

Building a bug bite image classification deep learning model using chatGPT.



Step: 1.1

1. Asking ChatGPT how to build a classification model

H I want to build insect bite classification model, I have a set of images with bug bite on human body.

 Sure, building an insect bite classification model is a great application of computer vision.  
Here are the steps you can follow to get started:

1. **Data collection and labeling:** Collect a large dataset of insect bite images, and label each image with the type of insect that caused the bite. This is a crucial step as the quality of your model will depend on the quality of your data.
2. **Data pre-processing:** Pre-process the images to make them ready for feeding into the model. This may involve cropping the images, resizing them, converting them to grayscale, etc.
3. **Model selection:** Choose a deep learning model architecture that's suitable for your task, such as a convolutional neural network (CNN). There are many pre-trained models that you can use as a starting point, and fine-tune on your own dataset.
4. **Training:** Train the model on your labeled dataset. You can use techniques such as data augmentation to increase the size of your dataset and reduce overfitting.
5. **Evaluation:** Evaluate the model on a held-out test set to determine its accuracy.
6. **Deployment:** Once you are satisfied with the performance of your model, you can deploy it in a production environment.

This is a high-level overview of the process. Let me know if you need more details on any of the steps, or if there's anything else I can help with!



Step Implementation Review: Here, ChatGPT gave a high-level end-to-end overview process of developing a deep learning model



Step: 2.1

2. Asking, ChatGPT to enlist various ways for data pre-processing

H I already have a large labelled dataset of good quality of insect bite, Now I need to proceed with data pre-processing, Can you give me more details of data pre-processing and how can i proceed with it ?

 Data pre-processing is an important step in preparing your data for training a deep learning model. Here are the common steps involved in data pre-processing:

1. **Resizing:** Resize your images to a standard size, so that they can be fed into your model as a fixed-size input. A common size for resizing is 256 x 256 or 224 x 224 pixels.
2. **Normalization:** Normalize the pixel values of your images to a standard range, such as [0, 1] or [-1, 1]. This is often done by dividing the pixel values by 255.
3. **Data augmentation:** Augment your data by applying random transformations to your images, such as rotation, flipping, scaling, and cropping. This helps to increase the size of your dataset and reduce overfitting.
4. **Splitting:** Split your data into training, validation, and test sets. The training set is used for training the model, the validation set is used for tuning the model hyperparameters, and the test set is used for evaluating the model.

Here's an example of how you can implement these steps in Python using the popular computer vision library, OpenCV:

Step: 3.1

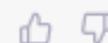
3. Asking, chatGPT to load images which are labelled in csv file for building a model



I have a csv file with file path of images and there label, I have to build a image classification model but first of all i have to load all the images



You can use the Python programming language and the following libraries to load the images:



1. Pandas: to read the CSV file containing the image file paths and labels.
2. Numpy: to convert the images to numerical arrays, which can be fed into a machine learning model.
3. OpenCV: to read and preprocess the images.

Here's a basic code example to get you started:

Step: 3.2

```
python Copy code  
  
import pandas as pd  
import numpy as np  
import cv2  
  
# Load the CSV file into a Pandas DataFrame  
df = pd.read_csv('file.csv')  
  
# Initialize an empty list to store the images  
images = []  
  
# Loop through each row in the DataFrame  
for index, row in df.iterrows():  
    # Read the image using OpenCV  
    image = cv2.imread(row['file_path'])  
  
    # Preprocess the image as needed  
    image = cv2.resize(image, (224, 224)) # resize to a standard size  
  
    # Convert the image to a Numpy array  
    image = np.array(image)  
  
    # Append the image to the list  
    images.append(image)  
  
# Convert the list of images to a Numpy array  
images = np.array(images)
```

This code will load the images from the file paths in the CSV file, resize them to a standard size, and convert them to Numpy arrays. You can then use the `images` array as input to your image classification model.



Step Implementation Review: Using the given code template was able to standardize the images and able to convert all images to array

Step: 4

4. Asking ChatGPT to Convert an array of images to a tensor using a TensorFlow

H convert images to tensor using tensorflow

To convert a list of images to a tensor using TensorFlow, you can use the following code:  

```
python Copy code  
  
import tensorflow as tf  
# Convert the list of images to a Numpy array  
images = np.array(images)  
  
# Change the data type to float  
images = images.astype(float)  
  
# Normalize the Numpy array (if desired)  
images = images / 255.0  
  
# Convert the Numpy array to a TensorFlow tensor  
images = tf.convert_to_tensor(images)
```

