Engineering Universe for Scientific Research and Management
ISSN (Online): 2319-3069
Vol. XV Issue II
February 2023

Content -Based Feature Extraction from Image: A Review

Ashish Sharma[1], Dr. Jitendra Sheetlani[2], Dr. Manish Shrivastava[3]
Research Scholar, SSSUTMS, Sehore, M.P. India, India[1]
Professor, SSSUTMS, Sehore, M.P. India[2]
Professor, Chameli Devi Group of Institution, Indore, M.P. India[3]

Abstract

In order to successfully accomplish classification and machine learning tasks, content-based feature extraction from the input image is necessary. Transforming a huge input image into a low-dimensional feature vector, which is an input to a classification or machine learning algorithm, is apparently involved in content-based feature extraction. The major difficulties with Content -Based Feature extraction have been mentioned in this study. One of the most important, captivating and revolutionary intellectual developments of all time has been the goal of universal education. The challenges of the present state are extremely complex, highly dimensional data and heterogeneous. Making appropriate decisions requires learning from the data and deriving wisdom from it. The purpose of this paper is to give an overview of methods used in Content -Based Feature extraction for various applications, an overview of the modern challenges in feature extraction. In this paper, we have reviewed various Content -Based Feature extraction techniques. These methods are classified as low-level feature extraction and high-level feature extraction. Low-level feature extraction is based on finding points, lines, edges, etc., while high-level feature extraction methods use low-level features to provide more important information for further processing of image analysis. Most high-level feature extraction methods use artificial neural networks (ANNs) to extract features across multiple layers.

Keywords: Image features, Feature extraction, Image classification, ANN

1. Introduction

The huge amount of Image generated has grown multiple times with millions or trillions of image sets generated every day with internet and social media. Content -Based Feature extraction is a critical task which involves huge amount of Image as input and transforming it in to optimal feature set. Special elements of an image are employed in image classification to distinguish between multiple images. These features are classified based on several essential image data elements, including color intensity, boundaries of objects in the image, texture, etc. [1]. The effectiveness of the feature extraction method extends the ability to process an image more effectively. These features can be used for image matching, pattern recognition, and information retrieval. For these applications to achieve a high degree of accuracy, the information must be clear and relevant. An input image is filled with duplicate and complex data. Transforming this data to limit a set of features (or feature vector) is the process of feature selection.

The field of Content -Based Feature Extraction From image is becoming more and more important in everyday life. There is a lot of talk about developing computer vision systems that would be able to identify road hazards, familiar faces or scenes in photos. The aim of this paper is to briefly introduce techniques that can be used to achieve the required level of object recognition.

2. Feature Extraction

Analysis is a process in which feature of the images are extracted and analyze for further processing. It is different from other image processing operations like restoration, coding and enhancement. Image analysis involves the detection, segmentation, extraction and classification techniques [2]. Feature extraction technique is used to extract the features by keeping as much information as possible from large set of data of image. Efficiency and effectiveness of feature selection and extraction are severe challenge nowadays. Numerous methods are used to extract features like color, texture and shape as feature vector. The techniques for feature extractions are classified are shown in Fig. 1.

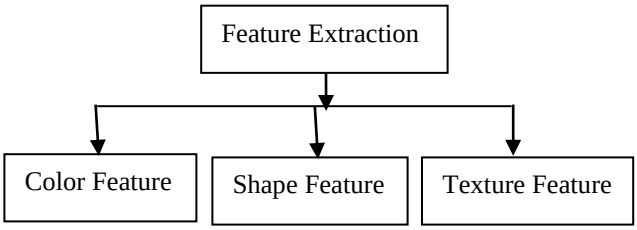

Fig. 1 Feature Extraction of Image

Feature extraction is the process of taking an input Image and correctly assigning it to one of the possible output classes. This process can be broken down into two general steps: feature selection and classification. Feature selection is critical to the entire process because the classifier will not be able to recognize poorly selected

features. Feature extraction is carried out after the preprocessing step in character recognition systems. The goal of feature extraction, a crucial stage in any pattern classification process, is to gather pertinent data that characterises each class. During the procedure, the characters'/objects' pertinent features are extracted to create a feature vector. The classifier matches up input units with target output units using these feature vectors. The classifier finds it considerably simpler to distinguish between distinct classes when these features are taken into consideration. Extraction of features the procedure of extracting the most crucial information from the raw data. Finding a set of factors that precisely and only determine a character's shape is the aim of feature extraction. Each character is represented by a feature vector during the feature extraction stage, which serves as its identity.

