The Utilization of Artificial Intelligence in Enabling the Early Detection of Brain Tumors

Shanzeh Haider[1], Odysseas Drosis[2]

[1] The American International School of Johannesburg, Midrand, South Africa
[2] Computer Science, École polytechnique fédérale de Lausanne, Lausanne, Switzerland

Student Author

Shanzeh Haider - high school

SUMMARY

Diagnosing brain tumors is challenging due to their location and varied presentations that may mimic common disorders. A cancer diagnosis can be missed even when advanced imaging is conducted due to interpretive error or an incompatible clinical history as presented. Machine learning, when applied to radiological imagery, can alert physicians earlier to the presence of tumors and improve diagnostic evaluation. This enhanced evaluation can lead to earlier detection of malignant tumors and positively improve prognosis, quality of life, and treatment. This research aims to investigate the application of machine learning to enhance diagnosis. The study developed two machine learning models, a logistic regression model, and a neural network model. We hypothesized that our methods would work sufficiently, proving the correct diagnosis rate, especially within the neural networks model, as they are more complicated in nature. Applying a dataset sourced from Kaggle into the respective algorithms showcased a promising future of machine learning applications to brain tumor diagnosis, with test accuracies in the logistic regression model high (68%) and the neural network model at a significant high (84%). The simple fact of achieving a 84% accuracy rate on new data in the neural network model represents a promising future for the early detection of brain tumors.

INTRODUCTION

An intracranial (brain) tumor is an abnormal mass of tissue made of cells that proliferate uninhibited and are not regulated by innate control mechanisms (1). Magnetic resonance imaging (MRI) is a vital tool in diagnosing tumors. MRI scanning is a non-invasive medical imaging procedure that produces precise images of the human body's organs and tissues, including the brain, using a magnetic field and radio waves produced by a computer (2). The images generated, combined with the patient's clinical history and presentation, are often used as part of the workup to diagnose a tumor.

Misdiagnosis in the primary stages of brain tumors can occur due to misleading symptoms, delayed treatment, and lack of MRI imaging. The symptoms of most brain tumors mimic other disease presentations. They can present with symptoms consistent with Alzheimer's disease, migraines, Lyme disease, meningitis, and Multiple Sclerosis (3). Therefore, a physician may not detect the tumor within the imaging scan if used or may not even order imaging given the patient's clinical presentation. This can lead to delayed diagnosis and treatment, negatively impacting survival rates.

There is an urgent need to develop and evaluate modalities that can facilitate earlier diagnosis of intracranial tumors. Artificial intelligence (AI) classifying brain tumors presents an opportunity to address this problem.

