Engineering Applications of Artificial Intelligence 104 (2021) 104353

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai

Pixel-level automatic annotation for forest fire image

Xubing Yang [a,*], Run Chen [a], Fuquan Zhang [a], Li Zhang [a,b,**], Xijian Fan [a], Qiaolin Ye [a], Liyong Fu [c]

[a] College of Information Science and Technology, Nanjing Forestry University, Nanjing, 210037, PR China
[b] MIIT Key Laboratory of Pattern Analysis and Machine Intelligence, Nanjing University of Aeronautics and Astronautics, Nanjing, 210016, China
[c] Institute of Forest Resource Information Techniques, Chinese Academy of Forestry, Beijing 100091, China

ARTICLE INFO

Keywords:
Fire detection
Convex hull
Pixel-level
Image annotation

ABSTRACT

We propose an automatic annotation method for forest fire images in the level of pixel, where supervise information is introduced by interactive convex hulls. Instead of usual rectangle-/*regular*-shaped regions, we propose a convex hull algorithm for visually selecting polygonal (*irregular*) fire and no-fire regions. Guided by the goals of forest fire monitoring systems: high fire detection rate (true-positive) and then low false alarm rate (false-positive), we construct a k-nearest neighbor (kNN) based KD-tree to speed annotation. Compared to state-of-the-art, the proposed method not only widens the view of fire detection from conventional two-class to multi-class classification problem to meet complex forest image background, but also relaxes the limit of *i.i.d* (independent and identical distribution) hypothesis on machine learning methods. Furthermore, it is simple to use, which just relies on pixel information and avoids considering additional auxiliary features from multiple color spaces. Experimental evaluations are carrying on forest fire images, MIVIA dead-directional videos, and more challenging omni-directional videos. The comparison demonstrates that the proposed pixel-level annotation method is able to achieve higher fire detection rate and lower false alarm rate at the same time.

1. Introduction

Image annotation is one of the fundamental tasks of computer vision with many applications in image retrieval, interest object detection, and visual recognition. It is a process of labeling an image with keywords (tags), or describing the contents of the image which helps in the intelligent retrieval of relevant images through query representation. In general, this labeling can be done manually or automatically. So-called automatic image annotation (AIA) techniques attempt to learn a model from training data, and then use the trained model to automatically assign semantic labels to unseen image (Bhagat and Choudhary, 2018; Shi et al., 2017). Instead of coarse-grained image annotation, more challengingly, pixel annotation is to label image in fine-grained pixel, where each pixel of the given image requires a label to be consistent with the desired object class. Obviously, it is the core of semantic object image segmentation in computer vision.

Vision-based fire detection technologies arise in recent decades (Bu and Gharajeh, 2019; Celik and Demirel, 2009), which mainly includes three steps for a fire monitoring system (Bu and Gharajeh, 2019): image region annotation for both fire and no-fire objects, moving object segmentation and decision on candidate regions. Region annotation is the key, which generates seed areas for the next detection. This step is usually guided by fire detection classifiers. In view of detection methods, it can be divided into two categories: color-based and motion-based ones (Hashemzadeh and Zademehdi, 2019a). The former focuses on color features from multiple color spaces, for example, pixel values from RGB space, and explores the separability between fire objects and no-fire ones. The latter aims to detect chaotic movement objects and then distinguishes fire objects from static or other no-fire moving objects (Foggia et al., 2015). The motion-based methods concentrate on detecting moving objects, whereas it also needs supervision information to discriminate the candidate object. Here so-called supervision information, such as the labels of training samples, is usually from color-based method, interactive image-preprocessing techniques, or even by manual. Compared to static background in the dead-directional videos, the fire objects are always viewed as a moving one, thus motion detection is expected to be used for real-time fire detection. Either color-based or motion-based one, undoubtedly, it is the base for them both on how to label samples from color information. On the other hand, it is noteworthy that motion-based methods may be efficient for detecting fire from indoor or city monitoring videos by dead-directional cameras, where most of no-fire objects appeared in the image are static, such as ground, sky, plants and shrubs, weeds, etc. But for forest monitoring system, it is quite different. In order to save human resource and material cost, instead of the dead-directional cameras, the system is always equipped with omni-directional dome (video)

* Corresponding author.
** Correspondence to: Room 50110E, the 5th Building of NFU, 159 Longpan Rd., Nanjing, China.
E-mail addresses: xbyang@njfu.edu.cn (X. Yang), zhangli@njfu.edu.cn (L. Zhang).

https://doi.org/10.1016/j.engappai.2021.104353
Received 20 October 2020; Received in revised form 15 March 2021; Accepted 14 June 2021
Available online 24 June 2021
0952-1976/

cameras, and usually installed on the hilltop watchtowers to cover wider spaces by scanning 360-angle degrees in horizontal direction and 180 degrees in vertical. That is, both foreground and background objects are changing at the same time in an omni-directional video. In this case, motion-based methods would fail to detect fire (Qureshi et al., 2016). In this paper, we focus on color-based methods for annotation.

For color-based fire detection, it involves in sample features description, color space transformation or selection, and classifier construction based on pixel-pattern samples in literatures. To our best knowledge, early it can go back to 2004. Chen et al. firstly provided several rules to detect fire pixels in RGB color space (Chen et al., 2004). Then in 2009, to well separate luminance from chrominance, Celik et al. introduced rule-based YCbCr (luminance, chrominance-blue and chrominance-red) color model (Celik and Demirel, 2009). They both concluded multiple rules for detecting fire pixels in RGB and YCbCr color spaces, respectively. Hereafter, such rule based methods are called rule-reasoning. Instead, Marbach et al. discussed the problem in YUV space and adopted YUV color model to represent video data, and classified candidate pixels to fire sector by luminance component (Y channel) and chrominance ones (U and V) (Marbach et al., 2006). Statistical analysis and experimental comparisons validated that it was more efficiently than RGB (Celik and Demirel, 2009). Yu et al. (2013) suggested that HSI (hue, saturation and intensity, also called HSL, where L means lightness) color model is more suitable for describing people-oriented color and experimentally provided threshold filters (piecewise function) for fire flame pixels. Instead of single color model, Qi and Ebert (2009) combined RGB with HSV (hue, saturation and value) saturation to construct sample features and proposed an algorithm for distinguishing non-fire moving objects by estimation of the spatial color variation in pixels.

The opinions from motion detectors mentioned that color-based detection is sensitive to the brightness, and slight scene change may cause big difference between detection results in different illumination conditions. However, when facing complex and multiple moving objects, for example, movements of weeds and woods by wind deformation in forest fire videos, it may result in high false alarm rate. Thus, some researches advised to bind color and motion together, and concentrated on the two-stepped methods (Hashemzadeh and Zademehdi, 2019a; Han et al., 2017). Moreover, for a tower-monitoring task, since motion detection fails to detect fire from background-changing video, color-based detection should be attached importance. For the foresaid rule-reasoning methods, actually, they follow the routing of pixel classification, where all pixels would be checked by the rules and only those pixels that pass the rules are detected as fire. Obviously, it is slow and hard to be detected by the batch. Instead of rule-reasoning, learning machine methods can do pixel detection in batch and thus undoubtedly speed pixel classification when it is well-trained. There exist a large amount of learning machines in literatures, supervised or unsupervised, and have been widely used for fire detection, including artificial neural networks (Maeda et al., 2009; Bui et al., 2019), SVM (Ko et al., 2009; Duong and Tinh, 2015), data clustering (Khatami et al., 2017a; Hashemzadeh and Zademehdi, 2019b), and deep learning based methods running on high-performance even powerful GPU-based computers (Muhanmad et al., 2019; Khatami et al., 2017b; Zhao et al., 2018), here we only named a few. In essence, they all are two-step methods. For example, for the SVM based fire detector (Duong and Tinh, 2015), the first step was to collect features from potential fire blobs and temporal change information of pixels, and accordingly formed two-class feature vectors (fire and no-fire training samples). Secondly, the classifier SVM was trained on the selected two-class samples and then applied to assign candidate objects to the class of fire or not. Similar to SVM, Hashemzadeh and Zademehdi introduced an unsupervised method for fire detection in 2019, named ICA K-medoids (Khatami et al., 2017a; Hashemzadeh and Zademehdi, 2019b). It firstly is to label samples by data clustering and image thresholding, and then is to construct classifier for detection. Nevertheless, the proposed methods seem scarcely aware of preconditions of the selected models.

In view of machine learning, model selection should be consistent with its hypotheses. Identity and independent distribution (i.i.d.), the key assumption on model paradises, is that the data should be generated from an unknown but fixed distribution. That is, intra-class samples are independently drawn from the same distribution. In this opinion, it is unreasonable for viewing forest fire detection as two-class classification, especially for the class of no-fire objects. It may be acceptable for the opinion supposing fire pixels from the same class, since they are generated from carbon-based combustibles. And in human vision, there has no much of color change for fire, mainly including yellow, red, and white. While for the class of no-fire, it is more complex. Intuitively, it is composed of complicated and colorful background objects. Thus, assigning no-fire objects to the same class, it is unacceptable.

Unbalance classification is another problem for fire detection. In the early stage of fire, the image area of fire pixels is obviously smaller that of no-fire. Training on such unbalance data would lead the classifier biased to the class of which have more training samples, which would result in high false alarm rate. Moreover, the digitization of color value is also device-independent. Different cameras have different number of image sensors, and each sensor contains hundreds of thousands of photosite in a lattice, covered by three types of filters to sense incoming light (spectral sensitivity). Therefore, the good fire detector should not be immutable but adjustable to fit new application scenarios.

In summary, the main differences between forest fire detection and machine learning classification lie in three-folds. (1) It cannot be simply viewed as a two-class classification problem, though detection objects are often described as “fire” or “no-fire”. (2) It is not appropriate to use an immutable classifier for all forest scenes. (3) it focuses first on fire detection rate and then on false alarm rate, but machine learning method looks them equally. Fire detection rate corresponds to true positive (TP, for shortly) in machine learning, and is defined as the decision when it is a fire. Likewise, false alarm rate corresponds to false positive (FP), is defined as the decision when it is no-fire but classified as fire. For a forest monitoring system, all fire should be detected out, while small number of false alarm can be tolerable. Hence, the goal of fire detection is to purse high fire detection rate and low false alarm rate. While for machining learning, it is different. The performance of classifier is usually evaluated by test correctness, where both TP and FN (false negative) are paid equal attention, so do TN (true negative) and FP.

In this paper, we will validate foresaid problems. Because there has no ground truth for fire images and videos. The evaluation for fire detectors still presently depends on the individual criteria in literatures. This motivates us focus on the image annotation. It is basic and important work, but also with heavy workloads. Inspiring by previous pixel classification, we attempt to annotate image in the level of pixel. Then regional annotation for fire- or no-fire-blocks will tend to easy. To meet complex forest environments, we will adopt polygonal regions based on convex hull instead of rectangle ones. Thus as a byproduct, a fast convex hull algorithm is derived. Based on selected pixel samples, distribution-free kNN (k nearest neighbor) based on K-D tree will be used for automatic pixel annotation, where both supervision samples and the number of classes can be visually and interactively determined by user.