3. Literature Review

Various researchers are working on the methods of content-based feature extraction. It is a type of data reduction technique. Its objective is to reduce the data set of features which are valuable information present in an image. In image analysis, all the features of the images are separated in such a way that the class separation can be preserved well.

Different colour spaces are utilised to segregate the images at the beginning of the image analysis process. Various colour spaces exist, including RGV, LUV, HSV, and HMD [7]. The following methods are used to extract features from images: colour histogram, colour coherence vector, colour moment based, and colour sylogram. These methods are based on extracting the average, skewness, and standard deviation of each image pixel's intensity [7]. Any image that complied with the prerequisites was successfully and quickly detected by Colour Histogram as having colour distribution properties. The following criteria (Stability, Accuracy) are not met because it was unable to match the big collection of photos [8]. The most straightforward, compact, and reliable approach among these is colour moment feature extraction [9].

The outcome was unexpected, as can be seen in the colour space extraction. Only edges and corners may be detected during image analysis. Low-level characteristics that specify the geometry of objects are employed for edge detection, which creates a line drawing [10]. Power circumstances, consistent intensity, and noise levels all have a significant impact on how well the edge extraction feature works. an area sensor FAST (Features of Fast Segment Testing) is the name of the algorithm based on SUSAN (Smallest Unvalued Segment Assimilation Nucleus) [11]. FAST indicated that corners are one of the most intuitive forms of features that exhibit a substantial two-dimensional intensity change and are thus well distinguished from neighbouring locations [12]. Corner detection was also chosen over edge detection.

4. Feature Extraction Techniques

There are various types of feature extraction techniques in data science some of them are describing below:

Support Vector Machine

"Support Vector Machine" (SVM) is a supervised machine learning algorithm that can be used for both classification or regression challenges. However, it is mostly used in classification problems. In the SVM algorithm, we plot each data item as a point in n-dimensional space (where n is a number of features you have) with the value of each feature being the value of a particular coordinate. Then, we perform classification by finding the hyper-plane that differentiates the two classes very well.

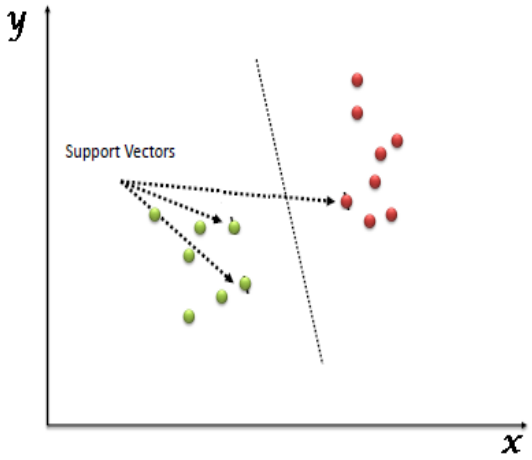

Fig.2 Support Vector Machine

Support Vectors are simply the coordinates of individual observation. The SVM classifier is a frontier that best segregates the two classes (hyper-plane/ line).

In the SVM classifier, it is easy to have a linear hyper-plane between these two classes. But, another burning question which arises is, should we need to add this feature manually to have a hyper-plane. No, the SVM algorithm has a technique called the kernel trick. The SVM kernel is a function that takes low dimensional input space and transforms it to a higher dimensional space i.e. it converts not separable problem to separable problem. It is mostly useful in non-linear separation problem. Simply put, it does some extremely complex data transformations, then finds out the process to separate the data based on the labels or outputs you have defined.