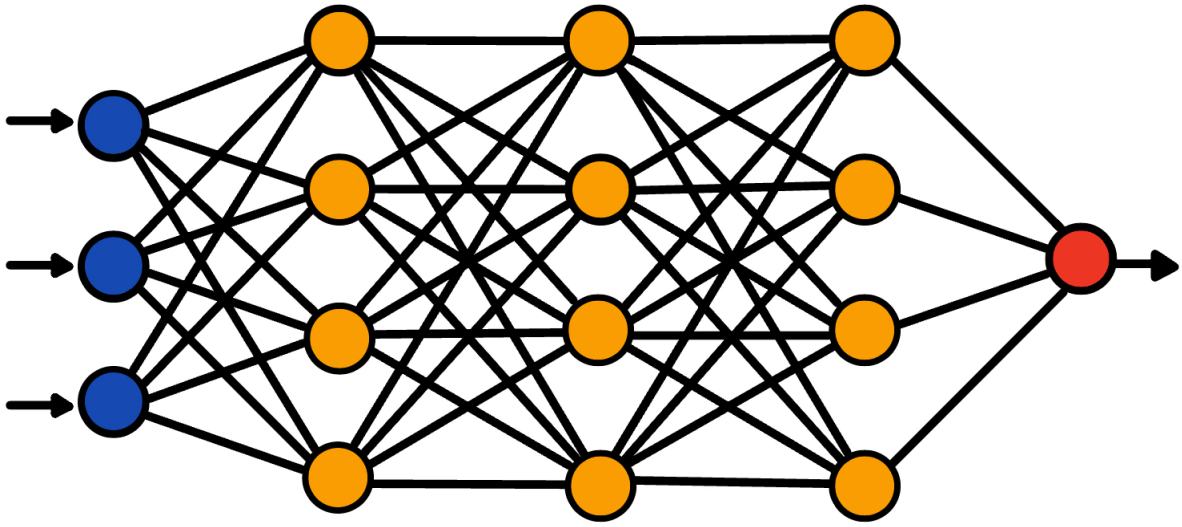

Figure 1. Neural networking and its layers. (Blue = input; orange = hidden; red = output). Created on Notability.

AI refers to using computers and machine learning to do tasks that usually call for human intelligence. By identifying patterns, AI can analyze enormous volumes of data quickly. AI algorithms can include neural networks and logistic regression models (4). Neural networks are mechanical structures with interconnected nodes that layer to mimic the function of brain neurons (Figure 1). They can cluster and categorize raw data algorithmically, identify hidden patterns and correlations, and, with time, continually learn and improve (5). Based on a given

dataset of independent factors, logistic regression calculates the likelihood that an event will occur (4). This is what allows computers to demonstrate intelligence in terms of analytical ability and deduction. Combining AI with physiological imaging can enhance the detection of characteristics that define tumors and enable more accurate and precise diagnosis and timely work up (6).

Previous studies have demonstrated the potential application of AI in the diagnosing intracranial tumors and other medical conditions. A study led by Amin Ul Haq (University of Electronic Science and Technology of China) proposed a machine learning model that utilized convolutional neural networks to improve the accuracy of brain tumor detection by AI. The model showed a high accuracy of 99.90% compared to baseline data models (7). In another recent study by the NIH, machine learning was demonstrated in diagnosing breast cancer using mammograms and neural networks. The study concluded with evidence that further research can accurately diagnose cancer using the scanning technology available, but this process can be limited by current legislations. Because of this, no quantitative data on accuracies was published (8).

All in all, the purpose of this exploration was to utilize a logistic regression and a neural network model to seek new ways to incite early brain tumor detection and expand on other research that has been done in this area.

Major results included a training accuracy of 95% and a test accuracy of 84% for the neural network model, which proved a high level of precision in the process of brain tumor detection. This is especially prominent when compared to the study led by Amin Ul Haq, which maintained a 99.90% accuracy (7). This is a promising discovery as improvements in the proposed neural network model may increase the test accuracy to a large extent. In reality, the future of brain tumor detection may be filled with highly accurate machine learning-generated models and more lives saved.

RESULTS

Model	Train Accuracy (%)	Testing Accuracy (%)
Logistic Regression	70	68
Neural Networks	95	84

Table 1: Train and test set accuracies of the logistic regression and neural network models. Data was collected from the output for each model.

Overall, the models we developed used a sensible and most accurate approach to brain tumor diagnosis. We obtained high train and test accuracies by creating a logistic regression and a neural network model. The two models we developed are logical in nature through the methods and data we collected. The procedure broadly entailed the resizing, splitting, training, and testing of the dataset. This was a very similar process for both models, yet both reacted differently to the dataset as reflected in the train and test accuracies (Table 1).

