Received: May 21, 2022. Revised: June 20, 2022.

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

A Proposed Method to Acquire More Geometric Features from Hand-Drawn Sketches

Mahmoud M. Abdelhamied[1]* **Yasser M. Abd El-Latif[2,3]** **Fayed M. Ghaleb[2]**
Ahmed M.H. Abdelfattah[2,4]

[1] *Higher Institute of Computer Science and Information System, Cairo, Egypt*
[2] *Faculty of Science, Ain Shams University, Cairo, Egypt*
[3] *College of Computing and Information Technology, Arab Academy for Science, Technology and Maritime Transport, Cairo, Egypt*
[4] *Computer Science and Engineering, King Salman International University, South Sinai, Egypt*
* Corresponding author's Email: en_mahmoud85@yahoo.com

Abstract: We introduce a novel method for learning geometric features of hand-drawn sketches. The method is based on two models: the first depends on a newly-designed algorithm based on the Decision Tree (DT) technique; while the other is based on the Feed-forward Neural Network (FNN) technique. Each model consists of training and testing phase, we extract features of object and build the knowledge base in training phase. In testing phase, we test the learning ability to appear advantages of using DT with new objects and FNN with large data. The results of our method show desirable performance in learning. Experiments on TU-Berlin, QuickDraw, and our own dataset reveal the effectiveness of the method. We achieve learning accuracy 99.98% on our own dataset, 98.07% on TU-Berlin, and 96.69% on QuickDraw. Experiments signify that geometric feature representation and manipulation by our method brings about a substantial improvement over state-of-the-art methods on sketch classification.

Keywords: Sketch recognition, Geometric features, Decision tree, Artificial neural network, Learning.

1. Introduction

Sketching is a natural way to mundanely record, express, and share ideas. Compared with texts, hand-free sketching provides a more expressive way to show people's approximate ideas through natural drawing [1, 2].

A hand-drawn sketch is roughly defined as a rapidly executed freehand scribble that is not usually intended as a finished artwork. There is still a challenging deficiency and broadness in formally defining sketches and their underlying terms. This is one major source of trouble for human-made machines to mechanically deal with human-made sketches (e.g., automatically generating or recognizing hand-drawn sketches of objects) [3]. Sketch recognition can be defined as the task of finding groups of ink in the sketch that represent individual shapes (or objects), and then of determining the class of the object represented by each ink group ("object recognition"). Sketch recognition targets classifying human-drawn sketches into categories [2], and is a well-established field in artificial intelligence (AI) [4, 5].

The ability to learn is a distinct feature of intelligent systems, biological or otherwise. Learning in artificial systems is viewed as the process of updating the internal representation of the system in react to external stimuli so that it can do a specific task. Machine learning (ML) is one of the essential and most active research areas in the field of AI. Both AI and ML are witnessing great advances in the time being.

ML is about making computers modify or adapt their actions (whether these actions are to make predictions or to control a robot) so that the action performance gets more accurate. Accuracy is measured by how well the chosen actions reflect the

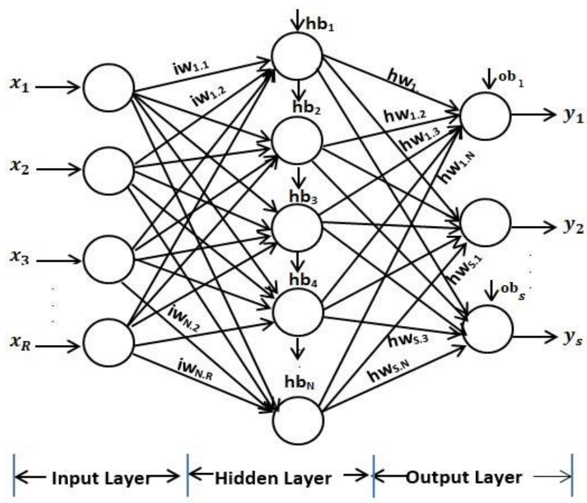

Figure. 1 A Feed-forward neural network with one hidden layer

correct ones [6]. It is only over the past decade or so that the inveterate multi-disciplinarily of machine learning has been recognized. It merges ideas from neuroscience, biology, statistics, mathematics, and physics, to endow computers with a “learning” capability.

The most common type of ‘learning algorithms’ is supervised learning. It takes a training set of examples with the correct responses (targets), then, based on the training set, generalizes an efficient response to all possible inputs different from the training set. This is also called learning from exemplars [6].

There are two popular data modelling techniques, which are ‘Decision Trees (DT)’ (also called Classification Trees) and ‘artificial neural networks (ANN)’. The two modelling techniques are very different from the way they look to the way they find relationships within variables. ANN is an assembly of nodes that looks somewhat like the human brain. A DT follows the top-down approach of looking at the data.

1.1 Decision trees

A DT is a non-parametric supervised learning method, it can be used to solve both regression and classification problems [7]. The goal is to create a model, which predicts the value of a target variable by learning simple decision rules deduced from the data features. A DT is a flowchart-like tree structure where an internal node represents a feature, the branch represents a decision rule, and each leaf node represents the result. The topmost node in a DT is known as the root node [8, 9]. Based on attribute values, a DT learns to divide the tree in a recursive manner. This flowchart-like structure helps one in decision-making. DTs are easy to understand and interpret because its conception is like a flowchart diagram that easily mimics human-level thinking.

A DT is a machine learning tool for building a tree structure from a training dataset of instances that can predict a classification (given unseen instances). DTs learn by starting at the root node and selecting the best attributes that split the training data in the best way (according to information gain) [9]. The root node, then, develops unique child nodes using an entropy function to measure the information acquired from the training data. This process continues until the tree structure can describe the given data set.

1.2 Artificial neural networks

An artificial neural network (ANN) consists of a series of algorithms that endeavour to recognize underlying relationships in a set of data through a process that mimics the way the human brain is thought to operate as a network of processing entities [10, 11]. In this meaning, an ANN refers to systems of neurons, either artificial in nature or organic. An ANN can adjust to changing input; so the network achieves the best possible result without the need to redesign the output standards. The concept of an ANN, which has its roots in artificial intelligence and neuroscience, is swiftly gaining popularity in the development of systems [10].