Neural Network

Feature extraction in neural networks contains the representations that are learned by the previous network to extract the interesting features from new samples. The features are then run through the new classifier which is already trained from scratch. Feature extraction, in case of the convents, consists of taking convolutional base of the previously trained network then running a new data through it and finally training the new classifier on top of output of the network. The first part of convnets, which contain a series of pooling and the convolution layers and finally connects with a classifier is called convolut Feature extraction is generally used on convolution base as convolution base are more generic than densely connected layers. Convolution base are reusable also. The feature maps of the convnet are the presence maps of generic concepts over a picture which is useful regardless of the computer-vision problem or any other problem.

In densely connected layers, feature extraction is not viable as it no longer contain any information about the objects that are located in input image as these layers get rid of where notion of space is whereas object location is still described by the convolutional feature maps exists. Densely connected features are useless where object location matters in problem.

Convolution bases of VGG16 network which is trained on ImageNet is popularly used for feature extraction. Among others are Xception, ResNet50, InceptionV3, VGG19, MobileNet model etc. ion base of the model.

Principal Component Analysis

Principal component analysis (PCA) is an unsupervised linear transformation technique which is primarily used for feature extraction and dimensionality reduction. It aims to find the directions of maximum variance in high-dimensional data and projects the data onto a new subspace with equal or fewer dimensions than the original one. In the diagram given below, note the directions of maximum variance of data. This is represented using PCA1 (first maximum variance) and PC2 (2nd maximum variance).

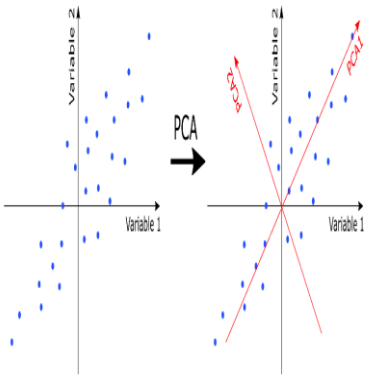

Fig. 3 PCA – Directions of maximum variance

The following represents 6 steps of principal component analysis (PCA) algorithm:

1. Standardize the dataset: Standardizing / normalizing the dataset is the first step one would need to take before performing PCA. The PCA calculates a new projection of the given data set representing one or more features. The new axes are based on the standard deviation of the value of these features. So, a feature / variable with a high standard deviation will have a higher weight for the calculation of axis than a variable / feature with a low standard deviation. If the data is normalized / standardized, the standard deviation of all fetaures / variables get measured on the same scale. Thus, all variables have the same weight and PCA calculates relevant axis appropriately. Note that the data is standardized / normalized after creating training / test split. **Python's sklearn.preprocessing StandardScaler** class can be used for standardizing the dataset.

2. Construct the covariance matrix: Once the data is standardized, the next step is to create *n X n*-dimensional covariance matrix, where *n* is the number of dimensions in the dataset. The covariance matrix stores the pairwise covariances between the different features. Note that a positive covariance between two features indicates that the features increase or decrease together, whereas a negative covariance indicates that the features vary in opposite directions. Python's **Numpy cov method can be used to create covariance matrix**.

3. Perform Eigen decomposition of covariance matrix: The next step is to decompose the covariance matrix into its **eigenvectors and eigenvalues**. The eigenvectors of the covariance matrix represent the principal components (the directions of maximum variance), whereas the corresponding eigenvalues will define their magnitude. **Numpy linalg.eig or linalg.eigh** can be used for decomposing covariance matrix into eigenvectors and eigenvalues.

4. Selection of most important Eigenvectors / Eigenvalues: Sort the eigenvalues by decreasing order to rank the corresponding eigenvectors. Select *k* eigenvectors, which correspond to the *k* largest eigenvalues, where *k* is the dimensionality of the new feature subspace (). One can used the concepts of **explained variance** to select the *k* most important eigenvectors.

5. Projection matrix creation of important eigenvectors: Construct a projection matrix, *W*, from the top *k* eigenvectors.

6. Training / test dataset transformation: Finally, transform the *d*-dimensional input training and test dataset using the projection matrix to obtain the new *k*-dimensional feature subspace.