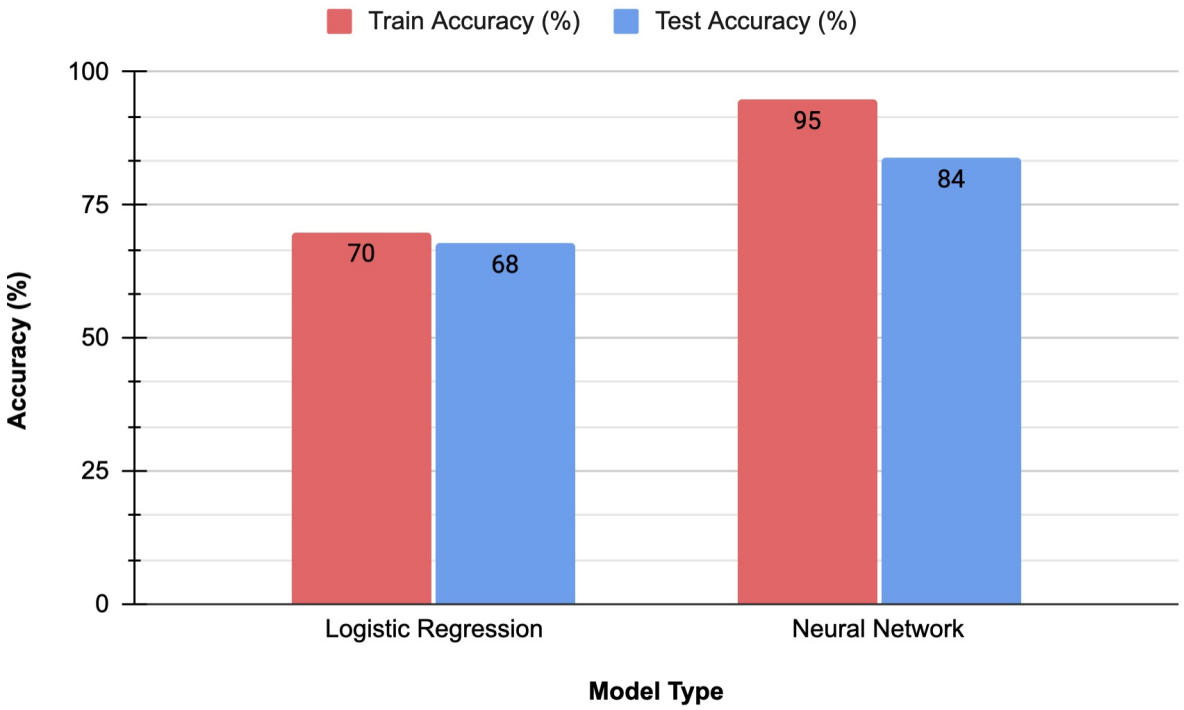

Figure 3. Accuracies (%) for logistic regression and neural network models. This data was collected through running the code for both models and reading outputted values.

Since the images in the dataset had different dimensions, we developed a low-memory algorithm that was trained fast to make predictions. We found that even with a simple model, we were able to obtain a high accuracy of 68% for the logistic regression model, and 84% on the neural network model (Table 1). The train accuracy demonstrated even more favorable results, with 70% for the logistic regression model and 95% for the neural network model. The data collected on train and test accuracies showed small differences of 2% for the logistic regression model and 9% for the neural network (Figure 3).

DISCUSSION

There were both strengths and limitations involved with the model-making we conducted, and the results of our study demonstrated a need for further model development. Experimental results consisted of training and test accuracies for the logistic regression and neural network model. For the logistic regression model, we found a training accuracy of 70% and a testing accuracy of 68%. Then, for the neural networks model, we obtained a training accuracy of 95%, and a testing accuracy of 84% (Table 1).

As expanded on in the method section, we resized the images to be of the same dimensions for both models. We did this to ensure that the models trained faster and more accurately. However, this caused a limitation within our experiment: the images being smaller reduced the quality of the images which could have decreased the performance of both the logistic regression and the neural network models (9). In addition, the size of the images we sourced from the dataset varied, and the resizing resulted in disproportionate dimensions for some images, with extremely low resolution for the already smaller images. Therefore, part of the difference between train and test set percentages for both models may have been caused by the resizing method we used. The systematic error the resizing caused was the largest problem encountered within the entire methodology.

In terms of understanding the results for each model, the quantification of the accuracy aids in the comprehension of the real world applications to this study. The accuracies serve as a percentage to recognize the precision of the model in both sets (9).