The rest of paper is organized as follows. We briefly review related work in Section 2. The proposed pixel annotation will be detailed in Section 3, including convex region selection, the number of classes, and automatic pixel annotation. Experimental estimations are arranged in Section 4 on the public forest fire images and videos. In Section 5, we conclude the whole paper.

a) Original image b) Rule-reasoning c) SVM-based d) Our proposed

Fig. 1. Illustration for forest fire images and fire-detected results. (a) original RGB images; (b), (c) and (d) are the fire detected images by rule-reasoning, SVM and our proposed method.

2. Related work

As aforementioned, we briefly review color-based methods. There exit two kinds of distinctive fire detectors. One is from rule-reasoning (Celik and Demirel, 2009; Chen et al., 2004), and the other is two-step methods based on machine learning (Ko et al., 2009).

2.1. Rule-reasoning method

Instead of RGB color space, Celik et al. discussed the problem in the YCbCr space, and provided a set of rules for fire flame pixel detection. By separating the luminance from chrominance, the rules, Eq. (1), are reasoned from statistical experiences on a large set of images. Suppose an image pixel denoted by five-dimensional vector (i, j, Y, Cb, Cr), where (i, j) is the pixel coordinate and (Y, Cb, Cr) is the color values of the pixel. If the pixel is able to pass through five rules in Eq. (1), it would be classified to the class of fire; otherwise, it is no-fire.

$$\begin{aligned}
&\text{Rule1}: Y(i,j) > Cb(i,j)\\
&\text{Rule2}: Cr(i,j) > Cb(i,j)\\
&\text{Rule3}: (Y(i,j) > Y_{mean})\&(Cb(i,j) < Cb_{mean})\&(Cr(i,j) > Cr_{mean})\\
&\text{Rule4}: |Cb(i,j) - Cr(i,j)| \geq \tau\\
&\text{Rule5}: Cb(i,j) \in \Omega
\end{aligned} \tag{1}$$

where $Y(i,j)$, $Cb(i,j)$ and $Cr(i,j)$ denote color values in the corresponding color channels. The symbol "&" denotes logical "and" operator, and $|\cdot|$ denotes absolute value. Y_{mean} is the mean value of Y channel (luminance), so do Cb_{mean} and Cr_{mean}. The parameter τ in Rule4 is an empirical value, determined by ROC (receiver operating characteristics) curve. The set Ω is bounded by three polynomials using least-square estimation w.r.t. Cr.

2.2. SVM-based method

Different from the rule-based, generally, the machine learning methods are based on models, which need to be trained on the collected data before detection/prediction. As one of the most popular machines, SVM has been successfully applied in many fields. It was introduced into fire detection by Ko et al. in RGB color space. The leading SVM-based detector consists of two steps: fire-pixel selection and fire pixel detection. At the first step, fire pixels were selected by thresholding Gaussian probability distribution, such that,

$$\prod_{k\in\{R,G,B\}} p_k(I_k(i,j)) > \tau \tag{2}$$

where $p_k(\cdot)$ is the probability density function (pdf), and $I_k(i,j)$ is the color value from the kth color channel, $k \in \{R, G, B\}$. τ denotes a threshold. The second step is to extract features for SVM verification by 1-level wavelet transformed image besides color information. SVM model is defined as Eq. (3).

$$\begin{aligned}
&\min_{\boldsymbol{w},\mathrm{b}} \ \|\boldsymbol{w}\|_2^2/2 + C\mathbf{1}^T\boldsymbol{\delta}\\
&s.t. \ \ y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b) - 1 + \delta_i \geq 0\\
&\qquad \delta_i \geq 0, i = 1, 2, \ldots, l
\end{aligned} \tag{3}$$

where $\boldsymbol{x}_i$ is the ith samples corresponding to the foresaid ith pixel features, and $y_i(\in \{1, -1\})$ denotes its label, where 1 is for the fire class and -1 for the no-fire. δ denotes the slack variable. The first item in the objective of (3), $2/\|\boldsymbol{w}\|_2$, is the so-called margin. According to margin principles (Nello and John, 2000), large margin means good generalization. By the QP optimization, one obtains the optimal decision function

$$f(\boldsymbol{z}) = \boldsymbol{w}^T\boldsymbol{z} + b. \tag{4}$$

For an unknown sample $\boldsymbol{z}$, if $f(\boldsymbol{z}) \geq 0$, it was labeled with 1, corresponding to the class of "fire".

Fig. 1 illustrates a toy for fire detection on the given RGB images, where Fig. 1a is the original, and Figs. 1b, 1c and 1d are the fire pixels detected by the rule-based, SVM-based and our proposed methods, respectively, where the pixels detected as no-fire are filled with black pixel (RGB values (0,0,0)). Intuitively, our proposed method beats the other two in both high fire detection rate and low false alarm rate, as illustrated in Fig. 1d.

The toy says that, for the rule-based method, there appears black hollow region like "hole" in the central of fire flame. This means that it fails to detect highlight fire pixels located at the area of flame center. While for the SVM-based method, as illustrated in Fig. 1c, there are so many no fire pixels in the objects like sky, trees or grass, inopportunely appearing in the fire detection images, which means that SVM has high false alarm rate though almost all fire pixels can be well-detected. To achieve high fire detection rate, the decision plane of SVM has to biased to the class of fire. As for detection speed, the rule-reasoning method runs the slowest because each pixel need to be validated by so many foresaid rules. SVM runs the fastest just because it is validated by decision function. Fig. 1d also illustrates the fire detection result by our proposed method. It seems able to achieve both high fire detection rate and low false alarm rate. Next, we discuss it in the next section.

3. Our method

There have appeared so many methods for fire detection in literatures, especially in the rising of machine learning and computer vision, even including deep learning based methods in recent years. However, without fine-grained annotation, it is still unclear on how to define or label ground truth for forest fire images, including public available image databases. In this paper, we attempt to introduce a pixel-level image annotation to bridge the gap, guided by the goals of forest fire

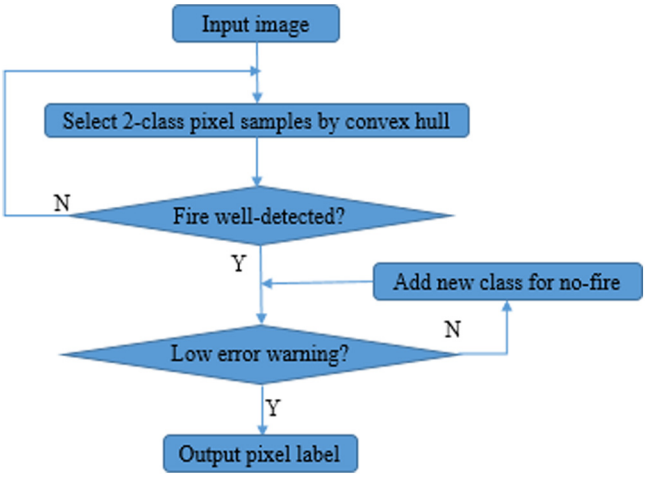


Fig. 2. Flowchart of the proposed fire pixel annotation.

detection. Our method is composed of three parts: selection for training pixels, model selection and automatic pixel annotation. Fig. 2 illustrates the flowchart, oriented from multi-class classification.

Without ground truth, the quality for pixel annotation mainly depends on human vision. According to human intuition, the optimal quality is that both goals, high fire detected rate and low false alarm rate, can be reached at the same time. However, in the real world it is very difficult. There has no choice but paying more attention on fire detection than on false alarm, as illustrated in Fig. 2, and we will not focus on false alarm until the fire can be well detected. In this case, if false alarm rate is still high, an improving strategy is to add a new class of no-fire samples with new label to training set, and then retrain the classifier. Repeat the steps until both goals approach. Thus, each pixel sample will be annotated with a label, typically, 1 or 0, where 1 is for the fire and 0 is for the no-fire. Note that here 0 is just a merged label from multiple no-fire classes. In experience, adopting multi-class method would be more suitable for uncertain complex scenes than classic two-class one, as showed in Fig. 1d, where they were viewed as three-class. Moreover, in view of machine learning, it is also able to avoid unbalance classification problem. We will detail them in next sections.

3.1. Samples selection by convex hull

In this subsection, we discuss how to select training samples easily and accurately. Compared to rigid objects, the object fire is a particular one. Physically, it is amorphous, tangible, but necessary to be accurately figured out from the image. For the shapeless objects, polygonal regions based on convex hull would be more suitable than usual rectangle ones. Fig. 3 illustrates three convex hulls marked with different colorful polygons. The red one is for fire regions, and both green and blue ones are for no-fire regions. The training samples will be generated from the pixels located at the regions, where pixels in the same region share a label. For example, all pixels in the red region are labeled with 1, and the pixels in the green and blue are labeled with 2 and 3, respectively. Those pixels in the selected regions will be used for training classifier. Naturally, how to rapidly obtain samples in the selected hull regions? Next, we provide a fast algorithm.

The algorithm is illustrated in Fig. 4, where all samples located in hull will be used to consist of training set. The key steps lie in two aspects: how to obtain convex hull and training samples inside the hull. The scatter points, marked red dots in Fig. 4a, stands for pixel coordinates, obtained by mouse click on a given image. Figs. 4b and 4c demonstrate the boundary points (hull vertexes) and the hull, marked with blue circles and magenta polygons, respectively. It can be proved

Fig. 3. Illustration for three convex hulls on the fire image, marked with colorful polygons.. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

that any points in the hull regions can linearly represented by those vertexes. Conveniently, a set of digits are showed in Fig. 4, texted with digits from 1 to 9 anticlockwise to describe the order of the hull vertexes. For an interactive algorithm, it is not difficult for user to point out the desired region vertexes. Next we come to select training samples in the hull. Fig. 4d to f demonstrate how to make a decision for a query point: inside or outside the hull.

Suppose that l ordered points, $\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_l$, are convex hull vertexes. here the order means clockwise or anticlockwise, since the vertexes are on the same plane. Without loss of generalization, we discuss the problem in d-dimensional space, where $\mathbf{z}_i \in R^d$, $i = 1, 2, \ldots, l$. The center of the hull, $\boldsymbol{m}$, is defined as

$$\boldsymbol{m} = \frac{1}{l}\sum_{i=1}^{l} \boldsymbol{z}_i \tag{5}$$

Hence, the hull is made up of l directed line segments, which anticlockwise directions can be described by vectors $\boldsymbol{z}_1 - \boldsymbol{z}_2, \boldsymbol{z}_2 - \boldsymbol{z}_3, \cdots, \boldsymbol{z}_l - \boldsymbol{z}_1$. Thus, there are l projections of $\boldsymbol{m}$ to the l line segments, and note them $\boldsymbol{q}_1, \ldots, \boldsymbol{q}_l$, respectively, as marked green square in Fig. 4f. According to the definition of projection, we have

$$\boldsymbol{q}_i = \boldsymbol{z}_i + \frac{< \boldsymbol{m} - \boldsymbol{z}_i, \boldsymbol{z}_{i+1} - \boldsymbol{z}_i >}{\|\boldsymbol{z}_{i+1} - \boldsymbol{z}_i\|^2} \cdot (\boldsymbol{z}_{i+1} - \boldsymbol{z}_i), \ i = 1, 2, \ldots, l \tag{6}$$

where $< \cdot, \cdot >$ denotes vector inner product, and $\boldsymbol{z}_{l+1} = \boldsymbol{z}_1$. Thus, we have the following linear equations for the l line segments[1]

$$\boldsymbol{p}_i^T(\boldsymbol{z} - \boldsymbol{z}_i) = 0, \ i = 1, 2, \ldots, l \tag{7}$$

where $\boldsymbol{p}_i = \boldsymbol{m} - \boldsymbol{q}_i$, the normal vector of the ith line, as illustrated in Fig. 4f. The superscript "T" denotes vector or matrix transpose throughout the paper.