An ANN works in a way similar to that of the human brain’s neural network. A “neuron” in an ANN is a mathematical function that collects and classifies information according to a specific architecture [12]. The network carries a strong similarity to statistical methods such as regression analysis and curve fitting. An ANN contains layers of interconnected nodes [13]. Each node is a ‘perceptron’ and is similar to multiple linear regression. The perceptron gives the signal produced by a multiple linear regression into an activation function that can be nonlinear [11, 14].

1.2 Feed-forward neural networks

Feed-forward neural network (FNN) was the simplest type of ANN that consists of a set of processing elements called “neurons” [14, 15]. A FNN consists of a series of layers, the first of which has a connection from the inputs to the network.

Each next layer has a connection from its previous layer. The final layer produces the network’s output. The input is composed of artificial input neurons and brings the initial data into the system for further processing by subsequent layers of artificial neurons. An example of a simple FNN with a single hidden layer is shown in Fig. 1. Where x_1, x_2,

…, x_R represent inputs element of the network, $iw_{1,1}$, …., $iw_{N,R}$ are corresponding weights, hb_1, hb_2, ….,hb_N is the bias in the hidden layer, $hw_{1,1}$, ….,$hw_{s,N}$ represent the sum in the hidden layer, y_1,y_2,…,y_s represent the outputs of the network.

In this paper we propose a novel method for learning geometric features of hand-drawn sketches and is based on two models. The first model depends on a DT for recognizing and classifying objects in their categories. This model also learns the recognition of sketched object. The second model depends on FNN for training and testing, the model learns to recognize the objects and classify them into correct categories. Our method works on extracting geometric features of sketched objects and classifying the objects into their categories. The method boosts the benchmark of sketch classification. The extensive experiments on the hand-free sketch benchmark datasets, our dataset in [2], the TU-Berlin sketch dataset, and the QuickDraw dataset; show great potentials of our method.

The main contributions of this work are as follows:

- Proposing a method for learning geometric features of hand-drawn sketches that is based on two model's DT and FNN.
- The DT model recognizes and classifies objects in their categories and builds a knowledge base for learning the recognition of sketched objects.
- The FNN model learns to recognize the objects and classifies them into the correct categories.
- Using multiple samples of the input dataset (our dataset, TU-Berlin, and QuickDraw) to show the great potential of our method in learning the recognition of sketched objects.
- Explain the ability of our DT model to identify a new object as a new category in the knowledge base and the performance of our FNN model for giving desirable results with a large number of categories.
- Comparing the proposed method with state-of-the-art methods.

This paper is organized as follows: related work is presented in section 2; the proposed method is presented in section 3; the experiential results are discussed in section 4; and finally, the study is concluded in section 5.

2. Related work

The research on the representation of sketches has lasted for a relatively long time. As in the studies of images and texts, the learning of characteristic features for sketches is also a hot topic for learning sketch representation.

The majority of such works [16, 17] achieved the learning goal through classification or retrieval tasks. Traditional methods focus on hand-crafted features, such as in [18], or ensemble structured features [16]. Some recent work tried to learn a neural representation of sketches. For example, [19] is considered the first attempt to recognize hand-drawn sketches and object categories by convolutional neural network (CNN). There are two popular CNNs. One of them is ALxNet CNN [20] and the other is a modified version of LeNet CNN [21]. Both are used for experiments and results showing minor improvements over the conventional state-of-the-art. The major work on utilizing deep convolutional neural network (DCNN) for free-hand sketch recognition was Sketch-a-Net. Sketch a- Net aims to exploit the unique characteristics of sketches, including multiple levels of abstraction, which is sequential in nature. It ensembles fusion. Pre-training strategies were applied to boost recognition performance. Compared to the earlier version of Sketch-a-Net, some modifications were applied in the latest network. In the second version, the authors used stroke timing and geometry information to define a data augmentation strategy that synthesizes sketches at varying abstraction levels and deformed them to achieve a richer training set and to alleviate the problem of over-fitting to scarce sketch data. This achieved 77.95% classification accuracy on the TU-Berlin sketch dataset [22].

Due to the massive apparent gap between sketches and images, Sketch-a-Net designed a particular CNN structure for sketches, which accomplished state-of-art performance at that time, with many following works, such as [23]. On the other hand, in [17] used an auxiliary classification task to immediately solve the sketch recognition by the backbone. Apart from the above-mentioned methods, which directly utilized the pixel level information from sketch images, researchers made use of vector form representation of sketches.

The very modern effort in this area is [24], where other DCNN, similarly named SketchNet, was introduced for sketch classification. But the main purpose of [24] is to automatically learn the shared structures that exist between sketch images and real images. The authors used SoftMax as a loss function and ranked the results to make the positive pairs obtain a higher score comparing to negative ones to achieve robust representation. To construct the auxiliary repository, the real images were collected from the web which covered all the sketch categories in the TU-Berlin sketch dataset [20]. To extract the real reference images for each training sketch, first, a preliminary model was trained based on AlexNet [20] following the fine-tuning process. Afterward, the top K predicted category labels of each training sketch were extracted based on the pre-trained AlexNet model. For each training sketch, the most visually similar real images were found from the image sets of top predicted categories to construct the training pairs. Thus, the sketch with the real images which is in the same class was used to generate the positive image pair while the sketch with the real images which is in distinct classes was defined as the negative image pair. SketchNet contained three subnets: R-Net was used to extract features from the real images. S-Net was applied to the sketch images. C-Net was proposed to discover the common structures between real images and sketches. Finally, the predictions were merged to achieve the final results. The best classification accuracy, achieved in the paper on the TU-Berlin sketch benchmark, was 80.42%.