The test accuracy of 68% for the logistic regression model corresponds to predicting the presence of tumors in 68% of the images accurately when presented with new data, leaving roughly 30% of the images that may not be predicted accurately in this overall scheme. The

68% of test images being classified correctly show a considerable level of accuracy from the model, which was expected due to the simplistic nature of logistic regression models (10). The comparable training and testing accuracies demonstrate that overfitting did not occur within the logistic regression model, as the 2% difference can be accounted for due to random and systematic errors from the model itself, not the method.

Compared to the neural network model, the logistic regression model did not perform as highly on the train and test sets, but these numerical values are not the only factors of reliability from a model. In fact, the overall performance of the logistic regression model was impressive, compared to the typical performance of logistic regression models, given their simple structure (10). Therefore, when taking into consideration the reliability of data collected from machine learning models, it is imperative to view and evaluate the differences between the percentage accuracy for train and test sets in both models, along with the percentage accuracies (9). The differences show the variation in how the model reacts to new data compared to data it has seen before, and it is recommended to not obtain the exact same train and test results, but to maintain a lower difference between these values (11).

For example, the testing accuracy being lower for the neural network model may be another source of error in the code: overfitting (11). Since the neural network model performed well on the training data but not as well on the data from the test set, there is a low but plausible chance that it may have overfit the training set. This could be as a result of the model memorizing the data it has previously seen and being unable to generalize to unknown examples identified in the test set. But, this is not conclusive in terms of the model, as the train and test accuracies are relatively close in value, and further steps would need to be taken to determine this or not. Potential future experiments might involve developing neural network models which explicitly combat overfitting, for instance using a larger and more consistent dataset and utilizing regularization.

This speculation does not apply to the logistic regression model due to the fact that the 2% difference within the model's train and test accuracy did not serve as a significant difference, which is the most apparent symptom of overfitting. Overfitting occurs when the model gives accurate predictions for the train set but not the test set (4). For the neural network model, an 11% difference was calculated, which leaves open the possibility of overfitting (Figure 3). However, it is crucial to understand that overfitting is not the only explanation, but only the most

common one for larger differences in percent accuracy. The larger difference in the neural network accuracies is more likely because of the higher complexity level of the model, or the fact that deep neural networks are highly sensitive to small changes in their input (11). In fact, in the realm of neural networks, an 11% difference in training and testing accuracy is fairly small (11). The fact that the model was able to classify 84% of the test data correctly suggests that it is learning the general patterns produced by tumors, which is not suggestive of overfitting. But, the possibility remains, however slightly, because of the percentage difference.

Since the neural network model did present higher results for both accuracy measurements, it shows that the algorithm was able to have a deep understanding of the images, and overall seemingly had more parameters leading to success (11). Both models proved strengths and weaknesses, but what is most important is that both models have the capacity to grow in accuracy measurements, and by implementing further code and parameters, we may be able to obtain extremely high rates of accuracy for both models.

To further this investigation, we could perform another study to check how the model performs on images from other scanners, having a variety of types of images in the dataset. We would be interested in collaborating with radiologists who can source data for the study directly. It would be interesting to find MRI scans of normal dimensions, and therefore, we will not have to resize, or resize them proportionally. We could also extend this by researching and investigating more to see if there is a way to implement the model onto a phone, or create a device specifically for this model.

Overall the impact this research is capable of is extensive and of great potential. Quicker, and more efficient brain tumor detection can be achieved through the use of machine learning models, and our model has successfully proved itself as a support to the hypothesis that over time we can improve levels of early brain tumor detection.