This code will convert the list of images to a TensorFlow tensor, change the data type to float, and normalize the values to a range of 0-1. This tensor can now be used as input to your `tf.nn.conv2d` class www.healtharkinsights.com/infographics

Read full infographic
on our website



Step: 4

4. Asking ChatGPT to Convert an array of images to a tensor using a TensorFlow



convert images to tensor using tensorflow



To convert a list of images to a tensor using TensorFlow, you can use the following code:  

python

 Copy code

```
import tensorflow as tf

# Convert the list of images to a Numpy array
images = np.array(images)

# Change the data type to float
images = images.astype('float32')

# Normalize the Numpy array (if desired)
images = images / 255.0

# Convert the Numpy array to a TensorFlow tensor
images = tf.convert_to_tensor(images)
```

This code will convert the list of images to a TensorFlow tensor, change the data type to float, and normalize the values to a range of 0-1. This tensor can now be used as input to your image classification model in TensorFlow.



Step Implementation Review: ChatGPT gave code for image conversion using tensorflow

Step: 5.1

5. Asking chatGPT for building a deep learning model for classification

H

Now i have converted an images to tensor and now i want to build a deep learning model for classification.



To build a deep learning model for image classification, you can use either PyTorch or TensorFlow. Here is a basic example of how to build a convolutional neural network (CNN) using PyTorch:



python

Copy code

```
import torch
import torch.nn as nn
import torch.nn.functional as F

# Define the CNN architecture
class CNN(nn.Module):
    def __init__(self):
        super(CNN, self).__init__()
        self.conv1 = nn.Conv2d(3, 32, kernel_size=3, stride=1, padding=1)
        self.conv2 = nn.Conv2d(32, 64, kernel_size=3, stride=1, padding=1)
        self.pool = nn.MaxPool2d(kernel_size=2, stride=2, padding=0)
        self.fc1 = nn.Linear(64 * 56 * 56, 1024)
        self.fc2 = nn.Linear(1024, 10)

    def forward(self, x):
        x = self.pool(F.relu(self.conv1(x)))
        x = self.pool(F.relu(self.conv2(x)))
        x = x.view(-1, 64 * 56 * 56)
        x = F.relu(self.fc1(x))
        x = self.fc2(x)
        return x

# Initialize the model and move it to GPU (if available)
model = CNN()
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
model = model.to(device)
```



Step: 5.2

And here is a basic example of how to build a CNN using TensorFlow:

```
python Copy code  
  
import tensorflow as tf  
  
# Define the CNN architecture  
model = tf.keras.Sequential([  
    tf.keras.layers.Conv2D(32, (3,3), padding='same', activation='relu', input_shape=(28,28,3)),  
    tf.keras.layers.MaxPooling2D(pool_size=(2,2)),  
    tf.keras.layers.Conv2D(64, (3,3), padding='same', activation='relu'),  
    tf.keras.layers.MaxPooling2D(pool_size=(2,2)),  
    tf.keras.layers.Flatten(),  
    tf.keras.layers.Dense(1024, activation='relu'),  
    tf.keras.layers.Dense(10, activation='softmax')  
])  
  
# Compile the model  
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
```

This code defines a simple CNN architecture with two convolutional layers and two fully connected layers. The model is then compiled with an Adam optimizer and categorical cross-entropy loss.

Note: These are just basic examples to get you started. You may need to modify the architecture and other hyperparameters to suit your specific problem and dataset.



Step Implementation Review: ChatGPT gave a code implementation of CNN architecture of two fully connected layers, using TensorFlow and PyTorch

Step: 6.1

6. Asking chatGPT for building a deep learning model for classification

H I want to fit my model on above data

To fit the model on your data, you'll need to split your data into training and validation sets, and then use the `fit` method in PyTorch or TensorFlow.

python

Copy code

```
# Split the data into training and validation sets
train_images = images[:8000]
val_images = images[8000:]
train_labels = labels[:8000]
val_labels = labels[8000:]

# Train the model
history = model.fit(train_images, train_labels, epochs=5, batch_size=32, validation

# Evaluate the model on the validation set
val_loss, val_acc = model.evaluate(val_images, val_labels)
print(f"Validation Loss: {val_loss}, Validation Accuracy: {val_acc}")
```

Note: These are just basic examples to get you started. You may need to modify the batch size, number of epochs, learning rate, and other hyperparameters to suit your specific problem and dataset.



Step Implementation Review: Here, the above-suggested model did not give the required accuracy due to a lack of data. Here we need to take the data augmentation approach to increase the amount of data



Step: 7.1

7. Asking, chatGPT to give code template for Data Augmentation



I would like to go for Data augmentation, i want to increase my dataset as it is very limited. 