For a given query point $\boldsymbol{v}$, define a function $\lambda(\boldsymbol{v})$ as below,

$$\lambda(\boldsymbol{v}) = \min_{1 \le i \le l} \{\boldsymbol{p}_i^T(\boldsymbol{v} - \boldsymbol{z}_i)\}. \tag{8}$$

Hence, the decision is that, it is inside the hull if $\lambda > 0$; it is on the hull if $\lambda = 0$; and outside hull when $\lambda < 0$.

To speed computation, we rewrite the Eq. (8) in matrix form. Let $\boldsymbol{Z} = (\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots, \boldsymbol{z}_l)$ and $\boldsymbol{Q} = (\boldsymbol{q}_1, \boldsymbol{q}_2, \ldots, \boldsymbol{q}_l)$. For the given point $\boldsymbol{v}$, define matrix

$$\boldsymbol{M} = (\mathbf{1}_l \boldsymbol{m}^T - \boldsymbol{Q}^T)(\boldsymbol{v}\mathbf{1}_l^T - \boldsymbol{Z}) \tag{9}$$

and

$$\lambda(\boldsymbol{v}) = \min_{1 \le i \le l} \{diag(\boldsymbol{M})\} \tag{10}$$

where $\mathbf{1}_l$ denotes the vector with all l entries 1s. The function $diag(.)$ denotes matrix diagonalization.

We conclude the foresaid in following theorems.

[1] The Eq. (7) is a line equation, which holds just in 2D space. For the d-dimensional case, strictly, it should be an intersection line between two orthogonal planes: Eq. (7) and the plane where the hull locates.

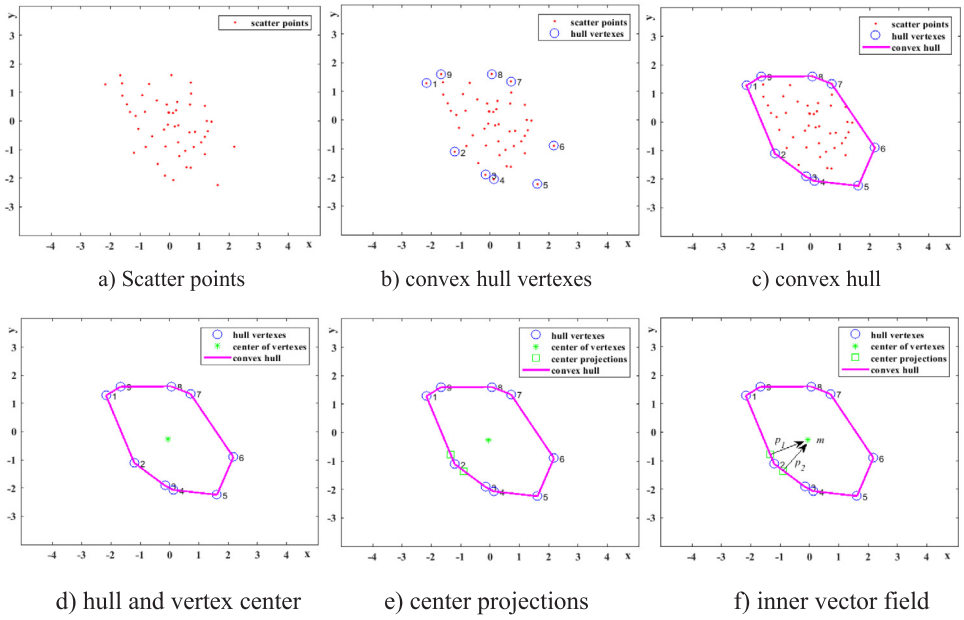


Fig. 4. Illustration for computing convex hull and its inner vector field.. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Theorem 1. *Given a set of convex hull vertexes in d-dimensional hyperplane,* $\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_l$*, the projections of the center* $\boldsymbol{m}$ *to the hull are* $\boldsymbol{q}_i = \boldsymbol{z}_i + \frac{\langle \boldsymbol{m}-\boldsymbol{z}_i, \boldsymbol{z}_{i+1}-\boldsymbol{z}_i \rangle}{\|\boldsymbol{z}_{i+1}-\boldsymbol{z}_i\|^2} \cdot (\boldsymbol{z}_{i+1}-\boldsymbol{z}_i)$*,* $i = 1, 2, \ldots, l$*, where* $\boldsymbol{m} = \frac{1}{l}\sum_i \boldsymbol{z}_i$ *and* $\boldsymbol{z}_{l+1} = \boldsymbol{z}_1$*.*

Theorem 2. *For any query point* $\boldsymbol{v}$ *and a given vertex set* $\mathbf{Z} = \{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_l\}$ *on the d-dimensional hyperplane, the relationship between* $\boldsymbol{v}$ *and the convex region* $\boldsymbol{R}$ *spanned by* $\mathbf{Z}$*,* $\boldsymbol{R} = \{\boldsymbol{r} | \boldsymbol{r} = \sum_{i=1}^{l} \alpha_i (\boldsymbol{z}_{i+1} - \boldsymbol{z}_i), \sum \alpha_i = 1, \forall \alpha_i \geq 0\}$*, is determined by Eqs.* (9) and (10)*. That is,* $\boldsymbol{v} \in \boldsymbol{R}$ *if* $\lambda \geq 0$*, and* $\boldsymbol{v} \notin \boldsymbol{R}$ *otherwise.*

For easy reading, we put the proofs in the appendix and conclude the foresaid steps in algorithm 1 named HullSampling.

As aforementioned in Section 2.1, define the five tuple for RGB image pixel is as (i, j, r, g, b), and rewrite it as vector pair $(\boldsymbol{z}, \boldsymbol{p})$, where $\boldsymbol{z} = (i, j)^T$ denotes pixel coordinate and $\boldsymbol{p} = (r, g, b)^T$ denotes pixel values from image color channels. Assume a pixel located at the position (i, j) of the image, its corresponding vector can be described as $(i, j, r_{ij}, g_{ij}, b_{ij})$, $1 \leq i \leq m$ and $1 \leq j \leq n$, where m and n denotes image resolution. Thus there are total mn pixel-vector samples, described by $\boldsymbol{S} = \{(\boldsymbol{z}_t, \boldsymbol{p}_t)\}_{t=1}^{mn}$.

3.2. Our approach

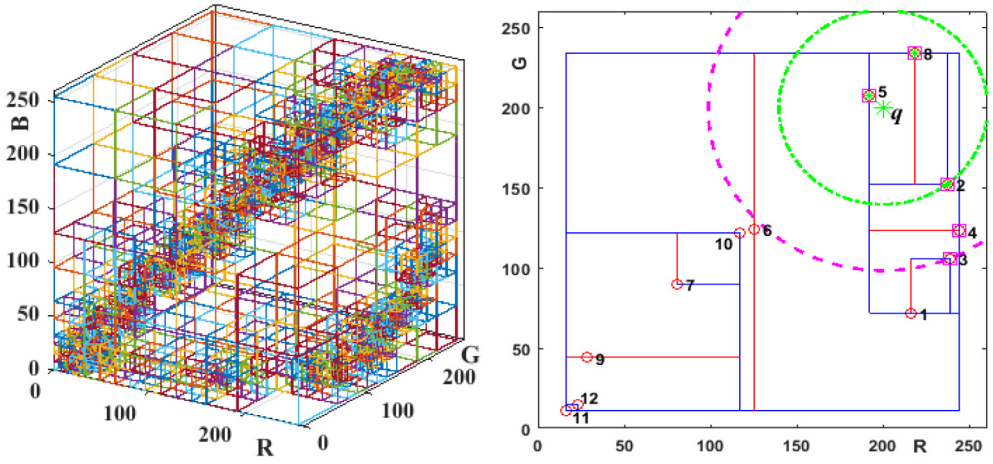
For the forest fire detection problem, the advantages of the kNN-based classifier lie in three-fold: (1) as one of the top ten data mining algorithms, it is very popular and widely used for data classification and regression; (2) kNN does not depend on any assumptions about the underlying data distribution, while model-based learning machines do, typically, the assumption for independent and identity distribution (*i.i.d*). In view of machine learning, originally, kNN is a model-free machine. It does not have training stage and conducts classification tasks by first calculating the distance between a test sample and all training samples to obtain its nearest neighbors, and then assigning the test sample with label by the majority rule on the labels of selected nearest neighbors. According to PAC (Probably Approximately Correct) principles, the key assumption for the models is that the data used for training and testing are generated independently and identically; (3) there are two theoretical evidences about probability bound of error of nearest neighbor decision rule. That is, for binary classification, when the number of samples tends to infinite, its probability error is descending to the optimal Bayes'; for multi-class classification, the upper probability error is bounded by TWICE the Bayes probability of error.

Since kNN needs to calculate the distances/similarities between a given point (also called query in information retrieval) and all training points and find the k closest ones, it is very computationally intensive, especially in high-resolution image data which may need millions of pixel queries. To descend computation distances and speed neighbor-finding steps, people deliberately add a training stage into traditional kNN with various tricks including partition trees, graph methods, hashing techniques and probabilistic approaches (Chen et al., 2019). KD-tree, as one of the most popular partition trees originated in 1975, adopts tree structure to store training points by axis-parallel subspace partitions. Owing to the tree properties, when the KD-tree is built, querying can be done quickly in the small portions of the search space. That is, for a test sample, finding its k nearest neighbors in ALL training samples in traditional kNN can be replaced in minority ones stored in KD-tree, where one just needs to query the minority of training points along subtrees of the trained KD-tree and thus the majority of search subspaces is pruned. Moreover, when adding a new training point to the exist KD-tree, one just need to do local adjustment for the subspace which the point belongs to, by traversing the tree, and starting from the root and moving to either the left or the right child depending on whether the point is on the left or right side of the splitting subspaces, until reach the subspace it should be located.