MATERIALS AND METHODS

Dataset: MRI Images

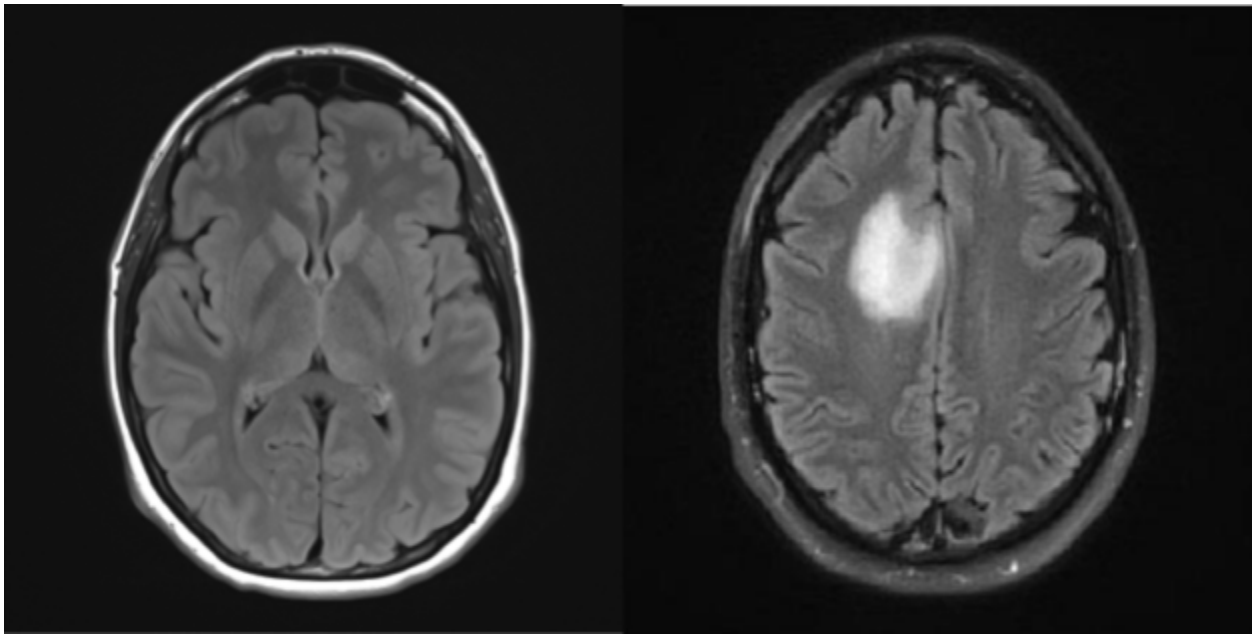

Figure 2. A sample of two MRI scans from the dataset without (left) and with (right) tumors. Scans were sourced from Kaggle and edited through Google Drawings.

The dataset used in this research project was sourced from Kaggle, an online platform with datasets that are usable for machine learning. The title of the dataset is “Brain MRI Images for Brain Tumor Detection”, last updated in 2019. The contributor who created the dataset, Navoneel Chakrabarty, holds an Engineering Doctorate at Eindhoven University of Technology and is based in the Netherlands. Chakrabarty has stated that the dataset’s images have been compiled from various sites, but has not given further information. The MRI dataset consists of 255 images of the human brain, with some containing tumors and others not (12). The dimensions of the images in the dataset varied, with some being 512 × 512 pixels or 232 × 309 pixels. Of the 255 images used 156 images were categorized as consistent with a definitive diagnosis of an intracranial tumor, and 99 images consistent with no tumor. Some images in the dataset were in higher resolution than others, and the tumor and tumorless images had varying distinctive properties (Figure 2).

Method:

To run and create code, we used the PyCharm IDE. We developed two models to predict the train and test accuracies, and understand how each model worked. The package used for both models was Scikit-learn. One model utilized logistic regression, and the other utilized neural networks. These models differ mainly through the following: complexity, non-linearity, and

interpretability (13). By using both, we were able to have a variety of understanding of the dataset and how models react to it.

For both models, the images in the dataset were resized to be of uniform dimensions of 400 x 400-pixels, which is essential to train any kind of model. This also helps as deep learning models have been known to train faster on smaller images (9).

Next, we split the data set into train and test sets to prepare it for the next step for both models. The train set is the part of the data that we worked with to create the algorithm, and the test set is what we tested the algorithm on. This split created an unbiased estimator and helped us understand how well the learned model will generalize to unseen data. The train set had a higher percentage of images from the dataset (75%) than the test set (25%) so the model had more data to learn from.