Data augmentation is a common technique to artificially increase the size of your training data and reduce overfitting. In TensorFlow, you can use the  

`tf.keras.preprocessing.image.ImageDataGenerator` class to perform data augmentation.

Here's an example:

python

 Copy code

```
import tensorflow as tf

# Define the data generator
datagen = tf.keras.preprocessing.image.ImageDataGenerator(
    rotation_range=40,
    width_shift_range=0.2,
    height_shift_range=0.2,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True,
    fill_mode='nearest'
)
```

Step: 7.2

```
# Use the generator to create augmented data
augmented_data = datagen.flow(images, labels, batch_size=32)

# Define the model
model = tf.keras.Sequential([
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dense(128, activation='relu'),
    tf.keras.layers.Dense(1, activation='sigmoid')
])

# Compile the model
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

# Train the model with the augmented data
history = model.fit_generator(augmented_data, epochs=100, validation_data=(val_images, val_labels))

# Evaluate the model on the validation set
val_loss, val_acc = model.evaluate(val_images, val_labels)
print(f"Validation Loss: {val_loss}, Validation Accuracy: {val_acc}")
```



Step Implementation Review: was able to augment the data using the above code template. After data augmentation, we were not able to get the required accuracy.

Step: 8.1

8. Here Data Augmentation was not able to increase my model accuracy, therefore Asked for other ways to improve accuracy.



Data Augmentation does not seem to increase my model accuracy, I would like to try hyperparameter tuning of model,



Hyperparameter tuning is the process of finding the best set of hyperparameters for a machine learning model to achieve optimal performance on a given task. Here are some common ways to perform hyperparameter tuning in TensorFlow:



1. **Grid Search:** Grid search is a simple and straightforward approach to hyperparameter tuning. It involves defining a grid of hyperparameter values and training the model for each combination of hyperparameter values. The model with the best performance is then selected as the final model.
2. **Random Search:** Random search is a more efficient alternative to grid search. Instead of trying all possible combinations of hyperparameters, random search selects random combinations of hyperparameters to try.
3. **Bayesian Optimization:** Bayesian optimization is a probabilistic approach to hyperparameter tuning that uses Bayesian statistics to model the relationship between hyperparameters and the performance of the model. This approach is more computationally expensive than grid search or random search, but it often results in a more accurate set of hyperparameters.

Step: 8.2

Here's an example of how you can perform grid search in TensorFlow:

python

Copy code

```
import tensorflow as tf
import numpy as np

# Define the model
model = tf.keras.Sequential([
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dense(128, activation='relu'),
    tf.keras.layers.Dense(1, activation='sigmoid')
])

# Define a dictionary of hyperparameters to search
hyperparameters = {
    'optimizer': ['adam', 'sgd'],
    'loss': ['binary_crossentropy', 'mean_squared_error'],
    'batch_size': [32, 64, 128],
    'epochs': [10, 50, 100]
}
```