Decision Tree

Decision trees are a method from the area of artificial intelligence and are used for machine learning. They are often binary trees, where each node has an if-then-else function on an attribute of the sample data. The ID3 algorithm (Iterative Dichotomiser 3, published by J Ross Quinlan in 1986, used to generate decision trees [25]) was the first algorithm to construct decision trees. ID3 had some problems and was improved. The improved version of ID3 is C4.5. It enhances the ID3 algorithm with the ability to handle both discrete and continues attributes, it can handle samples with missing attributes and supports pruning of the tree at the end of the algorithm (removing branches from the tree). Decision trees are in the proposed method used to calculate and order the features based on the information gain of each feature. During the method validation they are used for failure classification to show the influence of different features on the classification performance.

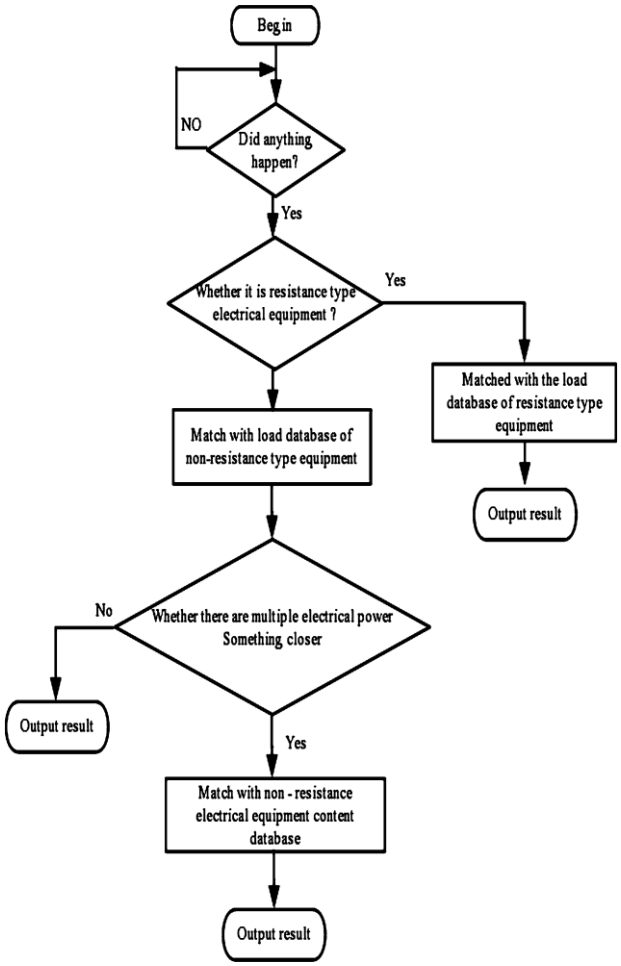

Fig. 4 Decision Tree Algorithm Flow Chart

The result of the algorithm is a binary decision tree, where the root of the tree is the attribute with the highest normalized information gain. Nodes in the following levels of the tree represent attributes with lower normalized information gain. If pure information gain is used for splitting, then classes with the most cases are favoured. Information entropy is the knowledge that is contained in an answer depending on one's prior knowledge. The less is known, the more information is provided. In information theory information entropy is measured in bits. One bit of information entropy is enough to answer a yes/no question about which one has no data.

Pruning is the reduction of the depth of a decision tree. The tree gets better at classifying unknown samples, but might get worse at classifying the test samples. Normally pruning increases the overall classification accuracy, but too much pruning can increase the number of false classifications. Decision trees are good for diagnostics in the context of condition monitoring. They classify data with low computation needs and the generated decision trees are highly comprehensible by humans. Another advantage of decision trees for condition monitoring is that they can be transformed into simple logical equations for each class that can be checked and modified by a human expert. Decision trees are used to solve a large variety of problem e.g. tag speech parts, land cover mapping [9], text mining [1] or condition monitoring.

K-Nearest Neighbor

The k-Nearest-Neighbors (kNN) method of classification is one of the simplest methods in machine learning, and is a great way to introduce yourself to machine learning and classification in general. At its most basic level, it is essentially classification by finding the most similar data points in the training data, and making an educated guess based on their classifications. Although very simple to understand and implement, this method has seen wide application in many domains, such as in recommendation systems, semantic searching, and anomaly detection.