Logistic Regression Model:

Essentially, the model making process for a logistic regression model involved estimating a set of parameters that define a linear relationship between the input variables and the output variable. This consisted of a simpler process than the neural network model making, with simple coding commands. For the model using this algorithm, the pixels that compose each image have weights associated with them. The purpose of the model was to find the weight of them, and then multiply them with the pixel's respective weights. We then had the model perform a summation of as many terms as pixels. It was encoded into the algorithm that if the summation is positive it is 1 and if not it is 0. These two binary values helped determine if the image presented a tumor or not.

Neural Networks Model:

Subsequently, neural networking was employed in the second model and images in the training set were used to develop and train the AI algorithms. We decided to use neural networks because it is a more complex model, which allows for pixels to not only be separated by their weights, but also interact with each other through edges and nodes. Neural networks also consist of layers and have more parameters to be learned, which can mean that this model has increased capacity in comparison to the logistic regression model (13). We implemented the VGG16 convolutional neural network, which has 16 layers, totalling around 138 million parameters (14). Karen Simonyan and Andrew Zisserman from Oxford University proposed this

model to enhance image recognition algorithms (14). On top of the VGG16, we added another customized layer with 100 nodes. This information was used to determine if the images contained brain tumors or not. The code was continuously altered in order to improve the accuracy as a whole, and proved to work better than the logistic regression model, as shown in the results section.

ACKNOWLEDGMENTS

We would like to recognize Inspirit AI as a contributor to the success of this project. Inspirit AI connected Shanzeh and Odysseas to pursue the research presented in this paper, and provided additional support where needed. We also commend Dr. Sophia Siddiqui and Victoria Lloyd for their extensive help in suggesting modifications to this manuscript.

REFERENCES

1. "Brain Tumors - Classifications, Symptoms, Diagnosis and Treatments." American Association of Neurological Surgeons. *Aans.org*, 2023, www.aans.org/en/Patients/Neurosurgical-Conditions-and-Treatments/Brain-Tumors. Accessed 2 Mar. 2023.
2. "MRI - Mayo Clinic." Mayo Clinic. *Mayoclinic.org*, 2021, www.mayoclinic.org/tests-procedures/mri/about/pac-20384768. Accessed 2 Mar. 2023.
3. Karakus, Bengi. "Exploring Brain Tumours and Brain Tumour Misdiagnosis." *Hatchbrenner.co.uk*, 2021, hatchbrenner.co.uk/news/exploring-brain-tumours-and-brain-tumour-misdiagnosis. Accessed 4 Mar. 2023.
4. "What Is Logistic Regression?." IBM. Ibm.com, 2023, www.ibm.com/topics/logistic-regression. Accessed 10 Mar. 2023.
5. "Neural Networks: What Are They and Why Do They Matter?" SAS Analytics. *Sas.com*, 2023, www.sas.com/en_us/insights/analytics/neural-networks.html. Accessed 20 Mar. 2023.
6. Cè, Maurizio et al. “Artificial Intelligence in Brain Tumor Imaging: A Step toward Personalized Medicine.” Current oncology (Toronto, Ont.) vol. 30,3 2673-2701. 22 Feb. 2023, doi:10.3390/curroncol30030203.
7. Haq, A.u., Li, J.P., Khan, S. *et al.* DACBT: deep learning approach for classification of brain tumors using MRI data in IoT healthcare environment. *Sci Rep* 12, 15331 (2022). doi:10.1038/s41598-022-19465-1.