Step: 8.3

```
# Define a function to perform grid search
def grid_search(hyperparameters, images, labels):
    best_accuracy = 0.0
    best_params = None

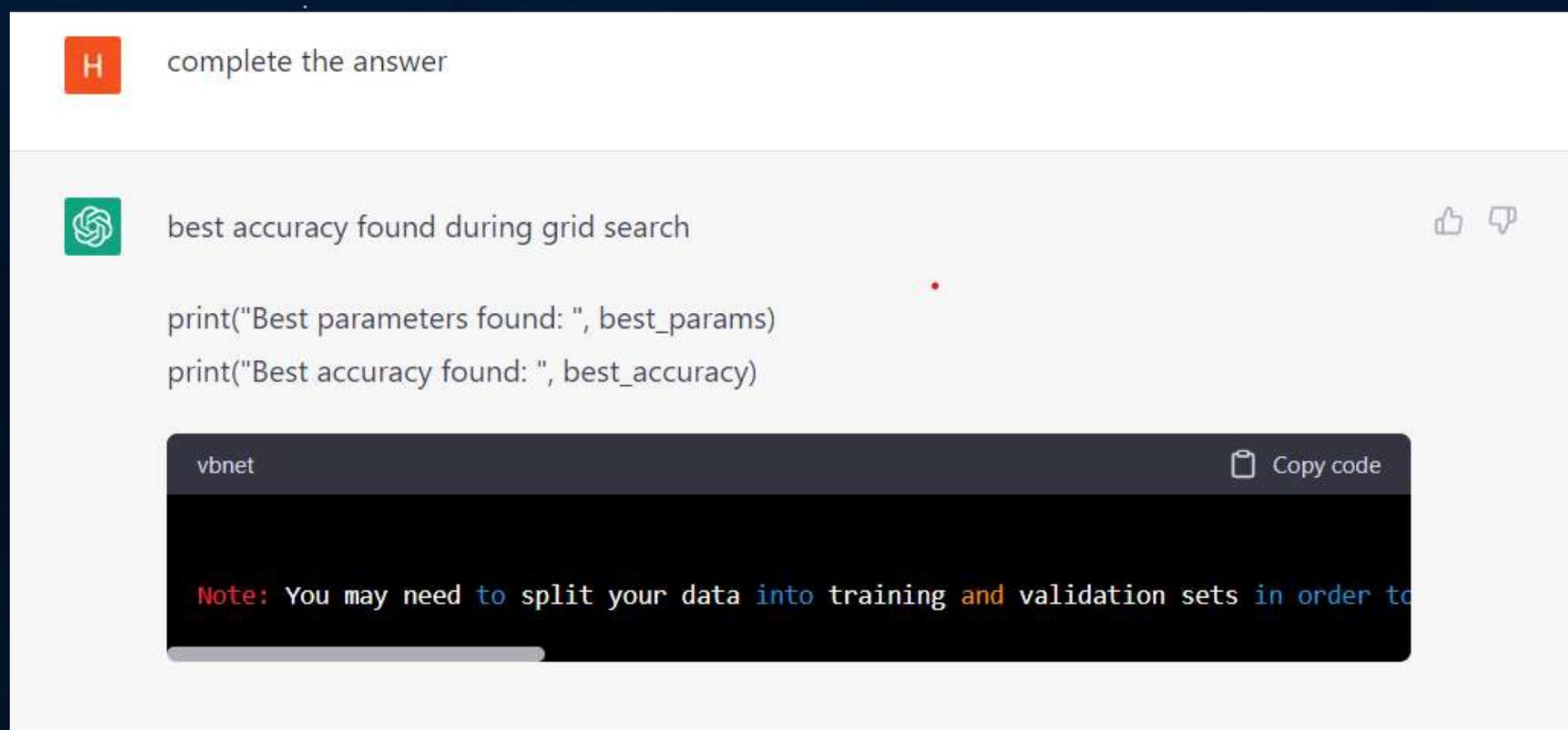
    # Loop through all combinations of hyperparameters
    for optimizer in hyperparameters['optimizer']:
        for loss in hyperparameters['loss']:
            for batch_size in hyperparameters['batch_size']:
                for epochs in hyperparameters['epochs']:

                    # Compile the model with the current hyperparameters
                    model.compile(optimizer=optimizer, loss=loss, metrics=['accuracy'])

                    # Train the model
                    history = model.fit(images, labels, batch_size=batch_size, epochs=epochs)

                    # Evaluate the model on the validation set
                    val_loss, val_acc = model.evaluate(val_images, val_labels)
```

Step: 8.4



H complete the answer

best accuracy found during grid search

```
print("Best parameters found: ", best_params)
print("Best accuracy found: ", best_accuracy)
```

vbnet Copy code

Note: You may need to split your data into training and validation sets in order to



Step Implementation Review: Fine-tuning of the model took hours to get fine-tuned, and we got the best parameters and best accuracy using GridSearchCV. But the model accuracy was not up to the mark.

Conclusion:

On the Computer Vision task, we can use chatGPT for having a general-purpose idea. chatGPT lacks mathematical intuition, and understanding of data Which chatGPT accepts too. It can be used to get a direction in solving a problem but the on-field application requires lots of experiments and scalable machines.

ChatGPT efficiency in binary image classification:



Efficient

- Clearly defining steps and explaining those steps.
- Generating a clear process and providing all the parameters necessary for specific requirements.
- Generated valid codes which can be implemented.
- It was able to give generate codes using multiple library.



Partially Efficient

- Was giving answers in a general sense. And a basic level of implementation was generated.
- Was unable to generate multiple ways for getting a solution.
- Certain generated codes were not up to date.
- Lacked mathematical intuition, which is necessary for developing a deep learning model.

Authors



Ani Kannal
Head of Data Science and Analytics
Healthark Insights



Harshit Panchal
Data Science Consultant
Healthark Insights

CONTACT US
821, Sun Avenue One
Manekbag, Ambawadi,
Amedabad, India
contact@healtharkinsights.com
healtharkinsights.com