As we would need to in any machine learning problem, we must first find a way to represent data points as feature vectors. A feature vector is our mathematical representation of data, and since the desired characteristics of our data may not be inherently numerical, preprocessing and feature-engineering may be required in order to create these vectors. Given data with N unique features, the feature vector would be a vector of length N, where entry I of the vector represents that data point's value for feature I. Each feature vector can thus be thought

of as a point in R^N.

Now, unlike most other methods of classification, kNN falls under lazy learning, which means that there is no explicit training phase before classification. Instead, any attempts to generalize or abstract the data is made upon classification. While this does mean that we can immediately begin classifying once we have our data, there are some inherent problems with this type of algorithm. We must be able to keep the entire training set in memory unless we apply some type of reduction to the data-set, and performing classifications can be computationally expensive as the algorithm parse through all data points for each classification. For these reasons, kNN tends to work best on smaller data-sets that do not have many features.

Once we have formed our training data-set, which is represented as an M x N matrix where M is the number of data points and N is the number of features, we can now begin classifying

There are two important decisions that must be made before making classifications. One is the value of k that will be used; this can either be decided arbitrarily, or you can try cross-validation to find an optimal value. The next, and the most complex, is the distance metric that will be used.

There are many different ways to compute distance, as it is a fairly ambiguous notion, and the proper metric to use is always going to be determined by the data-set and the classification task. Two popular ones, however, are Euclidean distance and Cosine similarity.

Euclidean distance is probably the one that you are most familiar with; it is essentially the magnitude of the vector obtained by subtracting the training data point from the point to be classified.

$$E(x, y) = \sqrt{\sum_{i=0}^{n} (x_i - y_i)^2}$$

Another common metric is Cosine similarity. Rather than calculating a magnitude, Cosine similarity instead uses the difference in direction between two vectors

$$\text{similarity} = \cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|}$$

Choosing a metric can often be tricky, and it may be best to just use cross-validation to decide, unless you have some prior insight that clearly leads to using one over the other. For example, for something like word vectors, you may want to use Cosine similarity because the direction of a word is more meaningful than the sizes of the component values. Generally, both of these methods will run in roughly the same time, and will suffer from highly-dimensional data.

5. Conclusion

Image processing is a very developed and widespread field. Feature retrieval techniques help to make the process of object identification faster and more reliable. This article presented a brief overview of current techniques. Many more exist or are being developed, yet those presented here already offer a place to start.

Reference

[1] Dong ping Tian, “A Review on Image Feature Extraction and Representation Techniques”, International Journal of Multimedia and Ubiquitous Engineering Vol. 8, No. 4, July, 2013,pp 385-396.

[2] Revathi, R., & Hemalatha, M. An Emerging Trend of Feature Extraction Method In Video Processing. Cs & It-Cscp, 2012 69–80.

[3] Jauregi, E., Lazkano, E., & Sierra, B. (2010). Object recognition using region detection and feature extraction. Towards Autonomous Robotic Systems, 1, 104–111

[4] Egmont-petersen, M., Ridder, D. De, & Handels, H. (2001). Image processing with neural networks – a review. Pattern Recognition, (Section 3).

[5] P.A. Devijver, J. Kittler, Pattern recognition: a statistical approach, Englewood Cliffs, London, 1982

[6] K. Fukunaga, Introduction to statistical pattern recognition, 2nd ed., Academic Press, New York, 1990.

[7] Pachouri, K. K. (2015). A Comparative Analysis & Survey of various Feature Extraction Techniques, 6(1), 377–379.

[8] Jinxia L, Yuehong Q. Application of SIFT feature extraction algorithm on the image registration. In: Tenth international conference on electronic measurement & instruments IEEE; 2011.