8. Dileep, Gayathri, and Sanjeev G Gianchandani Gyani. "Artificial Intelligence in Breast Cancer Screening and Diagnosis." *Cureus* vol. 14,10 e30318. 15 Oct. 2022, doi:10.7759/cureus.30318.
9. "Impact of Image Resizing on Deep Learning Detectors for Training Time and Model Performance." Applications in Electronics Pervading Industry, Environment and Society: APPLEPIES 2021, by Sergio Saponara and Alessandro De Gloria, e-book ed., Cham, Springer International Publishing, 2022.
10. “Evaluating a Linear Regression Model.” Ritchieng.github.io, 2016, www.ritchieng.com/machine-learning-evaluate-linear-regression-model/. Accessed 22 Mar. 2023.
11. Barkved, Kirsten. "How to Know If Your Machine Learning Model Has Good Performance | Obviously AI." *Obviously.ai*, 2022, www.obviously.ai/post/machine-learning-model-performance. Accessed 22 Mar. 2023.
12. Chakrabarty, Navoneel. “Brain MRI Images for Brain Tumor Detection.” Kaggle.com, 2019, www.kaggle.com/datasets/navoneel/brain-mri-images-for-brain-tumor-detection. Accessed 2 Jan. 2023.
13. Dreiseitl, Stephan, and Lucila Ohno-Machado. “Logistic Regression and Artificial Neural Network Classification Models: A Methodology Review.” Journal of Biomedical Informatics, vol. 35, no. 5-6, Oct. 2002, pp. 352–359, www.sciencedirect.com/science/article/pii/S1532046403000340, doi:10.1016/s1532-0464(03)00034-0. Accessed 22 Mar. 2023.
14. “Understanding VGG16: Concepts, Architecture, and Performance.” Datagen, 13 Apr. 2023, datagen.tech/guides/computer-vision/vgg16/.

Appendix (If applicable)

```
import os
import glob
from tensorflow.keras.applications.vgg16 import VGG16
from tensorflow.keras.layers import GlobalAveragePooling2D, Dense, Dropout
from tensorflow.keras.activations import sigmoid
from tensorflow.keras.utils import to_categorical
```

```
import tensorflow as tf
from sklearn.model_selection import train_test_split



data_path = '/kaggle/input/brain-mri-images-for-brain-tumor-detection/brain_tumor_dataset'
classes = ['no', 'yes']
img_size = 150
batch_size = 8
epochs = 3

# The below function reads and preprocesses the image. This is where the image is
deconstructed in terms of pixels and resized.
def parse_image(filename, label):
    img = tf.io.read_file(filename)
    img = tf.image.decode_png(img, channels=3)
    img = tf.image.resize(img, [img_size, img_size])
    img = img / 255.0
    return img, label

# The purpose of the function below is to get the file paths and labels for all images.
files ,labels= [],[]
for i, c in enumerate(classes):
    path = os.path.join(data_path, c)
    class_files = glob.glob(os.path.join(path, '*.png'))
    files.extend(class_files)
    labels.extend([i] * len(class_files))

dataset = tf.data.Dataset.from_tensor_slices((files, labels))
dataset = dataset.map(parse_image)
dataset = dataset.batch(batch_size)

# The function below split the dataset into two parts, train and test sets. Afterwards, we defined
the neural network architecture.
train_dataset, val_dataset = dataset.take(int(0.8 * len(files))), dataset.skip(int(0.8 * len(files)))
X_train, Y_train = next(iter(train_dataset))
X_val, Y_val = next(iter(val_dataset))
Y_train = to_categorical(Y_train, num_classes=len(classes))
Y_val = to_categorical(Y_val, num_classes=len(classes))

base_model = VGG16(weights='imagenet', input_shape=(img_size, img_size, 3))

# This is where the code adds a custom top layer to the VGG16 model.
x = base_model.output
```

```
x = GlobalAveragePooling2D()(x)
x = Dense(100, activation=sigmoid)(x)
x = Dropout(0.5)(x)
predictions = Dense(len(classes), activation='softmax')(x)

# This will create a new model with VGG16 base and custom top layers.
model = tf.keras.models.Model(inputs=base_model.input, outputs=predictions)

# The below function compiles and trains the model.
model.compile(optimizer=sgd, loss='categorical_crossentropy', metrics=['accuracy'])
history = model.fit(X_train, Y_train, batch_size=batch_size, epochs=epochs,
validation_data=(X_val, Y_val))

# The purpose of the below function is to evaluate the model on the validation set. This
essentially will output the validation loss and accuracy as the model found.
score = model.evaluate(X_val, Y_val)
print('Validation loss:', score[0])
print('Validation accuracy:', score[1])
```