Hindawi
Complexity
Volume 2017, Article ID 3169149, 12 pages
https://doi.org/10.1155/2017/3169149

Research Article

Deep Hierarchical Representation from Classifying Logo-405

Sujuan Hou,[1,2] Jianwei Lin,[1] Shangbo Zhou,[3] Maoling Qin,[1] Weikuan Jia,[1] and Yuanjie Zheng[1,2,4]

[1] *School of Information Science and Engineering, Shandong Normal University, Jinan 250014, China*
[2] *Institute of Life Sciences at Shandong Normal University, Jinan 250014, China*
[3] *School of Computer Science, Chongqing University, Chongqing 400030, China*
[4] *Key Laboratory of Intelligent Information Processing at Shandong Normal University, Jinan 250014, China*

Correspondence should be addressed to Yuanjie Zheng; zhengyuanjie@gmail.com

Received 30 June 2017; Accepted 29 August 2017; Published 10 October 2017

Academic Editor: Jia Wu

We introduce a logo classification mechanism which combines a series of deep representations obtained by fine-tuning convolutional neural network (CNN) architectures and traditional pattern recognition algorithms. In order to evaluate the proposed mechanism, we build a middle-scale logo dataset (named Logo-405) and treat it as a benchmark for logo related research. Our experiments are carried out on both the Logo-405 dataset and the publicly available FlickrLogos-32 dataset. The experimental results demonstrate that the proposed mechanism outperforms two popular ways used for logo classification, including the strategies that integrate hand-crafted features and traditional pattern recognition algorithms and the models which employ deep CNNs.

1. Introduction

A logo is a symbolic representation of any enterprise or organization or institution, which symbolizes the product or service of their respective work. Logos can be composed of a glyph, a textual message, an icon, or an image, depicted in various colors and styles. Detection and recognition of logos has always been important in a wide range of applications, such as product or brand identification, copyright infringement detection, contextual advertise placement, vehicle logo for intelligent traffic-control system [1], and brand-related statistics from social media streams [2]. At present, with the rapid development of multimedia information technology, the amount of logo data on the Internet continues to grow. Because of the surge in the amount of logos, designing effective management tools and systems is becoming imperative. This paper focuses on developing a fundamental tool for organizing logos by classifying them. Categorizing makes browsing and searching for logos more efficient and facilitates the development of related applications. For instance, in order to ensure the originality and uniqueness, when creating a logo for a new product or organization, it would be useful to be able to search through similar products or organizations to avoid trade infringement or duplication.

According to Bengio et al. [3], learning representation of the data makes it easier to extract useful information when building classifier. Hence, the success of classification algorithm largely depends on data representation because different representations can entangle and hide more or less the different explanatory factors of variation behind the data. At present, the study of representation for classification has attracted considerable attentions and it has had extensive applications, such as graph representation [4, 5] for classification, advertising video representation [6] for classification, logo classification [7–9], and other classifications employing various technologies, for example, bag mapping for the multi-instance learning [10]. Regarding the logo classification, Neumann et al. [7] classify the logos of University of Maryland logo database by combining local and global shape features. Sun and Chen [8] design a logo classification system to differentiate the logo images captured through mobile phone cameras with a limited set of images. Kumar et al. [11] propose a logo classification system based on the appearance of logo

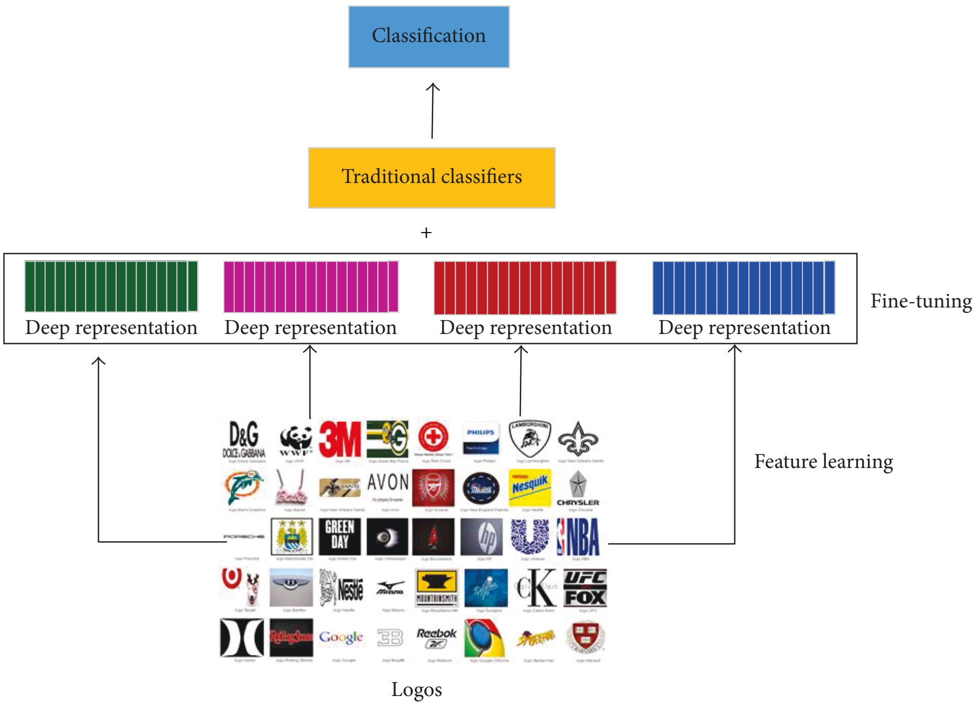


Figure 1: Overview of the proposed scheme.

images, which makes use of global characteristics of logo images for classification, like color, texture, and shape.

However, the success of most of existing work on classification, including logo classification, which adopts traditional pattern recognition algorithms primarily depend on the chosen class of features. These chosen features usually tend to be hand-crafted. A recent advance has been the use of deep neural networks to automate visual feature extraction in various domains. In particular, methods that use the convolutional neural network (CNN) model have achieved state-of-the-art results in computer vision tasks. However, as we know, training deep neural network is difficult due to its tendency to have many local optima. Nair and Hinton [12] address this problem by pretraining the deep model, which is called "greedy layerwise training." Recently, Bianco et al. [13] present a recognition pipeline specifically for logo using deep learning, which is composed of a logo region proposal followed by a CNN.

Considering that the methods adopting a CNN model have shown good performance in image style classification as well when pretrained modes are sufficiently fine-tuned, in this paper, we propose a mechanism that makes full use of both the advantages of fine-tuning CNN models and traditional pattern recognition algorithms for logo classification task. Specifically, we firstly fine-tune several of important deep learning models to obtain the logo representations and then combine the learned logo representations into traditional classification algorithms. Due to the limited amount of training data available for logo task, the deep models work on networks pertained on other large-scale image datasets. The contribution of this work is twofold:

(1) We build a publicly available logo dataset (named Logo-405), which can be shared in the research of logos.

(2) We present a logo classification mechanism that combines both the advantages of deep hierarchical convolutional neural networks and traditional pattern recognition algorithms.