For different purposes, there have exist many variants in literatures since it was proposed, such as balanced KD tree, optimized KD tree and buffer KD-tree, etc. Here we only name a few. For easily understanding the application, we focus on the naive KD-tree itself in this paper. Finally, it is noteworthy that the general rule for KD-tree should satisfy $n \gg 2^d$, where n and d denote the number and the dimension of training samples, respectively. Based on the foresaid discussion, let us back to our fire detection method and begin with a toy. Fig. 5a illustrates an example for partitioned subspaces of KD-tree grown from 8086 training points. The training set consists of three-class selected RGB pixel values, marked "R", "G" and "B" in axis labels, from the image by three corresponding convex hulls in Fig. 4. The foresaid partitioned subspaces are demonstrated as the cuboids, which are also called cells, bins or nodes in the fields of machine learning. When the tree is built, it means that the kNN classifier is trained, where

Algorithm 1. HullSampling

Input: Pixel coordinates obtained by mouse click
Output: The set of pixel values, Ω, for training
Step1: Ordered the vertexes from input pixel coordinates and note them $Z=\{\mathbf{z}_1,\mathbf{z}_2,\cdots,\mathbf{z}_l\}$.
Step2: Compute $\boldsymbol{m}$ and $\boldsymbol{Q}$ by Eqs (1) and (2), and set $\Omega=\varnothing$.
Step3 One scanning until all image pixels being checked. #
Step3.1 Take the current pixel as $(\boldsymbol{v},\boldsymbol{p})$;
Step3.2 compute $\boldsymbol{M}$ and $\lambda(\boldsymbol{v})$ by Eqs (5) and (6);
Step3.3 if $\lambda\geq 0$, then $\Omega=\Omega\bigcup\{\boldsymbol{p}\}$; else goto next pixel;
Step4: Return Ω



a) 3-dimensional subspace partitions b) kNN query for the 2-dimensional case

Fig. 5. Illustration of subspace partition for KD-tree in RGB color space.. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

all training points have been stored in nodes of the tree. Fig. 5b is a 2-dimensional case (R and G components) for explaining how to label a query point, where points texted with digitals from "1" to "12" stands for training points from the training set. The solid red and blue lines correspond to 2-dimensional subspace partitions. A test sample $\boldsymbol{q}$, marked green star in Fig. 5b, is the candidate point to be labeled. Let its coordinate be (200, 200). It is easy to know which subspace it belongs to by starting query from the root (the red line where the point 6 locates), and then moving down the tree recursively until the point being inserted. Since the partitioned subspaces are all axis-aligned, searching the nearest neighbor is implemented by a simple comparison to the distance between the splitting coordinate of the search point and current node or the points on the other side of the splitting subspaces. Neighborhoods for the 3NN and 5NN of the $\boldsymbol{q}$, measured by Euclidian distance, are illustrated as circles/discs (spheres in higher dimensional space) in Fig. 5b marked green and magenta dashed line, respectively. The figure shows that its three nearest neighbors (3NN) are the training points "5", "8" and "2", while 5NNs include additional two points: "4" and "3". Thus, the point $\boldsymbol{q}$ can be labeled by the majority rule of the labels of selected training points. Moreover, the built KD-tree does not need to be readjusted as long as the training set is fixed, even if facing changed k, the number of nearest neighbors, as observed from Fig. 5b. Obviously, k plays an important role for kNN, which is related to the quality of sample annotation.

How to determine an appropriate k? There have plenty of methods in literatures (Zhang et al., 2018a), mainly include two parts: (1) assigning a fixed expert-predefined k value for all samples; and (2) assigning different k for different unknown samples. However, for a real application, the fixed kNN has been shown to be impractical. In decades, different k for kNN has become a hot topic in fields of machine learning. Li et al. (2008) advised to use different numbers of nearest neighbors for different classes, rather than a fixed number across all categories, while Góra and Wojna (2002) suggested to combine rule induction with instance-based learning to learn the optimal k values. Introducing statistical confidence into kNN, Wang et al. (2006) proposed a local adjustment for the number of nearest neighbors. In 2018, an variant KD-tree for kNN classification, named kTree (Zhang et al., 2018b), was proposed, and the optimal k values for each training sample can be learned from the reconstructed tree model. In addition, for different applications there exist many kNN versions related to the k, such as cost-sensitive kNN (Zhang, 2020), local mean representation Gou et al. (2019), data-driven version (Zhang et al., 2018c), etc.

In this paper, since we focus on pixel annotation by means of man–machine interactive mode, to make the operation easy, we would adopt simple method, for example, cross validation, for choosing k value, which will be detailed in the experiment section. We conclude the foresaid in algorithm 2, named kNNLabeling.

Fig. 6 illustrates Algorithm 2, kNNLabeling, for annotating a forest fire image with the size of 288*512. Conveniently, training samples in each class are from the same convex region, marked color convex hulls. Firstly, the kNN is trained by two-class samples, as illustrated in Fig. 6b, where the red hull is for fire and the blue one is for ground. Fig. 6c is the fire image detected by the trained two-class kNN classifier. Compared to the original image in Fig. 6a, though the fire is well detected, the no-fire objects like smoke and sky are error detected. According to metric similarity, one of the reasonable explanations is that the error detected

Algorithm 2. kNNLabeling

Input: RGB image matrix $\boldsymbol{A}$

Output: Pixel Annotation Matrix $\boldsymbol{L}$

Step1: Initialize c and $\boldsymbol{L}$. Set c=2 and $\boldsymbol{L}=\boldsymbol{0}$, where c denotes the number of classes;
Call HullSampling (algorithm 1) to obtain two-class training set Ω labelled with "1" and "2" from $\boldsymbol{A}$, where $\Omega=\Omega_1\bigcup\Omega_2$, and Ω_i denotes the i-th class samples.

Step2: Train kNN classifier on Ω by KD-tree T.

Step3: Classify and label all pixels by the trained kNN classifier, and show fire detected image.

Step4: If fire detection rate is high, then goto Step 5; else goto Step1.

Step5: If false alarm rate is low, then goto Step 8; else goto Step6.

Step6: Set c=c+1 and call algorithm 1 to obtain a new no-fire samples Ω_c labelled with c.

Step7: Set $\Omega=\Omega\bigcup\Omega_c$ and goto Step2.

Step8: Stop and return the annotation matrix $\boldsymbol{L}$, where no-fire pixel labels from 2 to c are replaced with 0.

objects have higher similarity with the labeled fire objects than that of ground. To remove sky-like objects from the detected image, a new class of samples are selected by cyan convex hull and added to the training set, as illustrated in Fig. 6d and e. Retraining the kNN classifier on the obtained three-class training samples, and classifying all pixel, the fire detected result in Fig. 6f shows that almost all of no-fire objects have been removed but a little of smoke-like ones. Repeat adding new samples and retraining the classifier, illustrated in Fig. 6g, the final fire detected image is showed in Fig. 6h. Compared to the original image, it seems satisfactory for the final image annotating fire pixels.

For a candidate pixel input (query point), $\boldsymbol{q}=(r,g,b)^T$, its output label $y(\in\{1,2,\ldots,c\})$ will be determined by querying the KD-tree built on the training set Ω. Let $N_k(\boldsymbol{q})=\{\boldsymbol{p}_1,\boldsymbol{p}_2,\ldots,\boldsymbol{p}_k\}$, $\boldsymbol{p}_i\in\Omega$, denotes its first k nearest neighbors (kNNs), measured by the Euclidean distance $\|\boldsymbol{p}-\boldsymbol{q}\|_2$. The set $\{y_1,y_2,\ldots,y_k\}$, $y_i\in\{1,\ldots,c\}$, denotes their labels are $y_1,y_2,\ldots,y_k$, $y_i\in\{1,\ldots,c\}$, obtained by the foresaid c-class hulls (in steps 1 and 6 of the algorithm 2).

Instead of traversing all training samples, here the querying just need to traverse a small branch from the root to the cell (minimal subspace) that covers $\boldsymbol{q}$, then backtracking from the cell to its ancestors to find its kNNs, as illustrated in Fig. 5b. Benefiting from KD-tree, thus the amount of computation can be greatly reduced than that of actually preformed on classic KNN. The output y for $\boldsymbol{q}$ can be determined by the following expression,

$$y=\arg\max_{i}\ \max_{1\le i\le c}\{\sum_{j=1}^{k}I(i==y_j)\} \tag{11}$$

where $I(x)$ is an indicate function and the x is logical variable. $I(x)=1$ when x is true, and 0 otherwise. The symbol "= =" means the logical equality. General speaking, the Eq. (11) says that the $\boldsymbol{q}$ is assigned to the class most common among its k nearest neighbors.

4. Experimental simulation

In this section, we perform a comparison of our proposed method and state-of-the-art, including rule-reasoning method (Celik and Demirel, 2009), and supervised or unsupervised machine learning methods. Here, we will report fire detection results compared with unsupervised K-medoids clustering (Hashemzadeh and Zademehdi, 2019a; Khatami et al., 2017c), and supervised SVM (Ko et al., 2009; Duong and Tinh, 2015). To train two-class SVM on the same c-class training set, the positive training samples are drawn from the fire, and the negative ones are from all no-fire classes. To avoid imbalance learning problem, two classes should receive equal attention by drawing nigh the same number of samples in a minority class to the majority class. To our best knowledge, there has no pixel-level ground-truth for fire images, thus the results will be represented mainly by visualization. The data, besides public forest fire videos like VisiFire or Mivia,[2] are also gathered from the internet with more complex forest occasions to validate our proposed suitability.

Table 1
Brief information for selected forest fire images.

Datasets	File format	Image resolution (Row × Column)	Number of samples (Pixels)
Im1	JPG	300 × 400	120 000
Im2	JPG	542 × 900	487 800
Im3	JPG	375 × 500	187 500
Im4	JPG	575 × 950	546 250
Im5	JPG	514 × 900	462 600
Im6	JPG	963 × 1400	1 348 200
Im7	JPG	333 × 500	166 500
Im8	PNG	282 × 472	133 104
Im9	JPG	288 × 512	147 456
Im10	JPG	662 × 1000	662 000
Im11	JPG	300 × 450	135 000
Im12	JPG	180 × 300	54 000

The following experiments are divided into three subsections. The first one is carried on the forest fire images in different forest occasions,[3] the second is on the seven public videos with a fixed-angle camera, and the third one, on the angle-rotating fire videos. All comparisons in this section are conducted on a Dell PC with Core 2 Quad CPU @ 2.83 GHz and 4G RAM, running Matlab 2017b in window 7 operation system.

4.1. Forest fire images

In order to represent changeable forest occasions, the gathered forest fire images are with close- or long-range background under different light conditions. For convenience, we name them from "Im1" to "Im12" in Table 1, where the item "Row*Column" stands for image resolution. For a 3-channel RGB image, to maintain high-definition image quality, people sometimes are unwilling to reduce image resolution to just meet the computational demand. Thus, it unavoidably occurs a big challenge for next training classifier and object recognition on a massive dataset, which generated from image pixels, especially for higher-quality images. For example, Im6, the image resolution 963*1400 means that it has 1348200 image pixels and forms the dataset with 1348200 pixel samples.

[2] Available at http://signal.ee.bilkent.edu.tr/VisiFire/ and https://mivia.unisa.it/, respectively.

[3] Images and videos, codes for annotation used in the paper available at https://github.com/xbyang1000/fire-pixel-annotation/.