In [25] a few different CNN architectures inspired by the latest achievements on training deep neural networks (DNN) were applied to sketch classification tasks with tiny input sketches. Thus, the architecture of CNNs is simplified and can be trained in a reasonable time. These CNN architectures are utilized to recognize sketch categories of the TU-Berlin sketch dataset. The results show that working with a small size of input images not only eliminates the need for additional layers but also increases the accuracy of learning.

In [26] proposed a freehand sketch recognition scheme based on the feature-level fusion of CNNs in a transfer learning context. Sketch images usually have large-scale visual variations caused by drawing styles or viewpoints, which makes it difficult to develop generalized representations using the fixed computational mode of the convolutional kernel. Thus, Zhang in [27] employed an architecture to dynamically discover object landmarks and learn discriminative structural representations to address the problem of the fixed computational mode in the feature extraction process without extra supervision. In [28] presented an improved deformable convolutional neural network for recognizing sketches, such that the network can identify the deformation of a sketch to achieve higher accuracy on a sketch dataset. Zhang in [29] proposed a cousin network to transfer the knowledge of a network learned from natural images to a sketch network by extracting more relevant features. In [30] exploited a Hybrid CNN network composed of A-Net and S-Net to describe appearance information and shape information.

In [31] designed a mixed attention dense network for sketch classification. According to the sparse characteristics of the sketch, this network uses overlapping pooling of a large size and dense blocks are added on the top of the middle convolutional layers to achieve feature reuse. for extracting more representative local, detail information, mixed attention is applied in the dense blocks. The center loss is combined with the SoftMax cross entropy loss to improve the classification accuracy. They applied experiments on TU-Berlin dataset, they achieve accuracy 85.55% in this experiment.

In [32] proposed an algorithm based on a dual-channel convolutional neural network. The contour of the sketch is obtained by the contour extraction algorithm. the sketch and contour are used as the input image of CNN. feature fusion is carried out in the full connection layer, and the classification results are obtained by using a SoftMax classifier. Experimental results applied on TU-berlin dataset and achieve 73.24% accuracy rate of recognition.

In [33] presented a scheme for sketch recognition. It generates a discriminative features representation as a result of integrating asymmetry essential information from deep features. Five different well-known pre-trained deep convolutional neural networks are fine-tuned and utilized for feature extraction. They were used high-level deep layers of the networks to get multi-features hierarchy from sketch images. The performance of the proposed scheme is evaluated on two different sketch datasets such as TU-Berlin and Sketchy for classification and retrieval tasks. Experimental achieve sketch recognition with a rate of 72.93%.

In [34] proposed a transfer learning method for sketch-recognition in which they have used a pre-trained model for feature extraction and fine-tuned it on TU-Berlin dataset. they achieved 74% accuracy in in their experiment.

In [35] proposed a representation of sketches as multiple sparsely connected graphs. they designed a graph neural network (GNN), the multi-graph transformer (MGT), for learning representations of sketches from multiple graphs, which simultaneously capture global and local geometric stroke structures. they report extensive numerical experiments on a sketch recognition task. They applied on sketches from Google QuickDraw, they achieve 93.87% recognition accuracy.

Besides CNN models, it is essential to learn sequence models for learning how to represent sketches. Recurrent ANN [36] are the most successful sequential models during the last decades. The models based on Transformer are dominating the performance on almost all-natural language processing (NLP) tasks. particularly, bidirectional

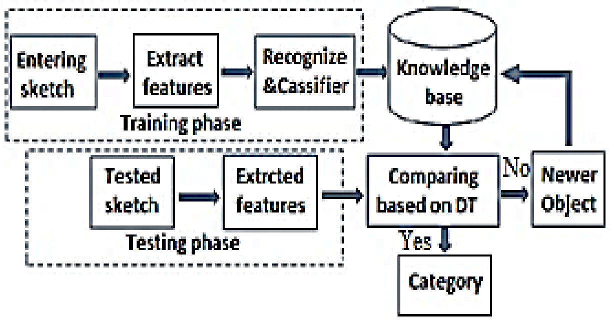

Figure. 2 The DT-based model of our method

encoder representation from transformer (BERT) [37] exploited the mask language model as a pre-training task. Such models are all trained in a self-supervised method and then fine-tuned on many finished tasks. The task of self-supervised learning [38] is generally defined as learning to predict the withheld parts of data. It thus forces the network to learn what we care about, such as image rotation [39].

Most of the earlier self-supervised learning models are specially designed for images, rather than sketches. [40] proposes a newer model of learning sketch (Sketch-BERT), which is inspired by the recent BERT model from NLP. An embedding method is tailored for sketches and encodes three level embeddings, for example point, positional, and stroke embedding. A refinement embedding network is utilized to project the embedding features into the input feature space of the transformer. The task self-supervised learning by sketch gestalt, which involves the targets of mask position prediction, and masks state prediction. It further presents in designing these tasks, the sketch gestalt model (SGM), which is inspired by the mask language model in NLP [40].

3. Methodology

The general outline of how the method works is depicted in the given diagrams of Fig. 2 and Fig. 3, which show that the method can recognize a sketch based on either a DT-model or on an FNN-model with almost no change in the general outline.

Whether based on DT or on FNN, the method consists of two phases: training and testing. The differences between the DT-based and the FNN-based models are in (1) how to build the knowledge base of each model; and in (2) how comparisons with earlier recognized sketches are done. More details are explained in section 3.1. It is worth noting here that, in an earlier work [2], a method for classifying sketched objects of a specific object category by learning a set of geometrical features of such category is presented and explained.

The method in [2] goes through three steps: (1) extracting the geometric features of sketched shapes of a specific category of objects; (2) learning the geometric features of this category; and finally (3) recognizing a newly-sketched object of the category by building a knowledge base containing the recognized objects with their geometric features.

3.1 The DT-based and FNN-based models

In this section, we explain the structure and the difference between the two models underlying the proposed method (cf. Fig. 2 and Fig. 3).

3.1.1. The DT-based model

a- Training phase

Firstly, a sketched object is given as input, we pre-process object for extracting features from it by applying some filter algorithms to enhance the object, such that if the object contains holes, or hock- let in it.