The remainder of this paper is organized as follows: Section 2 provides a description of the proposed mechanism; the experimental results and analysis are presented in Section 3; and Section 4 concludes this paper.

2. Proposed Approach

2.1. Overview. Figure 1 illustrates an overall workflow of the proposed scheme. It contains two stages; they are (1) feature learning phase, in which several deep representations for each logo are obtained by fine-tuning four popular deep convolutional network architectures and (2) classification phase, where the logo classification task is carried out by combining both the learned deep representations and traditional classification algorithms.

The proposed scheme combines both advantages of convolutional neural network in feature learning and traditional classification algorithm. During which four popular deep convolutional neural network architectures are firstly fine-tuned on our logo dataset (i.e., Logo-405) and one publicly

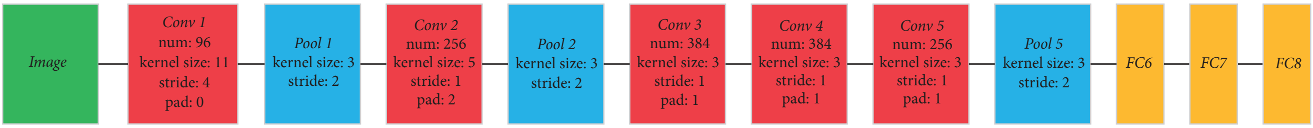


Figure 2: Network architecture of AlexNet [14].

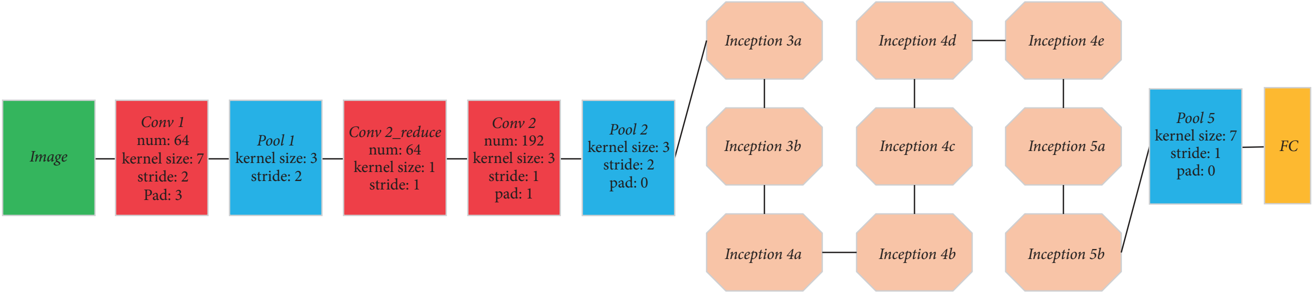


Figure 3: Fine-tuned network architecture of GoogleNet [15].

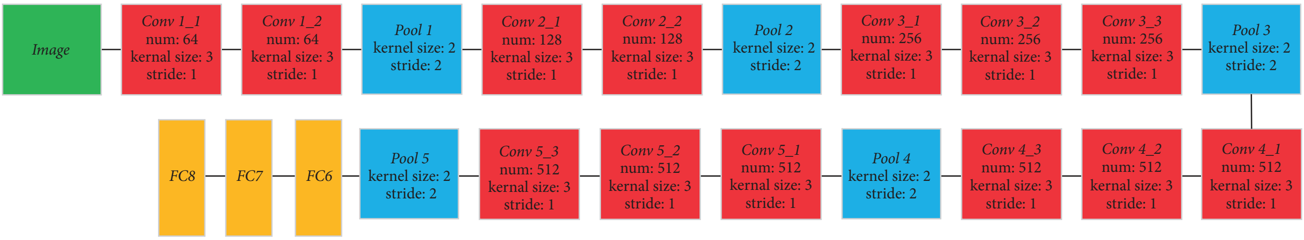


Figure 4: Fine-tuned network architecture of VGGNet [16].

available FlickrLogos-32 dataset, respectively. After that, four different deep representations are obtained for each logo image. Then, these learned deep representations are used to differentiate logo categories by training traditional classification models.

2.2. Transfer Learning by Fine-Tuning Deep CNNs. Convolutional neural networks (CNNs) [17] have been proven to be able to achieve great success in computer vision tasks, especially visual feature extraction.

Deep architectures of CNNs, called "deep convolutional neural networks (DCNNs)," have made much success in the ImageNet Large-Scale Visual Recognition Challenge (ILSVRC). There are several popular models for deep convolutional network architectures, including AlexNet [14], GoogleNet [15], VGGNet [16], and ResNet [18].

The early layers of these DCNNs are trained with a large dataset (ImageNet [19] is the common) to extract generic features. In this work, we use methods that fine-tune a pretrained model limited by the scale of logo dataset. Specifically, we use the AlexNet, GoogleNet, VGGNet, and ResNet implementation, respectively, trained on the ImageNet dataset as the pretrained models. In our transfer learning approach, as our dataset is relatively small (32,218 images) compared to ImageNet, we suppose fine-tuning the last layer of the deep models instead of the earlier layers would improve performance. To be detailed, we fine-tune the second-to-last layer of the deep models and initialize the last full connection layer to 405 outputs, corresponding to 405 categories of logo, to avoid training the model from starch for classification.

Figures 2–5 show details of four fine-tuned network architectures.

3. Experiments

3.1. Datasets. To evaluate the performance of the proposed mechanism, two datasets are adopted in the experiments, including Logo-405 and FlickrLogos-32 [20].

Logo-405 is a logo dataset crawled from Internet. It contains 405 categories of logos and 32218 logo images are included in total. To the best of our knowledge, Logo-405 is the largest logo dataset up to now. Figure 6 illustrates the logo images that are selected, one from each category.

Another benchmark dataset, named FlickrLogos-32, is a publicly available collection of logo photos. It contains 32 different logo brands by downloading them from Flickr. For each class, the dataset offers 10 training data images, 30 validation images, and 30 test images. An example of logo image of each class from FlickrLogos-32 dataset is illustrated in Figure 7.