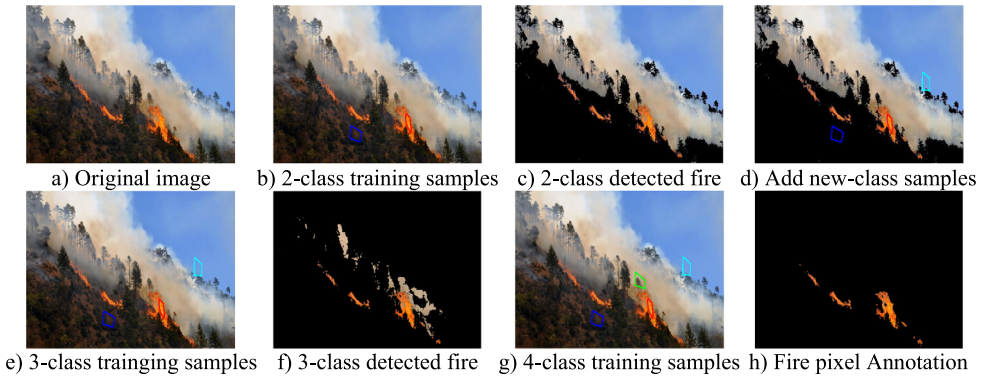


Fig. 6. Illustration for the algorithm kNNLabeling. The panel (a) is an original forest fire image; panels (b), (d), (e) and (g) demonstrate multi-class training samples generated from algorithm HullSampling (algorithm 1); and (c), (f) and (h) show the fire detected results by the corresponding multi-class kNN classifiers.. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Here we give an example. As illustrated in Fig. 6 on the data Im9, pixel-pattern samples are collected by 4-class hulls, spanned by their corresponding vertexes of user's mouse clicks. For a five-tuple pixel, (i, j, r, g, b), the location components (i, j) is used to determine whether it is selected or not by calculating the Eq. (10). If $\lambda((i, j)^T) \geq 0$, it means it is selected and added to training set Ω, and at the same time, its label, y, is also added to the label set Y. Thus, the training set Ω can be represented as $\Omega = \{(r_t, g_t, b_t)|\lambda(\boldsymbol{v}_t) \geq 0\}$, where $\boldsymbol{v}_t$ is the position vector, corresponding to the foresaid location (i, j). Here (r_t, g_t, b_t) is the tth sample of the training set Ω. For example, for a five-tuple pixel, (203, 220, 142, 175, 218), it is easy to know that it is located at the area of the 2nd hull (marked with blue polygon in Fig. 6b) by $\lambda((203, 220)^T) \geq 0$. Thus it is selected as the 200th sample in Ω, and simultaneously its label is set $y_{200} = 2$, to associate with its hull. As a result, there are total 1884 samples in Ω, selected from their corresponding 4-class hulls, and labeled them from "1" to "4". In training stage, a 3-dimensional KD-tree is built on Ω by setting $k = 3$ and Euclidian distance for similarity. In testing/detecting stage, for a candidate sample, saying (27, 21, 9), its 3NNs are all from the 4th class training samples by querying KD-tree from the root to its located cell/subspace, it is labeled with "4" by the majority vote and detected as no-fire. For sparse storage or visualization, replacing the labels "2" to "4" with "0", and reshaping the labels to matrix by the order of image, a logical matrix for annotation is obtained, where "1" is for fire, and "0" for no-fire.

The results are reported in Fig. 7. From left to right, the cells in each row are filled with original and fire images detected by our proposed, rule-reasoning, K-medoids clustering and SVM-based methods, respectively. Just for easy document-editing, those images are clipped into a uniform size, factually they are still at their original resolutions without loss of image pixel information.

Intuitively, our proposed method is able to achieve higher fire detection and lower false alarm than the other four methods, as illustrated in Fig. 7, where fire pixels can be well-detected and displayed in the detected images. Simultaneously, no-fire pixels seem to be removed from their original images, and instead filled with black pixels at the corresponding pixel positions. For the rule-based method, its fire detected results are also visualized in RGB color space by space transformation, though fire pixel detection is done in the YCbCr space. Compared to our proposed, the rule-based method fails to detect the fire pixels in the center of fire flame, and there have black hollows in the central part of flames in Im1, Im3, Im6 and Im12, as illustrated in Fig. 7. Moreover, it seems sensitive to the fire-like color, in Im5, where analog pixels are wrongly assigned to the class of fire. Since the fire detection rules are statistically induced from amount of images, it may be the reason that highlight pixels in white or sky-like objects are removed by the rules, no matter highlight objects like fire or not for a given image. This can be used to interpret why the fire pixels in the central of highlight flame are wrongly identified as no-fire ones. As for machine learning based methods, K-medoids and SVM, they both have high fire detection rates, where fire pixels can be well detected, but at the same time they also have high false alarm rates. Obviously, a leaning machine with higher false alarm rates would also lead to the failure of a fire-monitoring system. In fairness to all methods, they all should be trained on the same training set, without additional features but RGB three-channel color values. Recalling the Hashemzadeh and Zademehdi (2019a) and Ko et al. (2009), they also construct and utilize the first- or second-order statistics for fire detection. For unsupervised K-medoids, it receives the prompt of "out of memory" from Matlab on the data Im6 when all samples are used for training. Its corresponding fire detected result is obtained on the half of sample set. If without warning of memory limit, all samples would be used for data clustering. For SVM, in view of machine learning, it may result in under-learning problem when trained on small amount of training samples. In this paper, we focus on pixel annotation, and naturally try to make it easy for users in selecting supervised information by convex hulls. Hence, we have not provided enough supervised samples for training SVM. Since it is the first time to view fire detection as a multi-class classification problem, what we are wondering is its superiority to two-class method. Thus we take SVM-based method as the base, without more comparisons with multi-class SVM (Yan et al., 2018). On the contrary, it also provides an evidence that our proposed method is capable of better performance in both fire detection and false alarm, even if it is trained on a small-scaled training set.

Finally, we give an explanation for Im6, where some white pixels on the firefighting garments are wrongly classified as fire ones. As a contrast, Figs. 8a and 8c are the fire detected results by our proposed method and manual rectification for pixel annotation, respectively. To reveal the cause, two-class samples are selected by convex hulls, as illustrated in the green circles in Fig. 8b. One class is for the no-fire and consists of 27 no-fire samples, cropped from white characters on the garment; the other is for fire with 366 samples, cropped from the central of fire flame. Exploring similarity by Euclidean distance, unexpectedly, the minimal between-class distance is ZERO. This means that there must exist those samples having the same property values but from different classes.. As expected, the experimental result says that they share the same pixel values, (255, 252, 255), between the 9th sample in Class No-Fire and the 29th in Class Fire. Meanwhile, the maximum within-class distances are 15.84 and 35.33, respectively. That is, there exist some samples, which are more similar to the other class than its own. To fit the first goal of fire detection task, i.e. the fire pixel should be well-detected, it is the reason that some white-like pixels are wrongly identified as fire. Such explanation also can be used for the rule-based method, where fire pixels in highlight region are identified as no-fire ones, as illustrated in Fig. 7.

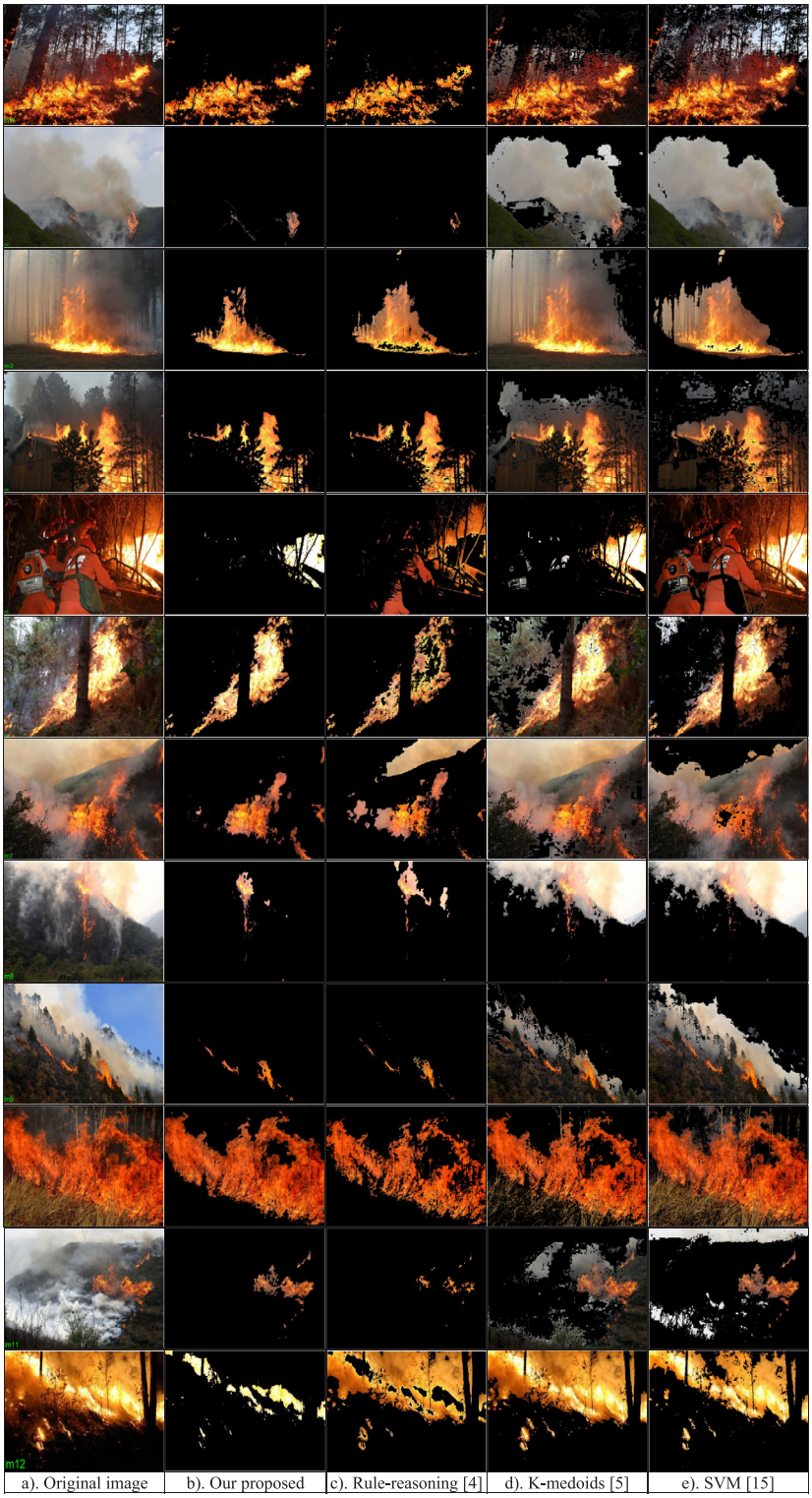


Fig. 7. Comparisons between our proposed and state-of-the-art, the left column (a.) are original images and next columns from (b.) to (e.) are the fire detected images by our proposed method, rule-reasoning, based K-medoids clustering and SVM-based fire detection methods, respectively.

In this case, if need to further lower false alarm, an efficient method is to introduce additional information, such as pixel neighborhoods, texture features, semantic information, and kernel-based machine learning methods, to raise the inter-class separability in the high-dimensional space. In this paper, we narrow the discussion for fire detection just in visual three-dimensional space. If possible, the authors advise to manually do the rectification for those pixel annotation, as done in Fig. 8c.

4.2. Dead-directional videos

In this subsection, the comparisons will be done on the forest fire videos from MIVIA database.

Opposite to the omnidirectional, the so-called dead-directional videos mean that they were obtained from angle-fixed cameras. Seven forest fire AVI videos, named fire3, fire4, fire5, fire7, fire8, fire10, and fire11, had been standardized at a fixed resolution 256*300 and

a). Fire detection result b). Two-class white pixels c). Corrected pixel annotation

Fig. 8. An explanation for false alarm problem on Im6. The left panel is the fire detected by the trained classifier, the middle demonstrates how to select two-class samples from highlight white regions by convex hulls, and the right one is the corrected pixel annotation by manual.. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

frame rate 15. For monitoring forest fire, what we concern are not only whether there is a fire, but also where it is. Obviously, they can be implemented if pixel annotation is well-done. Without pixel-level ground-truth for video frames, we ignore movement evaluation methods, for example, matrix difference (Foggia et al., 2015; Toreyin et al., 2006) for motion detection.