Secondly, the features of its geometric shapes are extracted (such as the area and the perimeter of each internal shape inside the object). The internal geometric shapes of the sketched object are also determined, and other features of the object, such as the ratio between the two biggest areas of internal geometric shapes that formed the object and positions between internal shapes from each other, are deduced.

Finally, the sketched object is recognized and classified in a category. The sketched object is stored in the database (knowledge base). Now we obtain the DT, which is used to recognize the object in the testing phase.

b- Testing phase:

Firstly, a test sketched object is given as input.

Secondly, the geometric and deduced features of an object are extracted (as in the training phase).

Finally, we compare the extracted features with the recorded features that are stored in the knowledge base. We apply the rules of the DT until we decide the category of the object. In the event that the decision gives us an unmatched category (that contains the new object with its features), the object is added to the knowledge base in a new category.

3.1.2. The FNN-based model

a- Training phase

A sketched object is also given as an input in its training phase. All features of the object are extracted too (as explained in the training phase of the DT

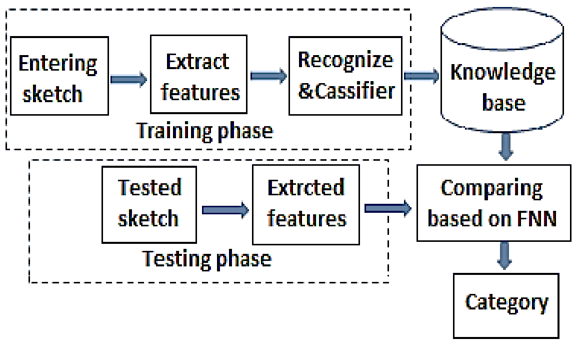

Figure. 3 The FNN-based model of our method

based model). However, the FNN-based model is based on ANN techniques and thus differs in the building process of the knowledge base. The training focuses on the process of adjusting weights and biases from the training set.

We determine the 11 inputs that represent the extracted and deduced features as inputs in the input layer. We determine targets that represent categories in the output layer. Then the FNN-based model determines the best sums of weighted for input features plus bias, which are used to reach the target. The sketched object is finally recognized and classified in its category. The sketched object is stored in the database (knowledge base).

b- Testing phase:

Firstly, a test sketched object is given as input.

Secondly, the geometric and deduced features of an object are extracted (building by these features a vector matrix that represents a vector of features).

Finally, a comparison operation works according to FNN techniques, whereas The FNN model search for the best sums weighted in the knowledge base, which is suitable for the inputs. It tries to reach the best target, which achieves less standard mean square error and is approximately equal to the value of the features vector of the test sketched object. Until we reach to the category of the object. The technicalities of the FNN model will be discussed with details in next subsection.

3.2 Further technicalities of FNN based model

Our FNN-based model uses FNN with three layers: input, hidden, and output. The input layer starts the workflow for our FNN. The input layer here contains 11 nodes, which take the object features as input. In the hidden layer, our model uses the sigmoid function as an activation function. In this network, we obtain the category of an object in the output layer. In our FNN model each neuron computes the sum of the weights of the input at the presence of a bias and passes this sum through an activation function (sigmoid function) so that the output is obtained. This process can be expressed using Eq. (1) [10, 11].

$$h_j = \sum_{i=1}^{R} iw_{j,i} X_i + hb_j \tag{1}$$

In Eq. (1), $iw_{j,i}$ is the weight of the connection between the neurons $i = (1, 2. . . R)$ and $j = (1, 2. . . N)$, hbj is a bias in the hidden layer, R is the total number of neurons in the input layer of our model, and X_i is the values of features that represent corresponding input data for the model. Here, the S-shaped curved sigmoid function [13, 16], $f(x) = \frac{1}{1+e^{-x}}$, is used as the activation function.
Therefore, the output of the neuron in the hidden layer of our network can be described as in Eq. (2) [11, 15].

$$ho_j = f_j(h_j) = \frac{1}{\left(1+e^{-h_j}\right)} \tag{2}$$

At the output layer of our model, the output of the neuron that represent category of test object in our model, is shown in Eq. (3).

$$y_k = f_k\left(\sum_{j=1}^{N} hw_{k,j}\, ho_j + ob_k\right) \tag{3}$$

Where $hw_{k,j}$ is weight connected between neurons j= (1,2, …, N) and k= (1,2, …, S), ob_k is a bias in output layer, N is the total number of neurons in the hidden layer, and S is the total number of neurons in the output layer.

In our FNN model the training phase is executed to adjust the weights and bias until some error standard is met. One of the most problem faces us is to select a suitable training algorithm. Also, it is very complex to design the ANN because many elements affect the performance of training, such as the number of neurons in the hidden layer, the interconnections among neurons and layer, the error function, and the activation function. In our FNN model, we used Trainlm as a network training function that updates weight and bias values according to Levenberg-Marquardt optimization. We depend on Trainlm because it is often one of the fastest backpropagation algorithms in the toolbox and is highly recommended as a first-choice supervised algorithm, although it does require more memory than other algorithms. Trainlm can train any network so long as its weight, net input, and transfer functions have derivative functions. In our FNN model, we

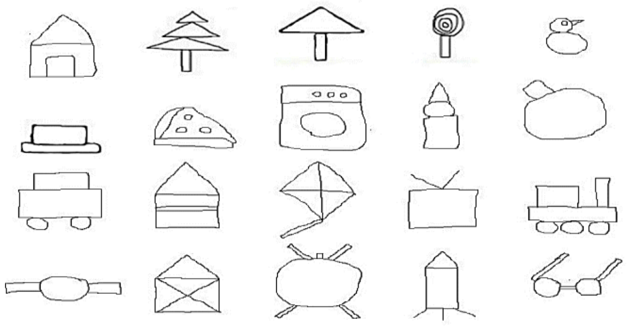
Figure. 4 A list of some recognized categories of our own dataset

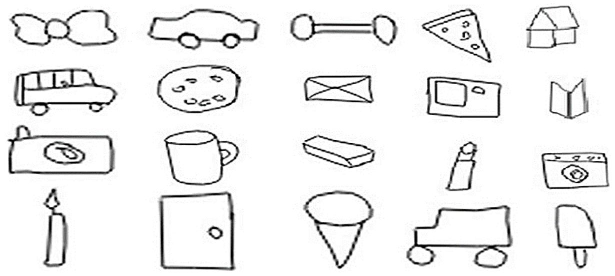
Figure. 5 A list of some recognized categories of the QuickDraw dataset

used validation vectors are utilized to prevent training early if our network performance on the validation vectors fails to improve or still the same for maximum validation failures (max fail) epochs in a row. Test vectors are used here as a further check that the network is generalizing well but do not have any effect on training.