3.2. Baseline Representation Methods. To validate the effectiveness of the proposed classification scheme, we compared the proposed method with other several baselines, including global-feature-based approach, local-feature-based method,

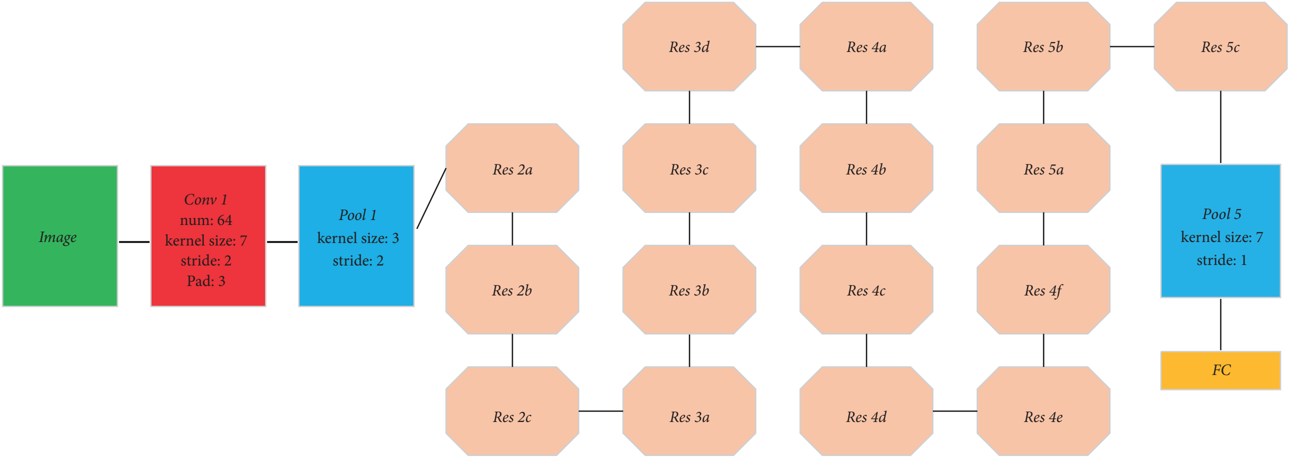


Figure 5: Fine-tuned network architecture of ResNet [18].

Figure 6: Logo image of each class from Logo-405.

and the models by fine-tuning deep CNNs. They are as follows:

(i) *Global-feature-based representation (GFBR)*: since the HSV (hue-saturation-value) space conforms to the more similarity of human perception, we adopted the quantized HSV histogram.

(ii) *Local-feature-based representation (LFBR)*: SIFT [21], as a typical local visual descriptor, has been proved to be able to capture sufficiently discriminative local elements with some invariant properties to geometric or photometric transformations and is robust to occlusion. We first perform hierarchical k-means in the training set to form a 10000-centered SIFT visual vocabulary and then adopt BOW (Bag-of-Word) technique to build the logo representation. The SIFT feature description was built followed by [1].

(iii) *Fine-tuning AlexNet representation (FTAN)*: it is a deep representation of logo image by fine-tuning AlexNet architecture. For Logo-405 dataset, the training was performed using stochastic gradient decent with image batch size of 32 images and the learning rate was reduced by hand after 54.42 K iterations from an initial setting of $1e^{-3}$, while, with respect to FickrLogos-32, the training was also performed using stochastic gradient decent with image batch size of

Figure 7: Logo images of each class from FlickrLogos-32.

32 images and the learning rate was reduced by hand after 1.89 K iterations from an initial setting of 1e^{-3}.

(iv) *Fine-tuning GoogleNet representation (FTGN)*: it is a deep representation of logo image by fine-tuning GoogleNet architecture. For Logo-405 dataset, the training was performed using stochastic gradient decent with image batch size of 32 images and the learning rate was reduced by hand after 54.42 K iterations from an initial setting of 1e^{-3}, while, with respect to FickrLogos-32, the training was also performed using stochastic gradient decent with image batch size of 32 images and the learning rate was reduced by hand after 1.89 K iterations from an initial setting of 1e^{-3}.

(v) *Fine-tuning VGG representation (FTVGG)*: it is a deep representation of logo image by fine-tuning VGG architecture. For Logo-405 dataset, the training was performed using stochastic gradient decent with image batch size of 32 images and the learning rate was reduced by hand after 54.42 K iterations from an initial setting of 1e^{-3}, while, with respect to FickrLogos-32, the training was also performed using stochastic gradient decent with image batch size of 32 images and the learning rate was reduced by hand after 1.89 K iterations from an initial setting of 1e^{-3}.

(vi) *Fine-tuning ResNet representation (FTRN)*: it is a deep representation of logo image by fine-tuning ResNet architecture. For Logo-405 dataset, the training was performed using stochastic gradient decent with image batch size of 8 images and the learning rate was reduced by hand after 217.59 K iterations from an initial setting of 1e^{-3}, while, with respect to FickrLogos-32, the training was also performed using stochastic gradient decent with image batch size of 8 images and the learning rate was reduced by hand after 7.56 K iterations from an initial setting of 1e^{-3}.

(vii) *Deep architecture in [13]*: it is a CNN network architecture specifically trained on FickrLogos-32 for logo classification.

3.3. Experiment Setup. For GFBR, considering that color is one of the most dominant and distinguishable global visual feature when describing an image, we define it in terms of a histogram in the quantized hue-saturation-value (HSV) color space with 256 components (H = 16 bins, S = 4 bins, and V = 4 bins).

With regard to LFBR, as previously described, the SIFT was extracted from each logo image and treated as local features. When carrying out LFBR in our task, all the SIFT features were quantized into 10,000 visual words using hierarchical k-means clustering technique.

With respect to the deep representations, the hyper parameter setting used in deep architecture is elaborated as in Section 3.2. Other parameters are adopted as their propositional setting value in [14–16, 18].

For classification algorithms, many classical models and their variants have been proposed, such as SVM [22, 23] and ensemble classifier [23]. In our experiments, 10-*fold* cross validation was conducted by adopting three classical classifiers, including kNN, random forest, and SVM.

Based on the experimental results of 10-*fold* cross validation, the performance of each strategy was measured by evaluating the mean average accuracy (MAA) and stand deviation (SD).

Table 1: Classification results by adopting fine-tuning deep CNN architectures.

(%)	FTAN	FTGN	FTVGG	FTRN
MAA	77.9084	82.3639	84.9192	84.8881

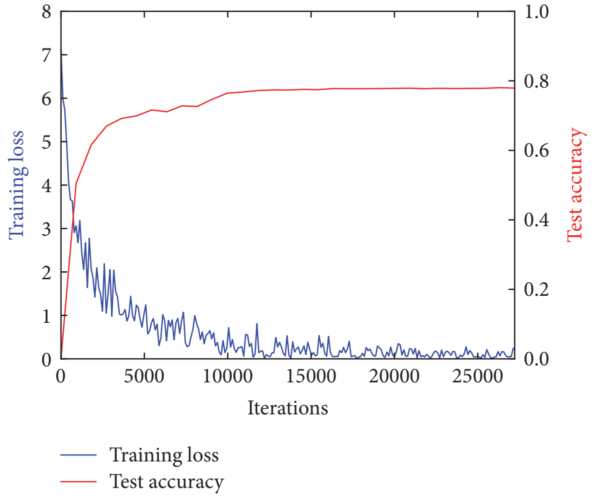


Figure 8: The training process of FTAN.

