AI-Integrated Smart Board to Analyze Sketches to Study the Stress Level of Patients

Kiran Kumar[1], Bushra Ali Al-Khudhuri[1], Preethy Kurian[1#]and Khoula Al Harthy[1#]

[1]Middle East College, Muscat, Oman
[#]Advisor

ABSTRACT

Stable Mental health is important for a human being to lead a normal life. But many times, human beings may go through unstable mental situations due to various stress factors. Hence raising awareness on these matters and taking immediate actions when someone is going through mental orders is very important. The proposed research aims to help people with depression and can play a critical role in addressing these issues by providing depressed individuals and their family members with access to appropriate resources, support, and information. With the help of the proposed research, technical solutions can be implemented to solve society's incorrect opinions and preconceived notions about psychiatry using an application that focuses on how to best protect them from depression as early as possible aiming to solving the growing mental illness in society. The proposed research covers the feasibility of developing a smart solution which acts as a self-assessment tool for depression. The self-assessment tool can take the sketches from whiteboard and can analyze the mental state of the person using AI. CNNs are a type of Artificial Neural Network (ANN) that consists of inputs, hidden layers of nodes, and outputs. Each neuron receives inputs, computes a weighted sum, applies an activation function, and outputs a result. The hidden layers allow the neural network to perform deep learning, which is necessary for achieving high prediction accuracy. In the case of sketch recognition, the use of CNNs for sketch recognition offers advantages over existing techniques and could potentially be useful in analyzing sketches to study the stress levels of patients. However, this approach would require a large, annotated dataset and further research to adapt the CNN architecture to specifically handle the unique characteristics of patient sketches related to stress levels. The existing techniques for sketch recognition are based on Recurrent Neural Networks (RNNs), which focus on the static nature of sketches and their temporal sequential properties. However, the proposed research uses a novel approach with Convolutional Neural Networks (CNNs) that use multiple sparse graphs to capture both the stroke geometry and temporal information of sketches.

Introduction

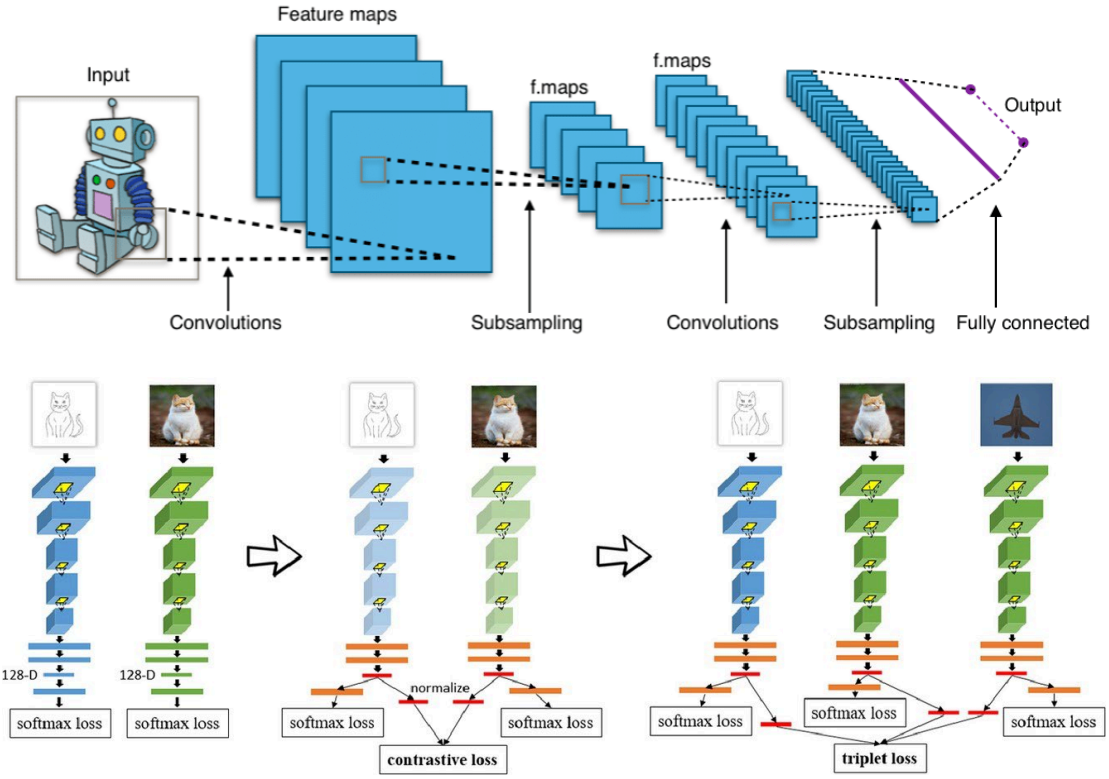
Sketches are an easy method to convey common concepts and are increasingly popular on touch-screen devices like tablets and smartphones, where gestural interaction is more comfortable. These devices are increasingly the primary platform for observing visual content today, which has sparked research into the potential of sketching as a search engine for photos and videos. Giving an input image a label or class is the work of image classification. In order to predict the class label of a brand-new, unseen image, a model is trained on a labeled dataset of images and their accompanying class labels. Convolutional neural networks (CNNs) are one of the most often used image categorization architectures. Because they can automatically learn the spatial hierarchies of features like edges, textures, and shapes that are crucial for identifying objects in images, CNNs are particularly good at classifying images.

