020	Deployment	Host App on AWS
May 2020	Testing	Final App Testing
June 2020	Writing	Thesis writing
July 2020	Submission	Thesis presentation

References

[1] T. Celik, H. Demirel, H. Ozkaramanli, and M. Uyguroglu. Fire detection in video sequences using statistical color model. In *2006 IEEE International Conference on Acoustics Speech and Signal Processing Proceedings*, vol. 2, pp. II–II, 2006.

[2] T.-H. Chan, K. Jia, S. Gao, J. Lu, Z. Zeng, and Y. Ma. Pcanet: A simple deep learning baseline for image classification? *IEEE Transactions on Image Processing*, 24, 04 2014. doi: 10.1109/TIP.2015.2475625

[3] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. 03 2017.

[4] Pallets organization. Flask's documentation, 2010. [Online; accessed 16-April-2010].

[5] M. Patacchiola and A. Cangelosi. Head pose estimation in the wild using convolutional neural networks and adaptive gradient methods. *Pattern Recognition*, 71, 06 2017. doi: 10.1016/j.patcog.2017.06.009

[6] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. pp. 779–788, 06 2016. doi: 10.1109/CVPR.2016.91

[7] D. Shen, X. Chen, M. Nguyen, and W. Q. Yan. Flame detection using deep learning. In *2018 4th International Conference on Control, Automation and Robotics (ICCAR)*, pp. 416–420, 2018.

[8] D. Shen, X. Chen, and W. Yan. Flame detection using deep learning. pp. 416–420, 04 2018. doi: 10.1109/ICCAR.2018.8384711

[9] J. Wan, D. Wang, S. Hoi, P. Wu, J. Zhang, and J. Li. Deep learning for content-based image retrieval: A comprehensive study. 01 2014.

[10] Z. Zhong, M. Wang, Y. Shi, and W. Gao. A convolutional neural network-based flame detection method in video sequence. *Signal, Image and Video Processing*, 12, 06 2018. doi: 10.1007/s11760-018-1319-4

[11] X. Zhu. *Semi-supervised Learning*, pp. 1142–1147. 01 2017. doi: 10.1007/978-1-4899-7687-1_749