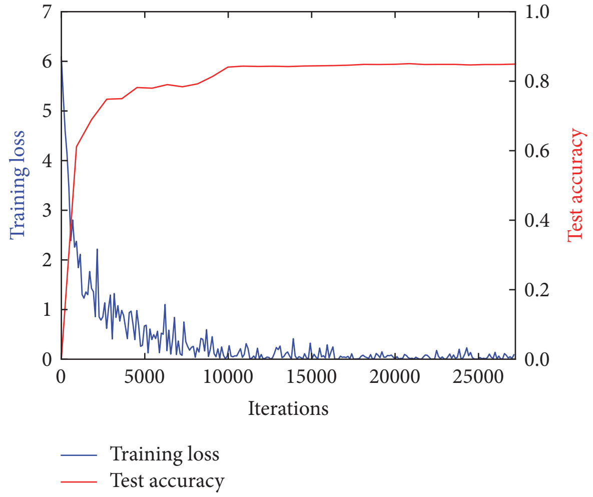


Figure 10: The training process of FTVGG.

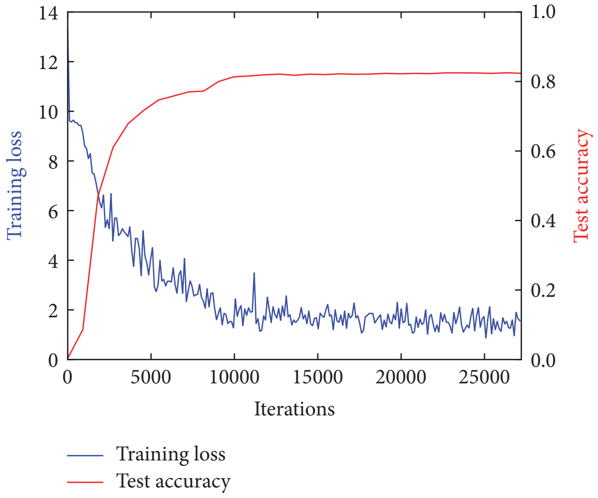


Figure 9: The training process of FTGN.

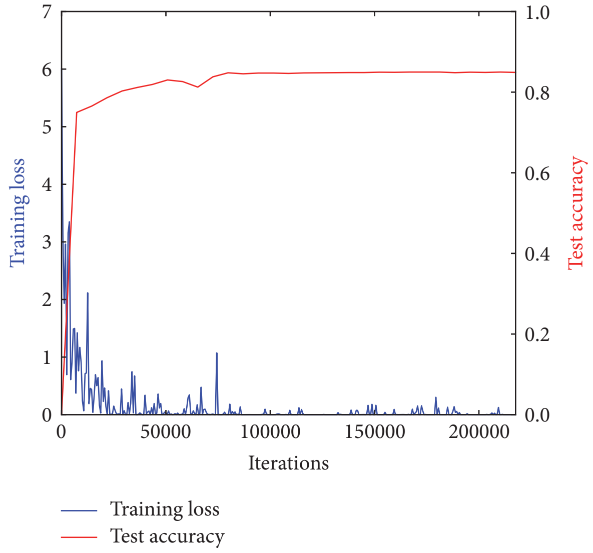


Figure 11: The training process of FTRN.

3.4. Experimental Result and Analysis on Logo-405. In this section, the results conducted on Logo-405 dataset by using three typical classifiers were reported, sequentially.

3.4.1. Results by Deep Architectures. We firstly listed the classification results by adopting fine-tuning deep architectures, as shown in Table 1.

The learning rate curves for the test accuracy and training loss of four fine-tuning CNNs were demonstrated in Figures 8–11, where the blue curve indicates the training loss rate and the red curve indicates the test accuracy.

As can be seen, in general, FTVGG achieved convergence faster than three others. In terms of test accuracy, all of them produced a dramatic increase at first, followed by a slight increase, and reached a steady state finally.

3.4.2. Classification Results by Combining Deep Representation and Traditional Classifiers. We conducted the classification tasks by combining deep representations and traditional classifiers. In this work, we adopted three typical classifiers, that is, *k*NN, *random forest,* and SVM. Since there are four deep representations obtained by fine-tuning deep CNN architectures, totally twelve different experimental combinations are produced.

Table 2: Classification comparison on SVM classifier with several strategies.

Approaches	MAA (%)	SD
GFBR	22.9685	0.8023
LFBR	65.8460	0.9976
FTAN + SVM	92.4142	0.3392
FTGN + SVM	91.0578	0.5200
FTVGG + SVM	95.9215	0.3597
FTRN + SVM	89.3848	0.5858

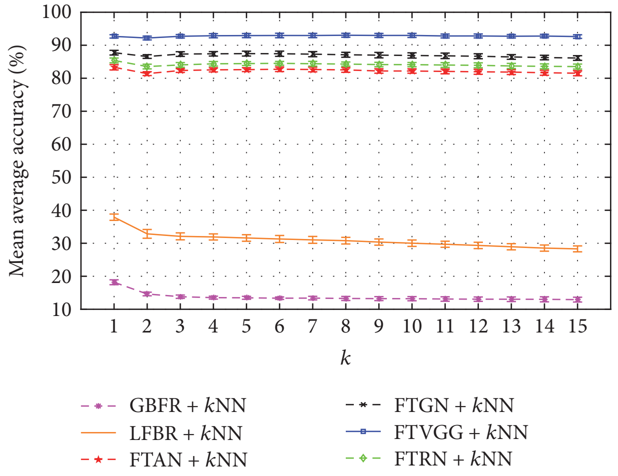


Figure 12: *k*NN classification performance comparison between different strategies.

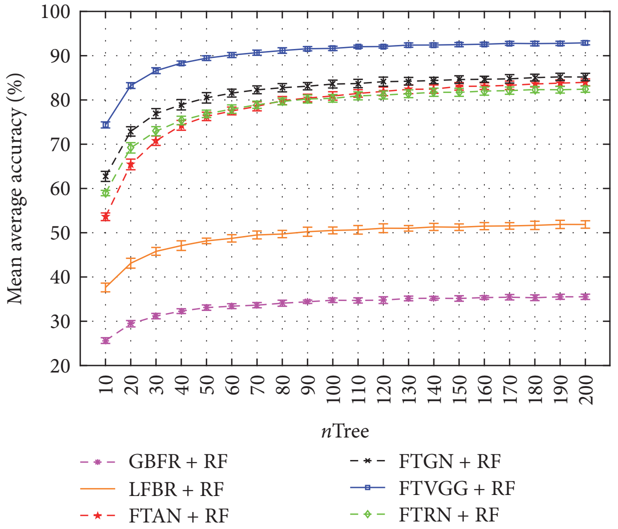


Figure 13: *Random forest* classification performance comparison between different strategies.

(1) Results by Combining Deep Representation and kNN Classifier. We conducted the *k*NN classification task with GFBR, LFBR, FTVGG + *k*NN, FTGN + *k*NN, FTAN + *k*NN, and FTRN + *k*NN in terms of 15 different values of *k* (the number of the nearest-neighbors), which differs from 1 to 15.

Figure 12 provides a graphical display of the experimental results with different representation strategies under different values of *k*. Both the MAA and SD of accuracy are illustrated in the results.