Literature Review

The situation of sketch-based image retrieval (SBIR), which involves searching a database of photographs (images) for a certain visual notion using a hand-drawn sketched query, is discussed in this work. In the present paper, we examine SBIR from the standpoint of a cross-domain modeling issue where a low dimensional embedding is discovered between the space of sketches and pictures. Sparse feature retrieval and dictionary learning have frequently been employed to address SBIR, following their successful use for recognition and search in natural images. (Stasiak et al., 2016). Deep convolutional neural networks (CNNs) have recently been investigated for SBIR, particularly within fine-grain retrieval tasks, such as retrieving a specific shoe within an array of shoes. CNNs have been gaining acceptance as a powerful and adaptable tool for machine perception problems. (Graham et al., 2019). Despite the first, encouraging results, it is unknown how well these multi-branch networks' learned embedding adapts to different object categories. (Evers et al., 2020). For instance, allowing a user to identify visual characteristics in datasets comprising a variety of objects (such as a particular piece of furniture, a spotting dog, or a particular structure type); a problem that was more thoroughly examined in earlier work (Wu et al., 2021). SBIR, or sketch-based image retrieval, gained popularity in the early 1990s thanks to color-blob-based query algorithms developed by Flickner et al. (Wu et al., 2021) that used area close proximity visualizations to match the coarse attributes of color, shape, and texture. Following that, a number of global images descriptors were proposed using spectral patterns derived from Haar Wavelets for matching blob-based requests. (Stasiak et al., 2016).

Methodology

For published systematic examinations and original research publications, we searched the databases of Google Scholar, PubMed, Medline, ScienceDirect, and Web of Science. This evaluation only includes publications that provided adequate scientific proceedings.



An output is produced by a neural network's layers, which are made up of interconnected nodes or neurons that process input data and transfer it through the network:

The input layer, the top layer of the network, is where the network receives input data. The input layer merely receives the information and transfers it to the following layer; it does s not do any processing.

The layers that come before the output layer and before the input layer are known as the hidden layers. The majority of the network's computation takes place in these levels, including feature extraction and abstraction.
The output layer, the last layer in the network, creates the network's output.

Results

CNN's Image Classification Process

CNNs process the input data using a number of interconnected layers. A convolutional layer acts like the first hidden layer of a CNN and often applies a series of filters to the input data in order to identify particular patterns. By sliding across the input data and conducting element-wise multiplication with the filter entries, each filter creates a feature map. Following the combination of these feature maps, the model is subjected to non-linear activation functions like the ReLU function, which introduce non-linearities into the model and enable it to recognize more intricate patterns in the data. CNNs process the input data using a number of interconnected layers. A convolutional layer acts like the first hidden layer of a CNN and often applies a series of filters to the input data in order to identify particular patterns. By sliding across the input data and conducting element-wise multiplication with the filter entries, each filter creates a feature map. Following the combination of these feature maps, the model is subjected to non-linear activation functions like the ReLU function, which introduce non-linearities into the model and enable it to recognize more intricate patterns in the data. A softmax layer—typically the final layer of a CNN—generates a probability distribution over all potential class labels for the input data. The class with the highest probability is picked as the model's forecast.