Unlike Wang et al. (2020), we aim to estimate the performance of pixel-level annotation method for complex forest scenes on both fire detection and false alarm rates, and at the same time, to show the locations of fire pixels. For supervised learning methods, the classifiers will be trained on the first frame of the corresponding video and then used for fire pixel detection on the rest unseen frames. While for unsupervised ones, they are run directly on the test frames. We report the fire detection results on the selected frames at the number of $f/4$, $f/3$, $f/2$ and f, where f denotes the total number of frames, as illustrated in Fig. 9.

Fig. 9 reports the comparison among four fire detection methods on forest fire videos. Both our proposed and SVM are trained on the first frame of the corresponding video, where training samples are selected by convex hulls. The selected four frames for test are drawn from the corresponding video in the sequence of the $f/4$, $f/3$, $f/2$ and f, as illustrated in Fig. 9b, where f denotes total number of the frames. The figures, from Fig. 9c to f, are fire detection results of the selected test frames. Visually, considering both high fire detection and low false alarm rates, our proposed method wins the best, SVM follows, and then the Rule-reasoning and K-medoids are the next. On the datasets Fire5, Fire8, Fire10 and Fire11, both SVM and our proposed have achieved similar fire detection performance, but significantly better than the other two. It shows that all fire regions in the test frames can be well-detected, and at the same time, no-fire regions are always ignored and showed black regions in the figures. This means they have both high fire detection rates and low false alarm rates. However, on the data Fire7, the detected figure for SVM also appears hollow region in the center of the highlight fire flame, similar to rule-reasoning method, which may result in missing detection of fire pixels.

4.3. Omni-directional videos

To validate the performance of fire detection on the changing background, the data for omni-directional videos is collected from Chinese CCTV MP4 video news,[4] reported on March 20, 2019. The scene of forest fire located at 42°25' N and 130°38' E, near to Jingxin Town on the border between China and Russia, southeast of Hunchun city of Jilin Province. The original RGB video of news lasts 3'57", 25 frames per second, and is made up of many clips, including spot coverage, fighting facilities preparation, personnel schedule and multiple fire video clips. The test videos for experiments are cropped from the original at the size of 288×354 from the original size at 480×354, where those unrelated clips or unrelated objects in the frame, subtitles or TV station logos, are removed. As a result, 7 clips are selected and named them Video1, Video2, ..., and Video7 at the time sequence of the news, as described in Table 2. In the selected videos, the differences between them and the foresaid dead-directional ones lie in changing of both prospects and background, even in the adjacent frames of the same second interval. Next, they will be used for experimental validation. Table 2 reports that the forest scene videos comprise close up and distant shots, different light conditions during the day or night. Several frames in the beginning or ending seconds are removed to make selected frame content consistent with the forest fire. For example, the Video1 lasts 12 s at the time interval between 55 and 67 s. Since the first 22 frames in the 55th seconds are not related to forest fire, they are excluded from the Video1. As a result, Video1 has 278 frames, not 300 (12*25).

Table 2
Brief information for the selected 7 video clips.

Datasets	Time interval of original video	Number of frames	Shot distance	Light condition
Video1	0'55"–1'07"	278	short	night
Video2	1'09"–1'21"	291	long	night
Video3	2'18"–2'28"	251	long	night
Video4	2'30"–2'34'	101	long	night
Video5	2'41"–2'48"	176	short	day
Video6	2'58"–3'13"	376	short	day
Video7	3'16"–3'22"	151	short	day

Following the above, the first frame is still used for sample selection and classifier training by Algorithm 1 and 2 for supervised methods. The rest frames are for test. Fig. 10 reports fire detection results on the $f/4$, $f/3$, $f/2$ and f frame.

Generally, Fig. 10 says that, among four fire detection methods, our proposed method wins the best, and then SVM follows, as illustrated in Fig. 10c and f. Intuitively, both of them are able to achieve high fire detection rate, where almost all of fire pixels can be well-detected in the figures. But for false alarm rate, SVM is defeated. Typically, some no-fire objects are wrongly detected as fire ones, as illustrated on Video5 and Video6, where no-fire objects on the fireman garment are also appeared in the results of fire detection. As always, it is hard for rule-reasoning method to detect fire located at highlight regions, as well as fire-like red objects. For the unsupervised K-medoids method, it is difficult for detecting fire if without extra information beyond pixels. Fig. 10e says its high false alarm rate would make K-medoids infeasible for fire detection, especially for the small areas of fire flame on Vedio2, Vedio3 and Vedio5.

We give an example for fire motion detection. Fig. 11 reports a comparison between dead-directional and changing-directional videos. The figures in the panel (c) of Fig. 11 is the fire motion detection by matrix difference, one of the popular video motion detection methods. The top figures are from the Fire4, one of the dead-directional videos, and the bottom ones are from changing-directional Video4. For better visualization, the motion detection is done between two time-discontinuous frames with large time interval, saying the $f/4$ and $f/3$ frame of the videos. The results are shown on Fig. 11c. The top one enables to correctly reflect the change of fire flame, where most of

[4] Available at https://www.ixigua.com/i6670257541009637901/?logTag=81NVZw9dirIrcd2v7ORXU.

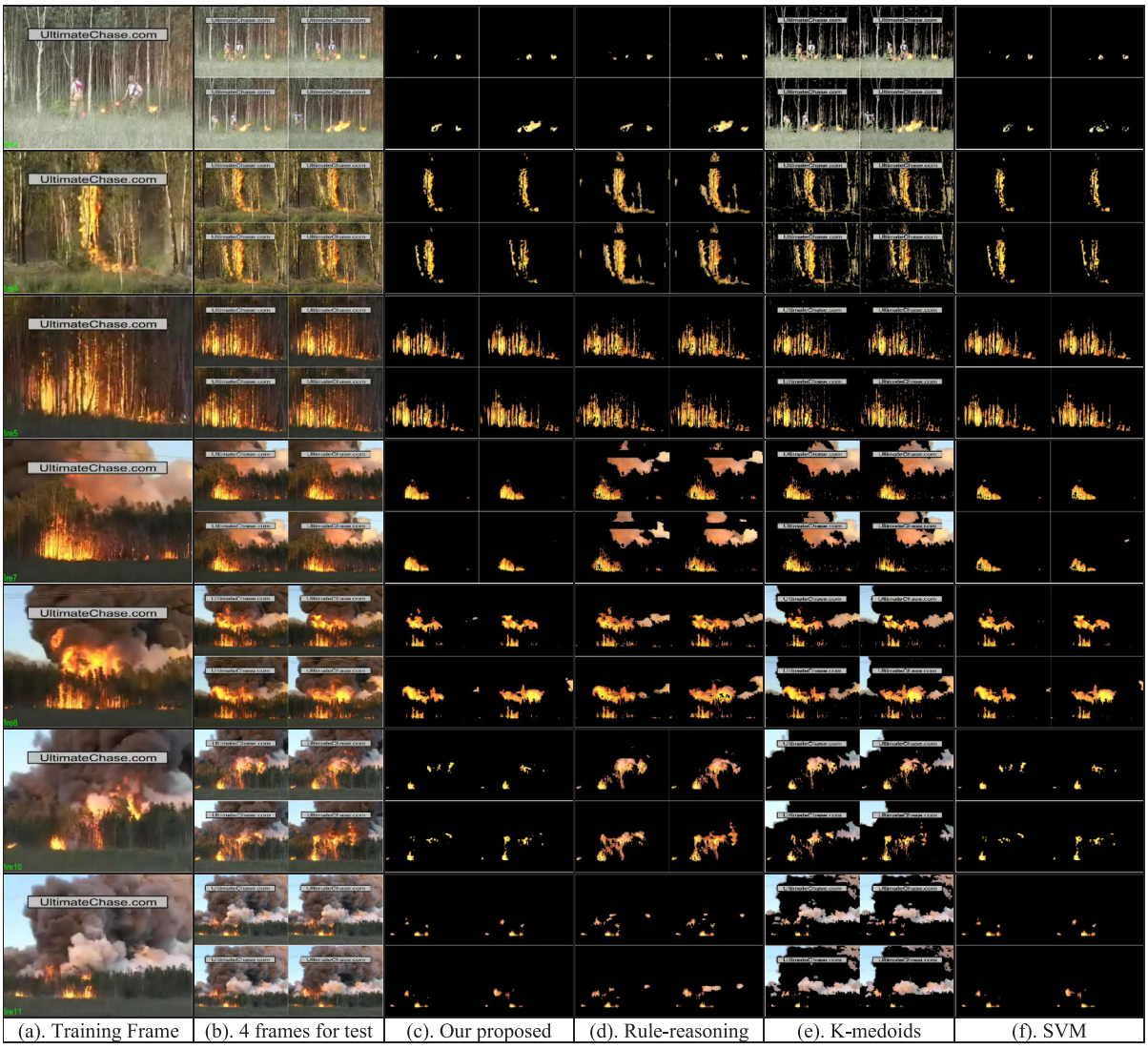


Fig. 9. The comparison of fire detection results on 7 MIVIA videos. The panel (a) illustrates the first frame for training classifiers from the corresponding video, where training samples are selected by our proposed convex hulls. The panel (b) demonstrates the test frames composed of four discontinuous frames. The panels, from (c) to (f), are fire pixels detected by the foresaid methods on the selected test frames.

the static no-fire objects are ignored. While for the bottom one, it is quite different. The motion detection method not only fails to reveal the tendency of fire flame, but also screws up the fire objects and no-fire objects, as illustrated in the bottom figure of Fig. 11c.

At the end of section, we discuss the computational complexity as below.

4.4. Computational complexity analysis

Similar to Chen et al. (2020), two indexes are adopted to discuss computational complexity: training time and test time (annotation time). The symbols, n, k, c and s, are the number of training samples, nearest neighbors, clusters and reasoning rules, respectively. In the training stage, our method is to build a KD-tree, while SVM need to solve a QP problem. Their time complexities are $O(n \log n)$ and $O(n^2 \sim n^3)$(the quadratic component for small C, and the cubic for large C, as described in Eq. (8)) (Chen et al., 2019; Léon and Lin, 2013), respectively. In test stage, for a given l test samples, the time complexity of kNN querying is $O(kl \log n)$, while SVM is $O(lt)$ by calculating l function values over t support vectors (generally $t \ll n$), as described in Eq. (8). The Park and Jun (2009) concluded that the complexity of K-medoids clustering tends to $O(c(l-c)^2)$. For the rule reasoning methods, as describe in Liu et al. (2016), its complexity is at least $O(l2^s)$, where a rule (expression) is identified to a logical symbol.

Run the programs 20 times on forest fire images and report time measured by CPU time at seconds. In Table 3, training time (TrTime) is time-consuming for training classifiers, and TeTime is for testing, where all pixel-level samples are used for test according to the order of pixels in the original image. The items named "Train set" and "Classes" denote the number of training samples and classes respectively, obtained from the proposed convex hull method.