The results of Figure 12 demonstrate that (1) the approaches which combine both fine-tuning deep representation and *k*NN classier, that is, FTVGG + *k*NN, FTGN + *k*NN, FTAN + *k*NN, and FTRN + *k*NN, consistently outperform the methods that adopt hand-crafted features, including GFBR and LFBR and (2) nearly all the strategies are not sensitive to the value of *k*, especially when *k* is greater than 4.

(2) Results by Combining Deep Representation and Random Forest Classifier. This section provides experimental results conducted on a *random forest* classifier with different strategies, that is, GFBR, LFBR, FTVGG + *random forest*, FTGN + *random forest*, FTAN + *random forest,* and FTRN + *random forest*. Experiments were carried out with 20 values of *nTree* (the number of trees for random forest classifier), differing from 10 to 200.

Figure 13 provides a graphical display of the experimental results with different representation strategies under different values of *nTree*, where RF indicates *random forest* classifier. Similarly, both the MAA and SD of accuracy are illustrated in the results.

We notice that (1) with respect to all the strategies, the performance apparently tends to be better when *nTree* increases and (2) the performance of the approaches that combine fine-tuning deep representation and *random forest* classifier, that is, FTVGG + *random forest*, FTGN + *random forest*, FTAN + *random forest,* and FTRN + *random forest*, is significantly superior to LFBR and GBFR.

(3) Results by Combining Deep Representation and SVM Classifier. This section provides experimental results conducted on SVM classifier with different strategies, that is, GFBR, LFBR, FTAN + SVM, FTGN + SVM, FTVGG + SVM, and FTRN + SVM.

Table 2 lists the experimental results with different representation strategies. Both the MAA and SD of accuracy are also illustrated in the results.

Similar conclusion can be drawn from Table 3 where the performance of the approaches that combine fine-tuning deep representation and SVM classifier, that is, FTAN + SVM, FTGN + SVM, FTVGG + SVM, and FTRN + SVM, is significantly superior to LFBR and GBFR.

Lastly, we conclude this section by reporting the best performance of each strategy to compare three groups of

Table 3: The comparison between different strategies at their best performance.

FTAN	FTAN + *k*NN	FTAN + RF	FTAN + SVM
77.9084	83.3292	83.9748	92.4142
FTGN	FTGN + *k*NN	FTGN + RF	FTGN + SVM
82.3639	87.7491	85.2070	91.0578
FTVGG	FTVGG + *k*NN	FTVGG + RF	FTVGG + SVM
84.9192	92.6811	92.9046	95.9215
FTRN	FTRN + *k*NN	FTRN + RF	FTRN + SVM
85.3234	85.3964	82.4291	89.3848
	GFBR + kNN	GFBR + RF	GFBR + SVM
	18.1854	34.7383	22.9685
	LFBR + kNN	LFBR + RF	LFBR + SVM
	37.8982	51.8752	65.846

strategies, including the approaches that adopt fine-tuning deep CNNs (i.e., FTAN, FTGN, FTVGG, and FTRN), the methods which combine fine-tuning deep architectures and traditional classifiers (i.e., FTVGG + *k*NN, FTGN + *k*NN, FTAN + *k*NN, FTRN + *k*NN, FTVGG + *random forest*, FTGN + *random forest*, FTAN + *random forest*, FTRN + *random forest,* FTAN + SVM, FTGN + SVM, FTVGG + SVM, and FTRN + SVM), and those strategies employing hand-crafted features (i.e., GFBR, LFBR). The comparison results are shown in Table 3, where RF represents *random forest* classifier.

We have the observations from Table 3 that the proposed mechanisms that combine fine-tuning deep architectures and traditional classifiers demonstrate the superiority compared with other two groups of approaches, including the ones that adopt fine-tuning deep architectures and hand-crafted ones. The proposed classification mechanism specially obtains 5.4%, 6.1%, and 14.5% improvement on *k*NN, *random forest,* and SVM, respectively, towards FTAN strategy. For FTGN, it obtains 5.4%, 2.8%, and 8.7% when combining *k*NN, *random forest,* and SVM, respectively. With regard to FTVGG, it improves 7.8%, 8.0%, and 11% on *k*NN, *random forest,* and SVM, respectively. However, there is little improvement for FTRN when combining traditional classifiers. For example, FTRN + SVM improves 4.1% while FTRN + *k*NN obtains only 0.1% improvements.

With respect to the three classifiers used in the experiments, we observe that SVM outperforms *k*NN and *random forest* in nearly all tasks. Several factors may have contributed to this result. First, Logo-405 is of the high-dimensional representation, where the feature dimension of each logo is as high as 4096 in our deep representation strategies. Second, Logo-405 belongs to small sample size data compared with other large-scale datasets, for example, ImageNet [19]. Last, Logo-405 is of balanced data to some extent, in which each class consists of several tens to a hundred of logo images. We know that SVM works well for such kind of data, while *k*NN and *random forest* do not.

3.5. Experimental Result and Analysis on FlickrLogos-32. In this section, we evaluated the proposed mechanism on FlickrLogos-32 [20]. The experimental results conducted on Logo-405 dataset by using three typical classifiers are reported, sequentially.

Table 4: Classification results by adopting fine-tuning deep architectures.

(%)	FTAN	FTGN	FTVGG	FTRN
MAA	82.5893	90.1786	91.5179	92.8571

3.5.1. Results by Fine-Tuning Deep Architectures. We also firstly listed the classification results by adopting fine-tuning deep architectures, as shown in Table 4.

The learning rate curves for the test accuracy and training loss of four fine-tuning CNNs were demonstrated in Figures 14–17, where the blue curve indicates the training loss rate and the red curve indicates the test accuracy.

As can be seen from the above results that the training process on FlickrLogos-32 obtains faster convergence compared with Logo-45 probably because of its smaller size. In general, FTRN achieved convergence a litter slower than three others. In terms of test accuracy, all of them produced a dramatic increase at first, followed by small fluctuation, and reached a steady state finally.

3.5.2. Classification Results by Combining Deep Representation and Traditional Classifiers. Similarly, we conducted the classification tasks by combining deep representations and traditional classifiers. In this work, we adopted three typical classifiers, that is, *k*NN, *random forest,* and SVM. Since there are four deep representations by fine-tuning deep CNN architectures, totally twelve different experimental combinations are produced.

(1) Results by Combining Deep Representation and kNN Classifier. We conducted the *k*NN classification task with GFBR, LFBR, FTVGG + *k*NN, FTGN + *k*NN, FTAN + *k*NN, and FTRN + *k*NN in terms of 15 different values of *k* (the number of the nearest-neighbors), which differs from 1 to 15.

Figure 18 provides a graphical display of the experimental results with different representation strategies under different values of *k*. Both the MAA and SD of accuracy are illustrated in the results.