First, Select A Dataset

Selecting a dataset for the picture classification problem is the first step. To train and test the CNN, a variety of publically accessible datasets including CIFAR-10, CIFAR-100, and MNIST can be employed. The CIFAR-10 dataset, which has 60,000 3232 color images divided into ten classes with 6,000 images each, will be used for this course.

```
import tensorflow as tf
from tensorflow.keras import datasets
from tensorflow.keras.models import Sequential
from tensorflow.keras import layers
```

Figure 1.

Prepare the Training Dataset

The CIFAR-10 dataset will then be loaded and prepared for training. This entails dividing the dataset into training and test sets before normalizing the picture pixel values to fall inside the 0 to 1 range.

```
# Download the data set
(train_images, train_labels), (test_images, test_labels) = datasets.cifar10.load_data()

train_images, test_images = train_images / 255.0, test_images / 255.0
```

Figure 2.

Making Training Data and Assigning Labels is Step Three

To train CNN, we will use the labeled training set of images. We will build a generator that reads the images from the directory and applies data augmentation using the flow_from_directory() function from the keras.preprocessing.image module.

By transforming the categorical class labels into one-hot encoded vectors, we will apply labels to the data. class_names = ['bird', 'cat', 'deer', 'automobile', 'airplane',

```
class_names = ['airplane', 'automobile', 'bird', 'cat', 'deer',
               'dog', 'frog', 'horse', 'ship', 'truck']
```

Figure 3.

Define and Train the CNN Model in Step Four

Using the Keras library, we will define the CNN architecture. A fully connected layer with a softmax activation function will be the final layer in the model after numerous convolutional layers, max pooling layers, and several other layers. The fit() method will then be used to train the model.

```
model = models.Sequential()
model.add(layers.Conv2D(32, (3, 3), activation='relu', input_shape=(32, 32, 3)))
model.add(layers.MaxPooling2D((2, 2)))
model.add(layers.Conv2D(64, (3, 3), activation='relu'))
model.add(layers.MaxPooling2D((2, 2)))

model.add(layers.Conv2D(64, (3, 3), activation='relu'))
model.add(layers.Flatten())
model.add(layers.Dense(64, activation='relu'))
model.add(layers.Dense(10))

model.compile(optimizer='adam',
              loss=tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True),
              metrics=['accuracy'])
history = model.fit(train_images, train_labels, epochs=10,
                    validation_data=(test_images, test_labels))
```

Figure 4.

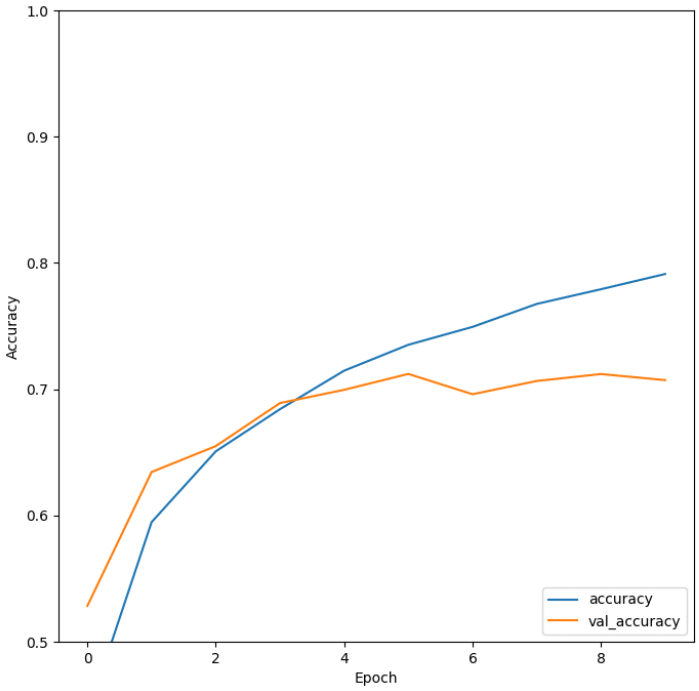
Examine the Model's Accuracy

Finally, we will use the evaluate() method to assess the trained model on the test set and determine the model's correctness.

```
plt.figure(figsize=(8, 8))
plt.plot(history.history['accuracy'], label='accuracy')
plt.plot(history.history['val_accuracy'], label = 'val_accuracy')
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.ylim([0.5, 1])
plt.legend(loc='lower right')

plt.savefig("output_report.png")
```

Figure 5.