[9] Krishnan, K. Sreekumar, “A Survey on Image Segmentation and Feature Extraction Methods for Acute Myelogenous Leukemia Detection in Blood Microscopic Images”, International Journal of Computer Science and Information Technologies, Vol. 5 (6) , 2014, 7877-7879

[10] Zhang, D., & Lu, G. (2004). Review of shape representation and description techniques. Pattern Recognition, 37(1), 1–19. https://doi.org/10.1016/j.patcog.2003.07.008

[11] Rosten E, Porter R, Drummond T. FASTER and better: a machine learning approach to corner detection. IEEE Trans Pattern Anal Mach Intel 2010; 32: 105–19.

[12] El-Gayar, M. M., Soliman, H., & Meky, N. (2013). A comparative study of image low level feature extraction algorithms. Egyptian Informatics Journal, 14(2), 175–181.

https://doi.org/10.1016/j.eij.2013.06.003
[13]Pachouri, K. K. (2015). A Comparative Analysis & Survey of various Feature Extraction Techniques, 6(1), 377–379.
[14]Erkan Bostanci," Spatial Statistics of Image Features for Performance Comparison", IEEE Transactions On Image Processing, VOL. 23.
[15]D. Zhang and G. Lu, "Review of shape representation and description techniques", Pattern Recognition, vol. 37, no. 1, (2004), pp. 1-19.
[16]R.W. Brause, M. Rippl, Noise suppressing sensor encoding and neural signal ortho normalization, IEEE Trans. Neural Networks 9 (4) (1998) 613–628.
[17]E. Oja, A simplified neuron model as a principal component analyzer, Journal of Mathematical Biology 15 (3) (1982) 267-273.
[18]E. Oja, Neural networks, principal components, and subspaces, International Journal of Neural Systems 1 (1) (1989) 61-68.
[19]P. Baldi, J. Hornik, Neural networks and principal component analysis: learning from examples without local minima, Neural Networks 2 (1) (1989) 53-58
[20]M. Kramer, Nonlinear principal component analysis using autoassociative neural networks, American Institute of Chemical Engineers Journal 37 (2) (1991) 223-243.
[21]S. Usui, S. Nakauchi, M. Nakano, Internal color representation acquired by a five-layer neural network, Proc. International Conference on Artificial Neural Networks, Helsinki, Finland, 1991, pp. 867-872.
[22]T. Hastie, W. Stuetzle, Principal curves, Journal of the American Statistical Association 84 (406) (1989) 502-516.
[23]G.E. Hinton, P. Dayan, M. Revow, Modelling the manifolds of images of handwritten digits, IEEE Transactions on Neural Networks 8 (1) (1997) 65-74.
[24]H.M. Abbas, M.M. Fahmy, Neural networks for maximum likelihood clustering, Signal Processing 36 (1) (1994) 111-126
[25]J.M. Cruz, G. Pajares, J. Aranda et al., Stereo matching technique based on the perceptron criterion function, Pattern Recognition Letters 16 (9) (1995) 933-944.
[26]M. Fukumi, S. Omatu, Y. Nishikawa, Rotation invariant neural pattern recognition system estimating a rotation angle, IEEE Transactions on Neural Networks 8 (3) (1997) 568-581
[27]Shustorovich, A subspace projection approach to feature extraction - the 2-D Gabor transform for character recognition, Neural Networks 7 (8) (1994) 1295-1301.
[28]P.N. Suganthan, H. Yan, Recognition of hand printed Chinese characters by constrained graph matching, Image and Vision Computing 16 (3) (1998) 191-201.
[29]M.N. Ahmed, A.A. Farag, Two-stage neural network for volume segmentation of medical images, Pattern Recognition Letters 18 (11-13) (1997) 1143-1151.
[30]M.D. Garris, C.L. Wilson, J.L. Blue, Neural network- based systems for handprint OCR applications, IEEE Trans. Image Process. 7 (8) (1998) 1097–1112.
[31]D. Tzovaras, M.G. Strintzis, Use of nonlinear principal component analysis and vector quantization for image coding, IEEE Trans. Image Process. 7 (8) (1998) 1218–1223.
[32]P.A. Devijver, J. Kittler, Pattern Recognition: A Statistical Approach, Englewood CliIs, London, 1982. G. Hauske, A self organizing map approach to image quality, Biosystems 40 (1) (1997) 93102.