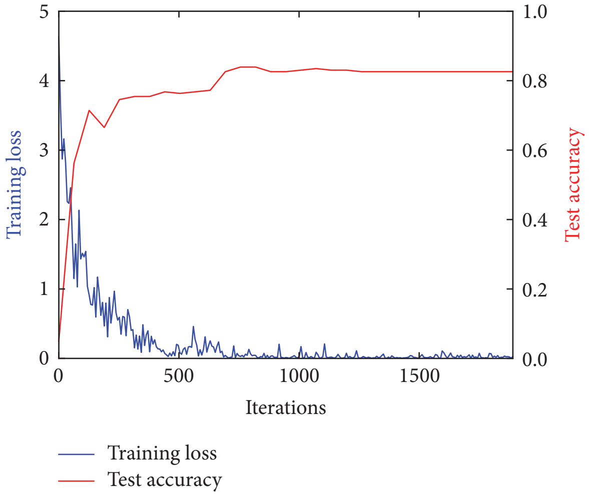


FIGURE 14: The training process of FTAN.

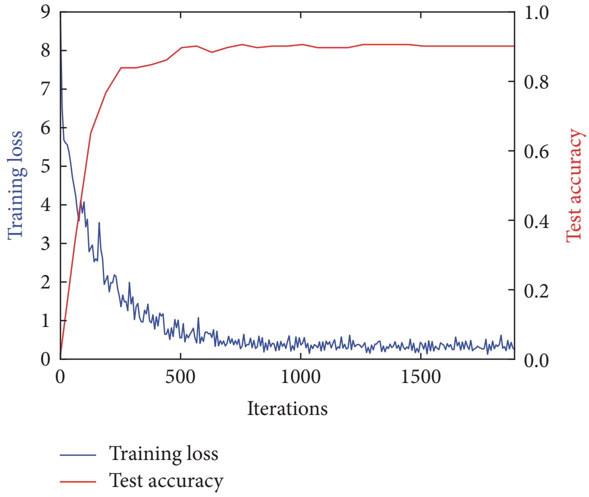


FIGURE 15: The training process of FTGN.

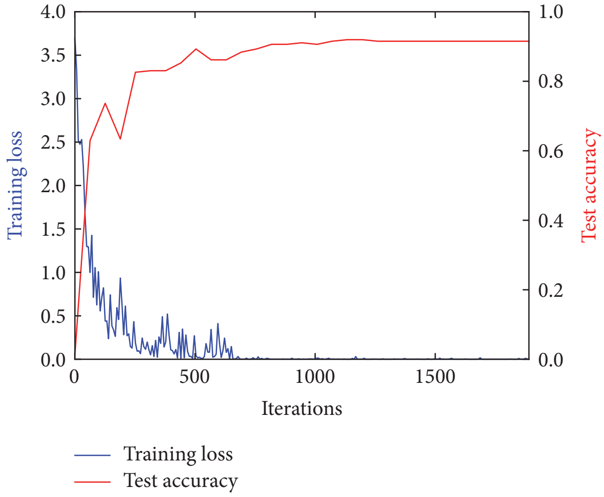


FIGURE 16: The training process of FTVGG.

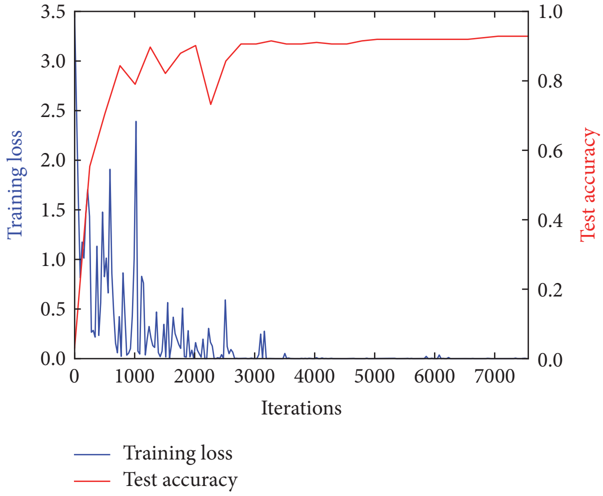


FIGURE 17: The training process of FTRN.

The results of Figure 18 demonstrate that (1) the approaches that combine both fine-tuning deep representation and kNN classier, that is, FTVGG + kNN, FTGN + kNN, FTAN + kNN, and FTRN + kNN, consistently outperform the methods that adopt hand-crafted features, like GFBR and LFBR and (2) nearly all the strategies are not sensitive to the value of k, especially when k is greater than 3.

(2) Results by Combining Deep Representation and Random Forest Classifier. This section provides experimental results conducted on a *random forest* classifier with different strategies, that is, GFBR, LFBR, FTVGG + *random forest*, FTGN + *random forest*, FTAN + *random forest*, and FTRN + *random forest*. Experiments were carried out with 20 values of *nTree* (the number of trees for random forest classifier) differing from 10 to 200.

Figure 19 gives a graphical display of the experimental results with different representation strategies under different

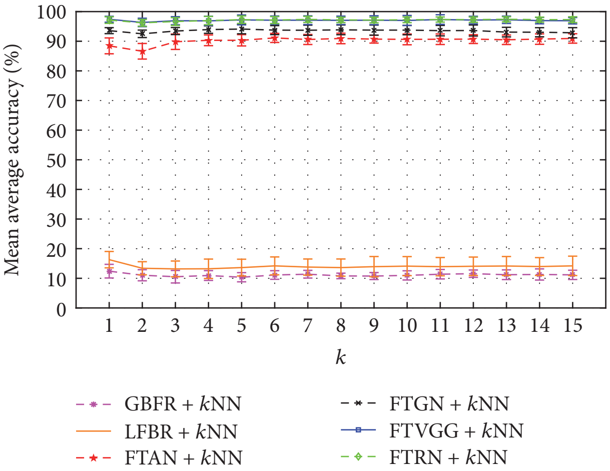


FIGURE 18: kNN classification performance comparison between different strategies.

Table 5: Classification comparison on SVM classifier with several strategies.

Approaches	MAA (%)	SD
GFBR	19.9107	2.6133
LFBR	72.1875	1.4888
FTAN + SVM	94.3304	1.3151
FTGN + SVM	94.7768	1.4125
FTVGG + SVM	98.1250	0.6243
FTRN + SVM	97.4554	1.0741

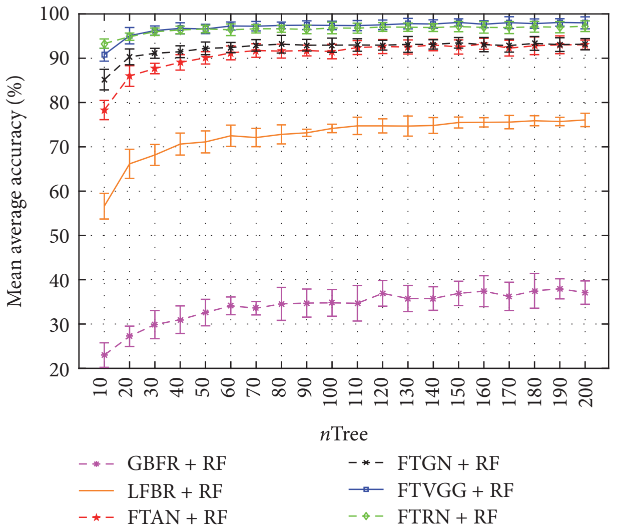


Figure 19: *Random forest* classification performance comparison between different strategies.

values of *nTree*. Similarly, both the MAA and SD of accuracy are illustrated in the results.