Table 3 tells us that rule reasoning wins the fastest detection speed, then SVM and our proposed follows. As for training time in both supervised methods, our method almost superiors to SVM; but for test time, it is defeated. Note that image annotation aims to label image as accurate as it can. Then the annotated images will be used for ground truth for modeling or evaluating models or classifiers.

5. Conclusion

Suffering from no ground-truth for forest fire images, in this paper, we aim to directly annotate image in RGB color space without extra color space transformation. Considering the demands of the real-world forest fire monitoring, we propose an automatic image annotation

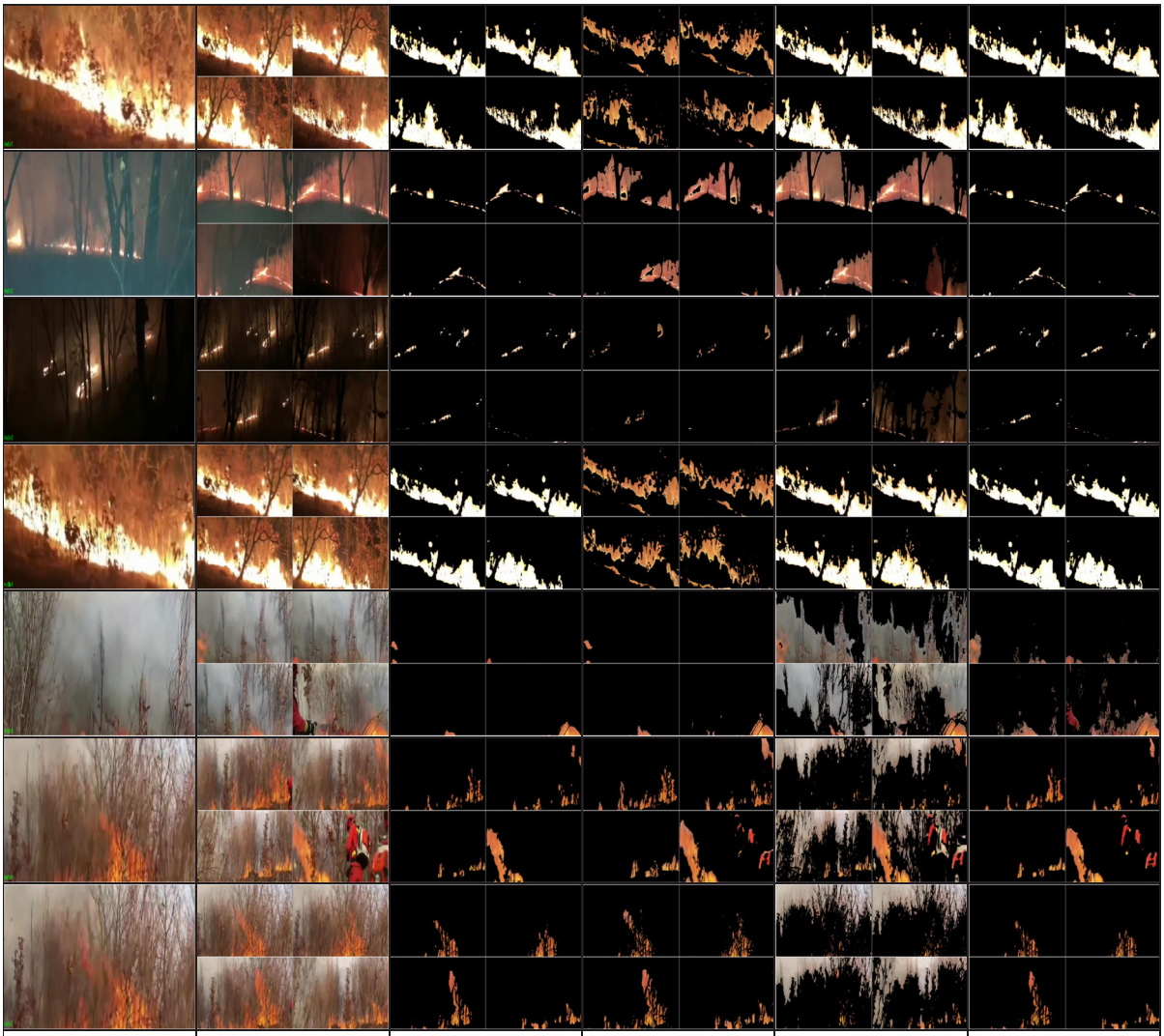


Fig. 10. The comparison of fire detection results on the selected 7 background-changing videos. The panel (a) is the first frame of each video, where training samples are selected for training classifiers by our proposed convex hulls. The panel (b) demonstrates the test frame composed of four discontinuous frames. The panels from (c) to (f), are fire pixels detected by the foresaid methods on the selected test frames.

Fig. 11. The fire motion detections between dead-background and changing-background videos. The top figures are from dead-directional video, while the bottom are from changing-directional one. Panels (a) and (b) show the first and second frame of the selected test frames, and the panel (c) is the results for motion detection.

Table 3
Time comparison between our proposed and the state-of-the-art.

Datasets	Train set/Classes	Test set	Our methods TrTime/TeTime	Rule-reasoning TeTime	K-medoids TeTime	SVM TrTime/TeTime
Im1	2424/3	120 000	1.257/0.045	0.312	254.437	4.571/25.459
Im2	1897/4	487 800	0.495/33.558	0.842	1445.817	1.607/35.631
Im3	2390/4	187 500	0.530/21.591	0.390	458.331	1.560/8.439
Im4	1438/3	546 250	0.421/67.892	0.889	2968.699	0.967/13.759
Im5	2391/4	462 600	0.562/33.946	0.796	1493.554	15.616/857.662
Im6	9626/5	1 348 200	0.608/148.055	2.246	–	21.060/46.862
Im7	4089/4	166 500	0.702/15.366	0.374	292.345	5.132/9.797
Im8	1383/3	133 104	0.218/11.482	0.343	282.674	0.795/6.084
Im9	1884/4	147 456	0.483/10.109	0.358	802.781	0.936/2.730
Im10	4392/3	662 000	0.608/79.389	1.217	5065.664	6.458/48.578
Im11	1707/3	135 000	0.484/8.002	0.327	501.075	0.998/4.072
Im12	652/3	54 000	0.483/5.397	0.249	71.760	0.390/1.700

"-" means the result is unavailable.

method in the level of pixel, termed as kNNLabeling. Instead of traditional view of binary classification, it absorbs multi-class thoughts into pixel annotation. To make it easier and more accurate, training samples can be selected interactively and intuitively by HullSampling, a proposed algorithm based on convex hulls, where each hull just relies on several convex vertexes. Thus K-d tree based KNN classifier can be trained on the selected training samples and then used for pixel annotation. Analysis and experiments show that, compared to the start-of-the-art, the proposed method is competent for pixel annotation on forest fire images and videos, at higher fire detection rate and at the same time lower false alarm rate, even on the more-challenging background-changing forest fire videos. Hence, it is reasonable that our methods would be useful for constructing ground-truth for the further research.

We should point out that, for simplify and ease of use, here the classifier for automatic annotation is just constructed on individual image pixel, without considering its neighbor pixels. Additionally, real-time detection is another important problem worthy of attention for forest fire monitoring systems. This is also our next focus. Herein what we focus on is how to make pixel annotation accurate as possible as we can, which would be useful for further automatic image block or region annotation.

CRediT authorship contribution statement

Xubing Yang: Conceptualization, Methodology, Paper writing. **Run Chen:** Experiment and Programming. **Fuquan Zhang:** Data preparation and curation. **Li Zhang:** Investigation and data visualization, Investigation. **Xijian Fan:** Experimental comparison, Validation, Supervision. **Qiaolin Ye:** Algorithm examination. **Liyong Fu:** Writing - review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

We would thank Dr. Pramod Vemulapalli and his group at Simon Fraiser University, for their kd-tree code. This research was supported in part by the Central Public-interest Scientific Institution Basal Research Fund, China (Grant No. CAFYBB2019QD003), Natural Science Foundation of China under Grant 31670554 and 61802193, the Fundamental Research Funds for the Central Universities, China (NJ2020023), and the Jiangsu Science Foundation, China under Grant BK20170934.

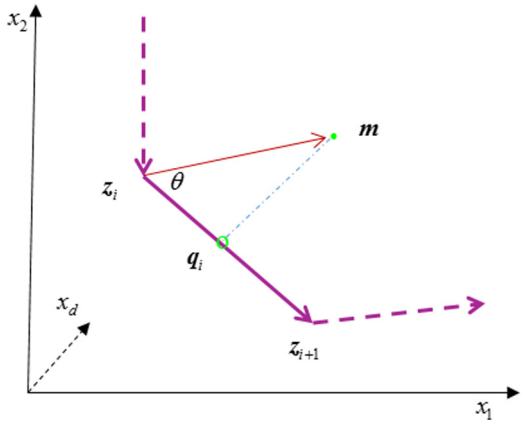


Fig. A.1. An example for finding the ith projection.

Appendix

Proof to Theorem 1. Support that $\boldsymbol{q}_i$ is the projection of $\boldsymbol{m}$ to the vector $\boldsymbol{z}_{i+1} - \boldsymbol{z}_i$, and θ denotes the angle between the vector $\boldsymbol{m} - \boldsymbol{z}_i$ and $\boldsymbol{z}_{i+1} - \boldsymbol{z}_i$, as illustrated in Fig. A.1, we have,

$$\cos\theta = \frac{\langle \boldsymbol{m} - \boldsymbol{z}_i, \boldsymbol{z}_{i+1} - \boldsymbol{z}_i \rangle}{\|\boldsymbol{m} - \boldsymbol{z}_i\| \cdot \|\boldsymbol{z}_{i+1} - \boldsymbol{z}_i\|} \tag{A.1}$$

such that,

$$\|\boldsymbol{q}_i - \boldsymbol{z}_i\| = \|\boldsymbol{m} - \boldsymbol{z}_i\| \cos\theta = \frac{\langle \boldsymbol{m} - \boldsymbol{z}_i, \boldsymbol{z}_{i+1} - \boldsymbol{z}_i \rangle}{\|\boldsymbol{z}_{i+1} - \boldsymbol{z}_i\|} \tag{A.2}$$

We have,

$$\boldsymbol{q}_i - \boldsymbol{z}_i = \|\boldsymbol{q}_i - \boldsymbol{z}_i\| \cdot \frac{\boldsymbol{z}_{i+1} - \boldsymbol{z}_i}{\|\boldsymbol{z}_{i+1} - \boldsymbol{z}_i\|} \tag{A.3}$$

Substituting (A.2) into (A.3), such that,

$$\begin{aligned} \boldsymbol{q}_i &= \boldsymbol{z}_i + \|\boldsymbol{q}_i - \boldsymbol{z}_i\| \cdot \frac{\boldsymbol{z}_{i+1} - \boldsymbol{z}_i}{\|\boldsymbol{z}_{i+1} - \boldsymbol{z}_i\|} \\ &= \boldsymbol{z}_i + \frac{\langle \boldsymbol{m} - \boldsymbol{z}_i, \boldsymbol{z}_{i+1} - \boldsymbol{z}_i \rangle}{\|\boldsymbol{z}_{i+1} - \boldsymbol{z}_i\|} \cdot \frac{\boldsymbol{z}_{i+1} - \boldsymbol{z}_i}{\|\boldsymbol{z}_{i+1} - \boldsymbol{z}_i\|} \\ &= \boldsymbol{z}_i + \frac{\langle \boldsymbol{m} - \boldsymbol{z}_i, \boldsymbol{z}_{i+1} - \boldsymbol{z}_i \rangle}{\|\boldsymbol{z}_{i+1} - \boldsymbol{z}_i\|^2} (\boldsymbol{z}_{i+1} - \boldsymbol{z}_i) \quad \square \end{aligned} \tag{A.4}$$

Proof to Theorem 2. Define line equations along the directed line segment $\overrightarrow{\boldsymbol{z}_i \boldsymbol{z}_{i+1}}$ as below,

$$l_i : (\boldsymbol{m} - \boldsymbol{q}_i)^T (\boldsymbol{x} - \boldsymbol{z}_i) = 0 \; i = 1, 2, \ldots, l \tag{A.5}$$

where $\boldsymbol{m} - \boldsymbol{q}_i$ is the normal vector of the line l_i.