Discussion

To find the best-performing embedding for SBIR, we examined our training procedures using all iterations of the sketch and picture architectures as well as weight-sharing schemes. We specifically assessed the network's capacity to generalize outside of the categories to which it is exposed during training. In the real world, where one cannot possibly train with a sufficiently diverse sample of potential query photos, this is crucial for SBIR. Additionally, we looked into how much sketch data was used. We discovered 13 studies that used CNNs to classify sketch-based image retrieval (SBIR). Three concepts can be used to distinguish between different classification techniques. The most popular techniques are those that use a CNN that has previously been trained using a large dataset and then tune its

parameters for the classification of sketch-based image retrieval (SBIR), as they perform the best with the restricted datasets that are currently available.

Conclusion

Important study areas include assessing the mental state of patients with major depressive disorders and determining the coherence of objective accounts. According to this study, by capturing the symptoms of depression, the analysis of sketch-based images may be able to automate prediction of depression status. Our approach is based on a smartphone, which is simple to use, and can help with the automatic recognition of depressive conditions.

References

Bell, C. C. (2001). Diagnostic and Statistical Manual of Mental Disorders, Fourth Edition, Text Revision: DSM-IV-TR Quick Reference to the Diagnostic Criteria from DSM-IV-TR. *JAMA: The Journal of the American Medical Association*, *285*(6), 811–812. https://doi.org/10.1001/jama.285.6.811

Evers, K., Maljaars, J., Carrington, S. J., Carter, A. S., Happé, F., Steyaert, J., Leekam, S. R., & Noens, I. (2020). How well are DSM-5 diagnostic criteria for ASD represented in standardized diagnostic instruments? *European Child & Adolescent Psychiatry*, *30*(1), 75–87. https://doi.org/10.1007/s00787-020-01481-z

Graham, S., Depp, C., Lee, E. E., Nebeker, C., Tu, X., Kim, H.-C., & Jeste, D. V. (2019). Artificial Intelligence for Mental Health and Mental Illnesses: an Overview. *Current Psychiatry Reports*, *21*(11), 116. https://doi.org/10.1007/s11920-019-1094-0

Götzl, C., Hiller, S., Rauschenberg, C., Schick, A., Fechtelpeter, J., Fischer Abaigar, U., Koppe, G., Durstewitz, D., Reininghaus, U., & Krumm, S. (2022). Artificial intelligence-informed mobile mental health apps for young people: a mixed-methods approach on users' and stakeholders' perspectives. Child and Adolescent Psychiatry and Mental Health, 16(1).

Sezgin, E. (2021). Can We Use Commercial Mobile Apps Instead of Research Mobile Apps in Healthcare Research? Frontiers in Public Health, 9. https://doi.org/10.3389/fpubh.2021.685439
World Health Organization. (2020). World Health Organization. Who.int; World Health Organization. https://www.who.int/
Image Classification Using CNN: Introduction and Tutorial. (n.d.). Datagen. Retrieved May 3, 2023, from https://datagen.tech/guides/image-classification/image-classification-using-cnn/
Bui, T., Ribeiro, L., Ponti, M., & Collomosse, J. (2018). Sketching out the details: Sketch-based image retrieval using convolutional neural networks with multi-stage regression. *Computers & Graphics*, *71*, 77–87. https://doi.org/10.1016/j.cag.2017.12.006
Chaturvedi, S. S., Tembhurne, J. V., & Diwan, T. (2020). A multi-class skin Cancer classification using deep convolutional neural networks. *Multimedia Tools and Applications*. https://doi.org/10.1007/s11042-020-09388-2