We find that, (1) with respect to all the strategies, the performance apparently tends to be better when *nTree* increases and (2) the performance of the approaches that combine fine-tuning deep representation and random forest classifier, that is, FTVGG + *random forest*, FTGN + *random forest*, FTAN + *random forest,* and FTRN + *random forest*, is significantly superior to LFBR and GBFR.

(3) Results by Combining Deep Representation and SVM Classifier. This section provides experimental results conducted on SVM classifier with different strategies, that is, GFBR, LFBR, FTAN + SVM, FTGN + SVM, FTVGG + SVM, and FTRN + SVM.

Table 5 provides the experimental results with different representation strategies. Both the MAA and SD of accuracy are also illustrated in the results.

Similar conclusion can be draw from Table 5 where the performance of the approaches that combine fine-tuning deep representation and *random forest* classifier, that is, FTAN + SVM, FTGN + SVM, FTVGG + SVM, and FTRN + SVM, is significantly superior to LFBR and GBFR.

Lastly, we conclude this section by reporting the best performance of each strategy to compare three groups of strategies; they are (1) the approaches that adopt fine-tuning deep architectures (i.e., FTAN, FTGN, FTVGG, FTRN, and the method proposed by Bianco et al. in [13]), (2) the methods which combine fine-tuning deep architectures and traditional classifiers (i.e., FTVGG + *k*NN, FTGN + *k*NN, FTAN + *k*NN, FTRN + *k*NN, FTVGG + *random forest*, FTGN + *random forest*, FTAN + *random forest*, FTRN + *random forest,* FTAN + SVM, FTGN + SVM, FTVGG + SVM, and FTRN + SVM), and (3) those strategies employing hand-crafted features (i.e., GFBR, LFBR). The results are shown in Table 6, where RF represents *random forest* classifier.

We have the observations from Table 6 that the proposed classification mechanisms which combine fine-tuning deep architectures and traditional classifiers demonstrate the superiority compared with other two groups of approaches, including the ones that adopt fine-tuning deep architectures and hand-crafted ones. The proposed scheme specially obtains 8.5%, 10.7%, and 11.7% improvements on *k*NN, *random forest,* and SVM, respectively, towards FTAN strategy. With respect to FTGN, it obtains 3.9%, 3.3%, amd 4.6% when combining *k*NN, *random forest,* and SVM, respectively. Regarding FTVGG, it improves 5.9%, 6.6%, and 6.6% on *k*NN, *random forest,* and SVM, respectively, while, with regard to FTRN, it can achieve 4.6%, 4.4%, and 4.6% improvements when combining *k*NN, *random forest,* and SVM, respectively. Compared to the method presented by Bianco et al. [13], the proposed mechanism obtains the improvement up to 7.125%.

4. Conclusion

With the amount of logo data on the Internet continuing to grow, designing effective management tools and systems is becoming imperative. This paper focuses on developing a fundamental tool for organizing logos by classifying them, which could make browsing and searching for logos more efficient. We design a combination mechanism that integrates both the advantages of deep learning models and traditional classification algorithms. Specifically, we firstly obtain the logo representations by fine-tuning several important deep architectures and then combine the learned logo representations with several traditional classifiers to carry out the logo classification task. While deep learning requires a large amount of data for training, we manage to achieve a high level of accuracy with a small-scale training set using transfer learning. Meanwhile, we build a Logo-405 dataset, which is larger than the existing logo datasets and can be publicly available. Experiments were conducted on both the Logo-405 dataset and FlickrLogos-32 dataset, and the results demonstrated that the proposed combination mechanism can effectively support logo classification and achieve better performance compared with other approaches, including the methods which integrate hand-crafted features and traditional pattern recognition algorithms and the models which employ deep CNNs.

Table 6: The comparison between different strategies at their best performance.

FTAN	FTAN + *k*NN	FTAN + RF	FTAN + SVM
82.5893	91.1161	93.2589	94.3304
FTGN	FTGN + *k*NN	FTGN + RF	FTGN + SVM
90.1786	94.1071	93.4821	94.7768
FTVGG	FTVGG + *k*NN	FTVGG + RF	FTVGG + SVM
91.5179	97.4107	98.0804	98.1250
FTRN	FTRN + *k*NN	FTRN + RF	FTRN + SVM
92.8571	97.4107	97.2321	97.4554
Bianco et al. [13]	GFBR + kNN	GFBR + RF	GFBR + SVM
91.0000	12.4107	35.625	19.9107
	LFBR + kNN	LFBR + RF	LFBR + SVM
	16.2946	74.1518	72.1875

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was made possible through support from the major project of Natural Science Foundation of Shandong Province (ZR2016FQ20, ZR2014FM001), Postdoctoral Science Foundation of China (2017M612338), Natural Science Foundation of China (61702313, 61572300), Taishan Scholar Program of Shandong Province in China (TSHW201502038), and Fundamental Science and Frontier Technology Research of Chongqing CSTC (cstc2015jcyjBX0124).

References

[1] A. P. Psyllos, C. N. Anagnostopoulos, and E. Kayafas, “Vehicle logo recognition using a sift-based enhanced matching scheme,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 11, no. 2, pp. 322–328, 2010.

[2] Y. Gao, F. Wang, H. Luan, and T.-S. Chua, “Brand data gathering from live social media streams,” in *Proceedings of the 2014 4th ACM International Conference on Multimedia Retrieval, ICMR 2014*, pp. 169–176, gbr, April 2014.

[3] Y. Bengio, A. Courville, and P. Vincent, “Representation learning: a review and new perspectives,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.

[4] J. Wu, S. Pan, X. Zhu, C. Zhang, and X. Wu, “Positive and Unlabeled Multi-Graph Learning,” *IEEE Transactions on Cybernetics*, vol. 47, no. 4, pp. 818–829, 2016.

[5] J. Wu, S. Pan, X. Zhu, C. Zhang, and P. Yu, “Multiple structure-view learning for graph classification,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. PP, no. 99, pp. 1–16, 2017.

[6] S. Hou, L. Chen, D. Tao, S. Zhou, W. Liu, and Y. Zheng, “Multi-layer multi-view topic model for classifying advertising video,” *Pattern Recognition*, vol. 68, pp. 66–81, 2017.

[7] J. Neumann, H. Samet, and A. Soffer, “Integration of local and global shape analysis for logo classification,” *Pattern Recognition Letters*, vol. 23, no. 12, pp. 1449–1457, 2002.

[8] S. Sun and Z. Chen, “Robust logo recognition for mobile phone applications,” *Journal of Information Science and Engineering*, vol. 27, no. 2, pp. 545–559, 2011.