Note the plane $\boldsymbol{P}$ as $\boldsymbol{P} = \{\boldsymbol{p} | \boldsymbol{p} = \sum_{i=1}^{l} \alpha_i (\boldsymbol{z}_{i+1} - \boldsymbol{z}_i), \forall \alpha_i \in R\}$, where R is real set. And note $\boldsymbol{R} = \{\boldsymbol{r} | \boldsymbol{r} = \sum_{i=1}^{l} \alpha_i (\boldsymbol{z}_{i+1} - \boldsymbol{z}_i), \sum \alpha_i = 1, \forall \alpha_i \geq 0\}$, $\boldsymbol{R} \subset \boldsymbol{P}$, corresponding to the region of the convex hull.

For $\forall \boldsymbol{v} \in \boldsymbol{P}$, as illustrated in Fig. A.1, $\boldsymbol{v}$ locates at the above of the line l_i (at the side of the hull) only and only if the angle between the normal vector $\boldsymbol{m} - \boldsymbol{q}_i$ and the vector $\boldsymbol{v} - \boldsymbol{z}_i$ is acute, such that the expression

$$(\boldsymbol{m} - \boldsymbol{q}_i)^T (\boldsymbol{v} - \boldsymbol{z}_i) \tag{A.6}$$

is greater than zero.

Thus, if the expression $(\boldsymbol{m} - \boldsymbol{q}_i)^T (\boldsymbol{v} - \boldsymbol{z}_i) > 0$ holds for all i, $i = 1, 2, \ldots, l$, it means that $\boldsymbol{v}$ is inside the hull, i.e., $\boldsymbol{v} \in \boldsymbol{R}$. Likewise, if one of them equals to zero, it is satisfied with the corresponding line equation A5, which means it is located at the hull. If exists some i, $i = 1, 2, \ldots, l$, there has $(\boldsymbol{m} - \boldsymbol{q}_i)^T (\boldsymbol{v} - \boldsymbol{z}_i) < 0$, it means that $\boldsymbol{v}$ is outside the hull.

Conclude the above and rewrite A6 for all i in matrix form. Define the matrix $\boldsymbol{M}$ and index function λ as

$$\boldsymbol{M} = (\boldsymbol{m}\boldsymbol{1}_l^T - \boldsymbol{Q})^T (\boldsymbol{v}\boldsymbol{1}_l^T - \boldsymbol{Z}) \tag{9}$$

and

$$\lambda(\boldsymbol{v}) = \min_{1 \leq i \leq l} \{diag(\boldsymbol{M})\} \tag{10}$$

where $\boldsymbol{1}_l$ denotes the vector with all l entries 1s. The $diag(.)$ denotes matrix diagonalization. The superscript "T" denotes matrix transpose.

Hence, for any $\boldsymbol{v}$, it is inside the hull if $\lambda(\boldsymbol{v}) > 0$; it is on the hull if $\lambda(\boldsymbol{v}) = 0$; outside of the hull otherwise. □

References

Bhagat, P.K., Choudhary, P., 2018. Image annotation: Then and now. Image Vis. Comput. 80, 1–23.

Bu, F., Gharajeh, M., 2019. Intelligent and vision-based fire detection systems: A survey. Image Vis. Comput. 91, 103803. http://dx.doi.org/10.1016/j.imavis.2019.08.007.

Bui, D., Hoang, N., Samui, P., 2019. Spatial pattern analysis and prediction of forest fire using new machine learning approach of Multivariate Adaptive Regression Splines and Differential Flower Pollination optimization: A case study at Lao Cai province (Viet Nam). J. Environ. Manag. 237, 476–487.

Celik, T., Demirel, H., 2009. Fire detection in video sequences using a generic color model. Fire Saf. J. 44, 147–158.

Chen, T., Wu, P., Chiou, Y., 2004. An early fire-detection method based on image processing. In: Proceeding of International Conference on Image Processing, Vol. 3. ICIP. pp. 1707–1710.

Chen, H., Yan, T., Zhang, X., 2020. Burning condition recognition of rotary kiln based on spatiotemporal features of flame video. Energy 211, 118656. http://dx.doi.org/10.1016/j.energy.2020.118656.

Chen, Y., Zhou, L., Tang, Y., et al., 2019. Fast neighbor search by using revised k-d tree. Inform. Sci. 472, 145–162.

Duong, H., Tinh, D.T., 2015. An efficient method for vision-based fire detection using SVM classification. In: Proceedings of IEEE Soft Computing and Pattern Recognition. Hanoi, Vietnam.

Foggia, P., Saggese, A., Vento, M., 2015. Real-time fire detection for video surveillance applications using a combination of experts based on color, shape and motion. IEEE Trans. Circuits Syst. Video Technol. 25 (9), 1545–1556.

Góra, G., Wojna, A., 2002. RIONA: A classifier combining rule induction and k-NN method with automated selection of optimal neighborhood. In: Proc. ECML. pp. 111–123.

Gou, J., Qiu, W., Zhang, Y., et al., 2019. A local mean representation-based K-nearest neighbor classifier. ACM Trans. Intell. Syst. Technol. 10 (3), 29.1–29.25.

Han, X., Jin, J., Wang, M., et al., 2017. Video fire detection based on Gaussian Mixture Model and multi-color features. Signal Image Video Proces. 11 (8), 1419–1425.

Hashemzadeh, M., Zademehdi, A., 2019a. Fire detection for video surveillance applications using ICA K-medoids-based color model and efficient spatio-temporal visual features. Expert Syst. Appl. 130, 60–78.

Hashemzadeh, M., Zademehdi, A., 2019b. Fire detection for video surveillance applications using ICA K-medoids-based color model and efficient spatio-temporal visual features. Expert Syst. Appl. 130, 60–78.

Khatami, A., Mirghasemi, S., Khosravi, A., Lim, C.P., Nahavandi, S., 2017a. A new PSO-based approach to fire flame detection using K-Medoids clustering. Expert Syst. Appl. 68, 69–80.

Khatami, A., Mirghasemi, S., Khosravi, A., Lim, C.P., Nahavandi, S., 2017c. A new PSO-based approach to fire flame detection using K-Medoids clustering. Expert Syst. Appl. 68, 69–80.

Khatami, A., Mirghasemi, S., Khosravi, A., et al., 2017b. A new PSO-based approach to fire flame detection using K-Medoids clustering. Expert Syst. Appl. 68, 69–80.

Ko, B., Cheong, K., Nam, J., 2009. Fire detection based on vision sensor and support vector machines. Fire Saf. J. 44, 322–329.

Léon, B., Lin, C.J., 2013. Support vector machine solvers. Large scale kernel machines. pp. 1–27.

Li, B., Chen, Y., Chen, Y., 2008. The nearest neighbor algorithm of local probability centers. IEEE Trans. Syst. Man Cybern. B 38 (1), 141–154.

Liu, H., Gegov, A., Cocea, M., 2016. Rule-based systems: a granular computing perspective. Granul. Comput. 1 (4), 259–274.

Maeda, E., Formaggio, A., Shimabukuro, Y., et al., 2009. Predicting forest fire in the Brazilian Amazon using MODIS imagery and artificial neural networks. Int. J. Appl. Earth Obs. Geoinf. 11 (4), 265–272.

Marbach, G., Loepfe, M., Brupbacher, T., 2006. An image processing technique for fire detection in video images. Fire Saf. J. 41 (4), 285–289.

Muhanmad, K., Ahmad, J., Lv, Z., et al., 2019. Efficient deep CNN-based Fire detection and localization in video surveillance applications. IEEE Trans. Syst. Man Cybern. A 49 (7), 1419–1434.

Nello, C., John, S., 2000. An Introduction to Support Vector Machines and other Kernel-Based Learning Methods. Cambridge University Press.

Park, H., Jun, C., 2009. A simple and fast algorithm for K-medoids clustering. Expert Syst. Appl. 36 (2), 3336–3341.

Qi, X., Ebert, J., 2009. A computer vision-based method for fire detection in color videos. Int. J. Imaging 2 (9), 22–34.

Qureshi, W.S., Ekpanyapong, M., Dailey, M.N., et al., 2016. Quickblaze: early fire detection using a combined video processing approach. Fire Technol. 52, 1293–1317.

Shi, Z., Yang, Y., Hospedales, T.M., et al., 2017. Weakly-supervised image annotation and segmentation with objects and attributes. IEEE Trans. Pattern Anal. Mach. Intell. 39 (12), 2525–2538.

Toreyin, B.U., Dedeoglu, Y., Gudukbay, U., et al., 2006. Computer vision based system for real-time fire and flame detection. Pattern Recognit. Lett. 27, 49–58.

Wang, J., Neskovic, P., Cooper, L., 2006. Neighborhood size selection in the k-nearest-neighbor rule using statistical confidence. Pattern Recognit. 39 (3), 417–423.

Wang, Y., Yu, Y., Zhu, X., et al., 2020. Pattern recognition for measuring the flame stability of gas-fired combustion based on the image processing technology. Fuel 270, 117486. http://dx.doi.org/10.1016/j.fuel.2020.117486.

Yan, C., Luo, M., Liu, H., et al., 2018. Top-k multi-class SVM using multiple features. Inform. Sci. 432, 479–494.

Yu, C., Mei, Z., Zhang, X., 2013. A real-time video fire flame and smoke detection algorithm. Procedia Engineering 62, 891–898.

Zhang, S., 2020. Cost-sensitive KNN classification. Neurocomputing 391, 234–242.

Zhang, S., Cheng, D., Deng, Z., et al., 2018c. A novel kNN algorithm with data-driven k parameter computation. Pattern Recognit. Lett. 109, 44–54.

Zhang, S., Li, X., Zong, M., et al., 2018a. Efficient kNN classification with different numbers of nearest neighbors. IEEE Trans. Neural Netw. Learn. Syst. 29 (5), 1774–1785.

Zhang, S., Li, X., Zong, M., et al., 2018b. Efficient kNN classification with different numbers of nearest neighbors. IEEE Trans. Neural Netw. Learn. Syst. 29 (5), 1774–1785.

Zhao, Y., Ma, J., Li, X., Zhang, J., 2018. Saliency detection and deep learning-based wildfire identification in UAV imagery. Sensors 18 (3), 712, 1-19.