One of the most important steps in our train FNN concerns error estimation. Researchers have achieved many ways to get calculations with short training times suitable for the network's application. Here we used the sum-of-squared errors that is the most popular error function. This is similar to utilizing the minimum least-squares optimization standard in linear regression. Similar to least squares, the sum-of-squared errors is calculated by looking at the squared differences between what our network foresees for each training test sketched object and the target value, or observed value, for that object. Formally, Eq. (4) is the same as one-half the traditional least-squares error [15]:

$$\mathrm{E} = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{C}(t_{ij} - \hat{\mathrm{t}}_{ij})^{2} \tag{4}$$

Where N is the total number of our training cases, C is equal to the number of our network outputs, t_{ij} is the observed output for the ith training case and the jth network output, and $\hat{\mathrm{t}}_{ij}$ is the network's forecast for that case.

4. Experimental results

We now present the experiments that we conducted to evaluate the accuracy of our proposed method, along with their results. We evaluate our new method based on three datasets, whose various evaluation parameters are described and analysed next. We choose some categories from data set, these were chosen based on their representation of drawings of two-dimensional facts, also containing shapes that are closer to the geometric features.

4.1 Datasets

1) Own-collected sketch samples: We work here using the 36 categories described in [2]. The volume of the dataset in [2] is expanded here so that each category now includes 10 different drawings. Fig. 4 contains a list of some recognized categories, with one sample shown from each category.

2) QuickDraw dataset: The QuickDraw dataset is publicly available from the Google application Quick, Draw!, which is an online game to draw a sketch in 20 seconds or less. There are about 50 million sketch drawings across a total of 345 categories of common objects. This dataset is collected from global players around the world, making it a rich and diverse dataset. We additionally use here the ramer-douglas-peucker (RDP) algorithm [37] to simplify the sketches. We evaluate our new method on 36 categories of the QuickDraw dataset. Fig. 5 contains a sample of the recognized categories of the QuickDraw dataset.

3) TU-Berlin dataset: The TU-Berlin dataset contains less quantity but better-quality sketch samples than that of QuickDraw. It is the first large-scale exploration of human sketches that analyses the distribution of non-expert sketches of everyday objects such as 'cup', 'chair', or 'bicycle'. There are 250 object categories in TU-Berlin with 80 sketches in each category. Here, we evaluate our new method on 36 categories of the dataset. Fig. 6 contains a sample of the recognized categories of TU-Berlin dataset.

4.2 Results

In this section, we illustrate the results of the proposed method for learning geometric features of hand drawn sketches. We show the accuracy of the two models and explain the effect of using both of them on our proposed method.

In all experiments, the number of categories is

Received: May 21, 2022. Revised: June 20, 2022.

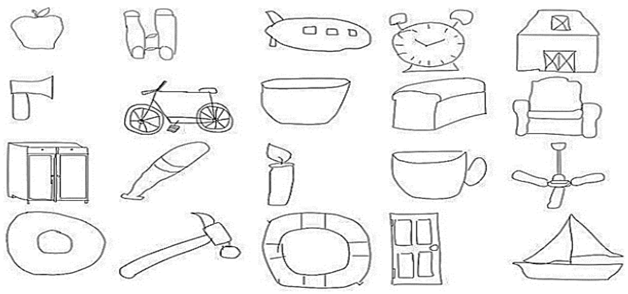
Figure. 6 A list of some recognized categories of the TU-Berlin dataset

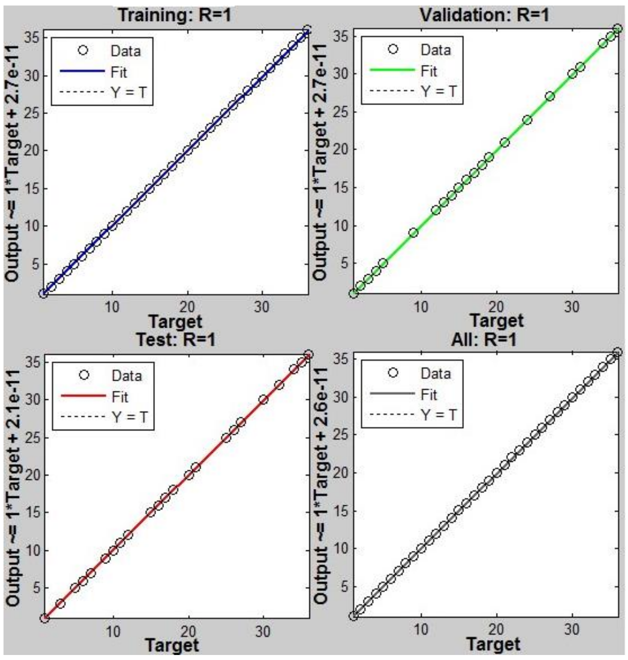

Figure. 7 NN training regression of FNN-based model with our dataset

Table 1. A collection of accuracy results by our method with regard datasets

dataset	Number of categories			
	5	10	20	36
Our dataset 5 drawing	100 %	100 %	100 %	100 %
our dataset 10 drawing	100 %	100 %	99.99 %	99.98 %
TU-Berlin	99.99 %	99.97 %	99.36 %	98.07 %
QuickDraw	99.99 %	98.27 %	97.87 %	96.69 %

increased gradually: we dealt with 5, 10, 20, and 36 categories, respectively, for all the used datasets. All the sketched objects are randomly split into 60% for training, 20% for validation, and 20% for testing.