[9] N. V. Kumar, V. V. Kantha, K. Govindaraju, and D. Guru, “Features fusion for classification of logos,” *Procedia Computer Science*, vol. 85, pp. 370–379, 2016.

[10] J. Wu, S. Pan, X. Zhu, C. Zhang, and X. Wu, “Multi-Instance Learning with Discriminative Bag Mapping,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 2017, pp. 1–16, 2017.

[11] N. V. Kumar, V. V. Pratheek, K. N. Govindaraju, and D. S. Guru, “Features Fusion for Classification of Logos,” *Procedia Computer Science*, vol. 85, pp. 370–379, 2016.

[12] V. Nair and G. E. Hinton, “Rectified linear units improve Restricted Boltzmann machines,” in *Proceedings of the 27th International Conference on Machine Learning (ICML ’10)*, pp. 807–814, Haifa, Israel, June 2010.

[13] S. Bianco, M. Buzzelli, D. Mazzini, and R. Schettini, “Deep learning for logo recognition,” *Neurocomputing*, vol. 245, pp. 23–30, 2017.

[14] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Proceedings of the 26th Annual Conference on Neural Information Processing Systems (NIPS ’12)*, pp. 1097–1105, Lake Tahoe, Nev, USA, December 2012.

[15] C. Szegedy, W. Liu, Y. Jia et al., “Going deeper with convolutions,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR ’15)*, pp. 1–9, Boston, Mass, USA, June 2015.

[16] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” in *Proceedings of the 3th International Conference on Learning Representations (ICIR ’15)*, pp. 1–14, May 2015.

[17] S. Haykin and B. Kosko, *GradientBased Learning Applied to Document Recognition*, Wiley-IEEE Press, 2009.

[18] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016*, pp. 770–778, usa, July 2016.

[19] J. Deng, W. Dong, and R. Socher, “ImageNet: a large-scale hierarchical image database,” in *Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 248–255, Miami, Fla, USA, June 2009.

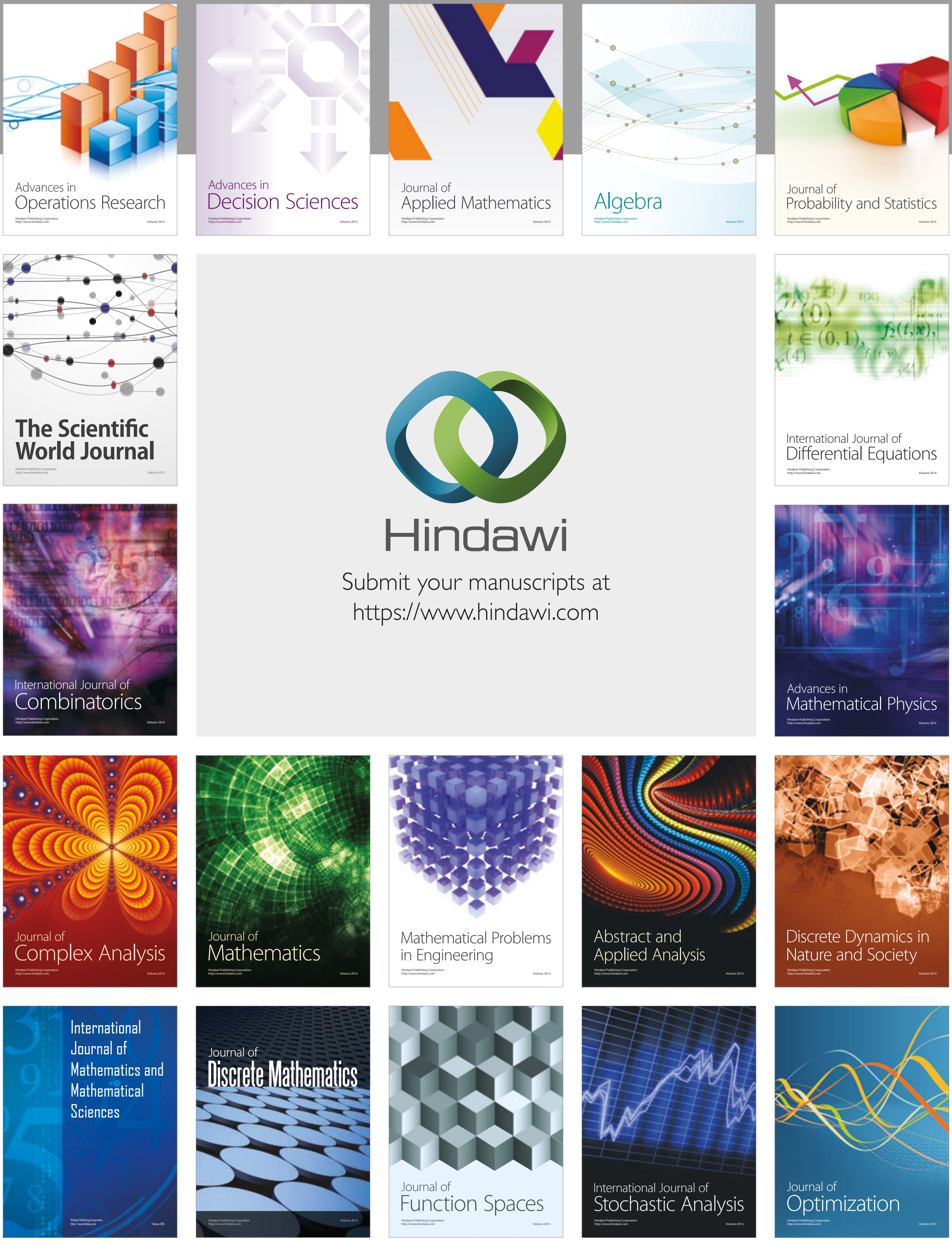
[20] S. Romberg, L. G. Pueyo, R. Lienhart, and R. Van Zwol, “Scalable logo recognition in real-world images,” in *Proceedings of*

the 1st ACM International Conference on Multimedia Retrieval, ICMR'11, Trento, Italy, April 2011.

[21] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[22] H. Zhang, L. Cao, and S. Gao, "A locality correlation preserving support vector machine," *Pattern Recognition*, vol. 47, no. 9, pp. 3168–3178, 2014.

[23] B. Gu, V. S. Sheng, and S. Li, "Bi-parameter space partition for cost-sensitive SVM," in *Proceedings of the 24th International Conference on Artificial Intelligence (ICAI '15)*, pp. 3532–3539, Las Vegas, Nev, USA, July 2015.


Advances in Operations Research
Advances in Decision Sciences
Journal of Applied Mathematics
Algebra
Journal of Probability and Statistics
The Scientific World Journal
International Journal of Differential Equations
International Journal of Combinatorics
Advances in Mathematical Physics
Journal of Complex Analysis
Journal of Mathematics
Mathematical Problems in Engineering
Abstract and Applied Analysis
Discrete Dynamics in Nature and Society
International Journal of Mathematics and Mathematical Sciences
Journal of Discrete Mathematics
Journal of Function Spaces
International Journal of Stochastic Analysis
Journal of Optimization