The results showed that DT-based model is characterized by giving quick good results with a small group of data or categories. But in the case of a large amount of data DT model doesn't provide the desirable results. Experiments show that the results with a large number of categories reduce by half than those with a small number of categories. This is because the DT model needs a lot of analysis for reaching the decision needed from a large amount of data. In which, if we increase the number of categories, this led to an increase in the amount of data and makes multiple changes in feature values. Therefore, gives undesirable weakness to constantly adapt to the ever-increasing change in data. However, our DT model has the ability to identify a new class that did not exist before, the DT model analyses and registers the new object as a new category in the knowledge base.

We illustrate the results with the FNN based model. The network is implemented using FNN on the different types of datasets. It recognizes objects more easily than the DT model, FNN based model offers good capabilities for handling a large amount of data and increasing in the number of categories. From each object, 11 gradient features are extracted. Two layers feed-forward network with a stochastic gradient descent learning approach is used in ANN. The backpropagation algorithm is applied with a learning rate of 0.01, momentum is 0.9, and a number of epochs 1000 is selected.

Figs. 7, 8, and 9 show the neural network training regression of our FNN based model with the three types of datasets used here. The FNN based model achieves accuracy greater than 99.9% in own collected sketch samples dataset, greater than 98% in TU-Berlin, and greater than 96.6% in the QuickDraw dataset. We perform the experiments using the FNN model which works with the dataset described in section IV-A. We evaluate the experiments in our dataset, in an ascending sequence, where we perform experiments with 5 categories in the first time, then 10,20, and 36 categories, respectively and each category contains 5 drawings of each object. As well, we evaluate the experiments with our developed dataset that contains 10 drawings for each category with the same sequence. We evaluate experiments with 5, 10, 20, and 36 categories respectively in both the TU-Berlin and QuickDraw datasets. Table 1 illustrates the results of experiments about our FNN model with the different types of datasets used here.

Our FNN based model has the advantage of giving good desirable results with the different types of the used datasets in different cases such as if the input data isn't changing in training, testing, or the case of approximated data. Our FNN based model has a great ability to adapt with the increase in the number of categories and the increase in the volume of data.

The results indicate a slight change in the results in parallel with the increase in the number of categories. This makes our model FNN more efficient

Received: May 21, 2022. Revised: June 20, 2022.

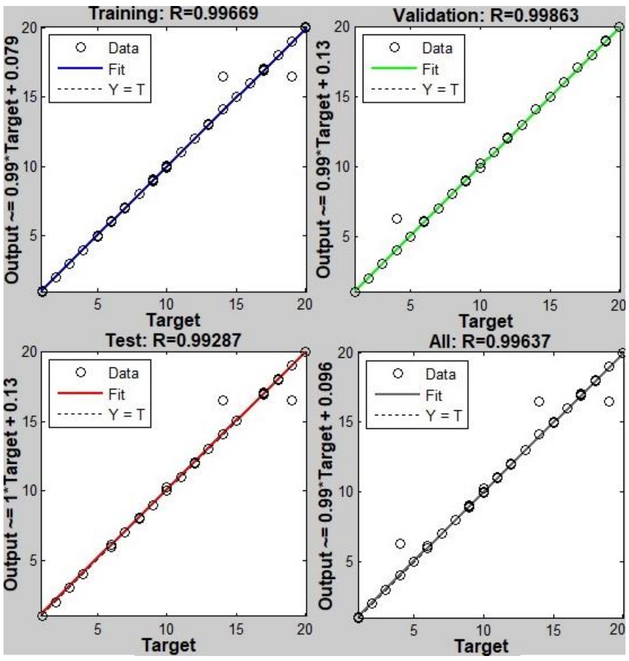

Figure. 8 NN training regression of FNN-based model with TU-Berlin dataset

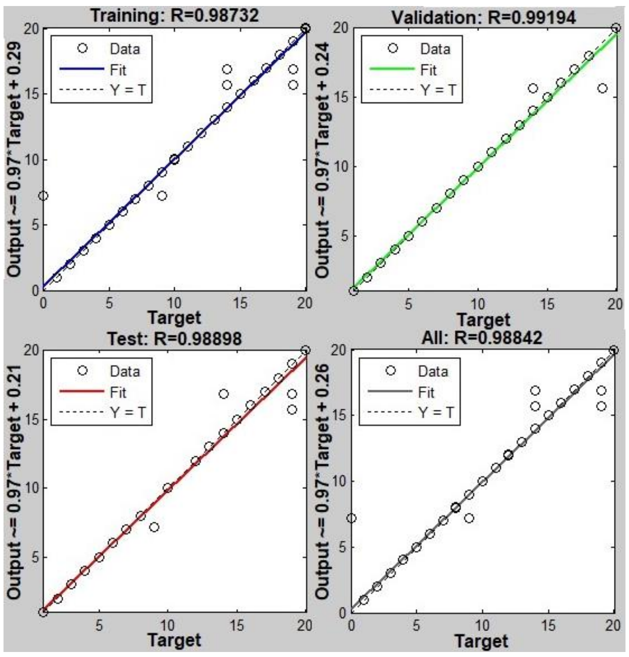

Figure. 9 NN training regression of FNN-based model with QuickDraw dataset

Table 2. A collection of accuracy results by some methods and our method with regard to the TU-Berlin and QuickDraw datasets

Method	**TU-Berlin**	**QuickDraw**
HOG [18]	56.13 %	56.0 %
Ensemble [16]	66.98 %	61.5 %
Sketch-a-Net [22]	-	77.95 %
SketchNet [24]	-	80.42 %
CIFAR-CNN [25]	-	86.42 %
TSCNN [25]	-	89.17 %
GNN [35]	-	93.87%
Transfer Learning [26]	72.50%	-
Entropy- DCNNs [33]	72.93%	-
Double-channel CNN [32]	73.24%	-
Pre-Trained Model [34]	74%	-
Deformable-CNN [28]	79.10%	-
CNG-SCN [29]	80.10%	-
Dynamic Landmarks [27]	82.95%	-
Hybrid-CNN [30]	85.07%	-
Dense-CNN [31]	85.55%	-
DSSA [23]	79.47 %	68.00 %
ResNet18 [41]	83.97 %	83.30 %
ResNet50 [41]	86.03 %	90.75 %
TCNet [17]	86.79 %	91.30 %
Sketch-BERT [40]	88.30 %	91.40 %
Our proposed FNN	**98.07 %**	**96.69 %**

in dealing with the increase in data volume effectively.

4.3 Comparative experiments

In order to verify the effectiveness of the proposed method, we compared the proposed FNN model with other different methods that we referred to in the related work section. we concentrate in our experimental to choose methods, which work with TU-Berlin dataset and QuickDraw dataset or one of them at least. we depend on this style for facilizing comparative between the state-of-the-art methods and our method.

Table 2 illustrates the results of different state-of-the-art methods and our methods with the two types of benchmark datasets (TU-Berlin and QuickDraw).

From the above experiments, our method achieves the highest accuracy with the TU-Berlin dataset 98.07%, and 96.69% with the QuickDraw dataset. According to the results, we can see that the accuracy of our method is higher and significantly more efficient than other methods. Our method is characterized by good potential with the two types of datasets in learning the recognition of sketched objects.

Our new method based on the FNN model has resulted in more high accuracy in the results compared to the mentioned methods, which used different techniques than us. Our method depends on mainly extracting geometric features for designing a learning system.

5. Conclusions

Our proposed method has a high effect on the learning process and recognizing of object sketches by extracting their geometric features. The method is based on the decision tree model and the feed-forward neural network model. Each model recognized and classified objects in their categories and built the knowledge base for learning the recognition of sketched objects. Our decision tree model is characterized by treating new data and classifying it in a new category. It gives desirable results in the case of using a small number of categories or a little amount of data, yet it showed undesired weakness when the number of categories is increased during the learning process. On the other hand, our FNN model is characterized by giving desirable results with a small number of categories, as well as by its remarkable responsiveness and efficiency when the numbers of categories are increased or large data is used.

The performance of the proposed method is comprehensively evaluated on augmented-variants of TU-Berlin sketch benchmark, QuickDraw dataset, and our own sketch datasets for sketch classification and retrieval tasks.

Our method achieved the highest accuracy with the TU-Berlin dataset 98.07%, 96.69% with the QuickDraw dataset, and 99.98% with our own sketch dataset. The experimental outcomes reveal that our method brings about a substantial improvement over the state-of-the-art methods for sketch classification and retrieval.

Conflicts of interest

The authors declare no conflict of interest.

Author contributions

The paper conceptualization, methodology, software, validation, formal analysis, investigation, resources, data curation, writing—original draft, preparation, and visualization have been done by 1st author. The writing—review and editing, supervision and project administration have been done by other authors.

References

[1] A. Abdelfattah, W. Zakaria, and N. Abdelghaffar, "Modeling of sketch-based understanding: Can computers interpret what non-artists draw?", In *Shapes* 3.0, pp. 2–6, 2015.

[2] M. Abdelhamied, Y. A. E. Latif, A. Abdelfattah, and F. Ghaleb, "A novel method for recognizing sketched objects by learning their geometrical features", *Journal of Environmental Science and Computer Science and Engineering & Technology (JECET)*, Vol. 9, No. 2, pp. 214–223, March 2020.

[3] A. Abdelfattah and W. Zakaria, "Employing a restricted set of qualitative relations in recognizing plain sketches", In: *Proc. of Joint German/ Austrian Conf. on Artificial Intelligence, Springer*, pp. 3–14, 2017.

[4] N. Abdelghaffar, A. Abdelfattah, A. Taha, and S. Khamis, "Accentuating features of description logics in high level interpretations of hand-drawn sketches", *KI-Künstliche Intelligenz*, Vol. 33, No. 3, pp. 253–265, 2019.

[5] W. Zakaria, A. Abdelfattah, N. Abdelghaffar, N. Elsaadany, N. Abdelmoneim, H. Ismail, and K. Kühnberger. "Towards the recognition of sketches by learning common qualitative representations", In: *Proc. of ProSocrates: Symposium on Problem-solving, Creativity and Spatial Reasoning in Cognitive Systems*, at Delmenhorst, Germany, July 2017.

[6] S. Marsland, *Machine learning: an algorithmic perspective*, Chapman and Hall/CRC, 2011.

[7] S. S. Shwartz and S. B. David, *18. decision trees. Understanding machine learning*, Cambridge University Press, Cambridge, 2014.

[8] B. Kami´nski, M. Jakubczyk, and P. Szufel, "A framework for sensitivity analysis of decision trees", *Central European Journal of Operations Research*, Vol. 26, No. 1, pp. 135–159, 2018.

[9] B. A. Kindhi, "Optimization of Machine Learning Algorithms for Predicting Infected COVID-19 in Isolated DNA", *International Journal of Intelligent Engineering and Systems*, Vol. 13, No. 4, pp. 423-433, 2020, doi: 10.22266/ijies2020.0831.37.

[10] R. Quiza and J. Davim, *Computational methods and optimization*, Machining of Hard Materials, Springer, pp. 177–208, 2011.

[11] Y. Yu and C. Han, "Rough Sets -Least square and Neural Networks in Fault Diagnosis Shield Applied Research", *International Journal of Intelligent Engineering and Systems*, Vol. 3, No. 3, pp. 42-49, 2010,

[12] C. Floros and P. Ballas, "Machine learning techniques and risk management: Application to the banking sector during crisis", *Machine Learning Applications for Accounting Disclosure and Fraud Detection*, IGI Global, pp. 185–200,2021.

[13] Z.Yang, and R. Yang, *Artificial Neural Network and Bioinformatics*, Comprehensive Biomedical, Elsevier, 2014.

[14] E. Kusuma, G. Shidik and R. Pramunendar, "Optimization of Neural Network using Nelder Mead in Breast Cancer Classification", *International Journal of Intelligent Engineering*

and Systems, Vol. 13, No. 6, pp. 330-337, 2020, doi: 10.22266/ijies2020.1231.29.

[15] H. Faris, I. Aljarah, and S. Mirjalili. "Training feedforward neural networks using multi-verse optimizer for binary classification problems", *Applied Intelligence*, Vol. 45, No. 2, pp. 322–332, 2016.

[16] Y. Li, Y. Song, and S. Gong, "Sketch recognition by ensemble matching of structured features", In: *BMVC*, Vol. 1, p. 2, 2013.

[17] H. Lin, Y. Fu, P. Lu, S. Gong, X. Xue, and Y. Jiang, "Tc-net for isbir: Triplet classification network for instance level sketch-based image retrieval", In: *Proc. of the 27th ACM International Conf. on Multimedia*, pp. 1676–1684, 2019.

[18] R. Hu and J. Collomosse, "A performance evaluation of gradient field hog descriptor for sketch-based image retrieval", *Computer Vision and Image Understanding*, Vol. 117, No. 7, pp. 790–806, 2013.

[19] R. Sarvadevabhatla and R. Babu, "Freehand sketch recognition using deep features", *arXiv preprint arXiv:1502.00254*, 2015.

[20] A. Krizhevsky, I. Sutskever, and G. Hinton, "Imagenet classification with deep convolutional neural networks", *Advances in Neural Information Processing Systems*, Vol. 25, pp. 1097–1105, 2012.

[21] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient based learning applied to document recognition", In: *Proc. of the IEEE*, Vol. 86, No. 11, pp. 2278–2324, 1998.

[22] M. Eitz, J. Hays, and M. Alexa, "How do humans sketch objects?", *ACM Transactions on graphics (TOG)*, Vol. 31, No. 4, pp. 1–10, 2012.

[23] J. Song, Q. Yu, Y. Song, T. Xiang, and T. Hospedales, "Deep spatial-semantic attention for fine-grained sketch-based image retrieval", In: *Proc. of the IEEE international Conf. on Computer Vision*, pp. 5551–5560, 2017.

[24] H. Zhang, S. Liu, C. Zhang, W. Ren, R. Wang, and X. Cao, "Sketchnet: Sketch classification with web images", In: *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 1105–1113, 2016.

[25] H. Atabay, "Hand drawn sketch classification using convolutional neural networks", *Gonbad Kavous University*, Iran, 2016.

[26] M. Sert and E. Boyac, "Sketch recognition using transfer learning", *Multimed Tools Appl*, Vol. 78, pp. 17095–17112, 2019.

[27] H. Zhang, P. She, Y. Liu, J. Gan, X. Cao, and H. Foroosh, "Learning structural representations via dynamic object landmarks discovery for sketch recognition and retrieval", *IEEE Trans. Image Process*, Vol.28, No.9, pp. 4486–4499, 2019.

[28] S. Yao and W. Ke, "Sketch recognition based on deformable convolutional network", *Computer Science Engineering*, Vol. 6, No. 2, pp. 2456–1843, 2020.

[29] K. Zhang, W. Luo, L. Ma and H. Li, "Cousin network guided sketch recognition via latent attribute warehouse", In: *Proc. of the AAAI Conf. on Artificial Intelligence*, pp. 9203–9210, 2019.

[30] X. Zhang, Y. Huang, Q. Zou, Y. Pei, R. Zhang, and S. Wang, "A hybrid convolutional neural network for sketch recognition", *Pattern Recognition Letters, Elsevier*, Vol. 130, pp. 73–82, 2020.

[31] M. Zhu, C. Chen, N. Wang, J. Tang, and C. Zhao, "Mixed attention dense network for sketch classification", *Applied Intelligence, Springer Science + Business Media*, LLC, part of Springer Nature, Vol. 51, pp. 7298–7305, 2021.

[32] L. Zhang, "Hand-drawn sketch recognition with a double-channel convolutional neural network", *EURASIP Journal on Advances in Signal Processing*, Springer open Nature, Vol. 73, 2021.

[33] S. Hayat, S. Kun, S. Shahzad, P. Suwansrikham, M. Mateen, and Y. Yu, "Entropy information-based heterogeneous deep selective fused features using deep convolutional neural network for sketch recognition", *IET Computer Vision Published by John Wiley & Sons Ltd on behalf of The Institution of Engineering and Technology*, Vol. 15, pp. 165–180, 2021.

[34] M. Noor, M. Nazir, S. Rehman, and J. Tariq, "Sketch-Recognition using Pre-Trained Model", In: *Proc. of National Conf. on Engineering and Computing Technology*, NUML, 2021.

[35] P. Xu, C. Joshi, and X. Bresson, "Multi-Graph Transformer for Free-Hand Sketch Recognition", *arXiv:1912.11258v3 [cs.CV]*, 2021.

[36] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modelling", a*rXiv preprint arXiv:1412.3555*, 2014.

[37] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding", *arXiv preprint arXiv:1810.04805*, 2018.

[38] A. Kolesnikov, X. Zhai, and L. Beyer, "Revisiting self-supervised visual representation learning", In: *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, pp. 1920–1929, 2019.

[39] S. Gidaris, P. Singh, and N. Komodakis, "Unsupervised representation learning by

predicting image rotations", *arXiv preprint arXiv:1803.07728*, 2018.

[40] H. Lin, Y. Fu, X. Xue, and Y. Jiang, "Sketch-Bert: Learning sketch bidirectional encoder representation from transformers by self-supervised learning of sketch gestalt", In: *Proc. of the IEEE/CVF Conf. on Computer Vision and Pattern Recognition*, pp. 6758–6767, 2020.

[41] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition", In: *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 770–778, 2016.