Food Recognition using Transfer Learning

Ankit Basrur
Department of Information Technology
Dwarkadas J. Sanghvi College of Engineering
Mumbai, India
ankit.basrur@djsce.edu.in

Dhrumil Mehta
Department of Information Technology
Dwarkadas J. Sanghvi College of Engineering
Mumbai, India
dhrumil.mehta@djsce.edu.in

Abhijit R. Joshi
Department of Information Technology
Dwarkadas J. Sanghvi College of Engineering
Mumbai, India
abhijit.joshi@djsce.ac.in

***Abstract*— This paper proposes the application of Transfer Learning in classifying a food dish. Traditional methods involve using Artificial Neural Networks (ANN) and Convolutional Neural Networks (CNN), which are highly inefficient when the classes in a dataset increase. Therefore, more modern ways of classification become vital to adapt to evolving human tastes. Thus, we have achieved excellent results by leveraging Neural Networks in the form of ResNet, VGG19, EfficientNet, and DenseNet. Additionally, a web crawler has been integrated to provide the recipe for the same dish.**

Keywords*— *Neural Networks, Image Processing, Bag-Of-Words, Deep Learning, Web Scraping

I. Introduction

The diversity in Indian food is as extensive as the country itself. India's climate, soil, and access to natural resources have contributed to its food multiplicity. With great diversity becomes more challenging for one to know the names and recipes of all cuisines. Thus, we have built a system that recognizes and suggests a recipe by an image's input. Moreover, the increasing growth of mobile camera technology and the proliferation of the internet makes our model even more suitable. The developed system has the ability to serve visually impaired individuals through automatic food recognition and provide recipe suggestions via images.

Advanced Neural Network Algorithms were leveraged in constructing our model. Neural Network is a branch in Machine Learning and is represented by a weighted graph in the form of layers [1]. Each Neural Network has an input layer followed by multiple hidden layers and then the final output layer. Neural networks are trained and fine-tuned to improve their accuracy. However, we can use them once trained to classify and cluster datasets at a very high velocity.

We have utilized OpenCV and PIL to resize and convert the dataset into RGBA format for pre-processing. We have trained and analyzed the performance of multiple classification algorithms - DenseNet, VGG19, ResNet, and EfficientNet on the processed dataset. For Web Crawler, we have used BeautifulSoup, which can parse HTML pages and extract relevant information.

II. Related Work

Many past literature studies, like those of [2], assume that food items' texture, shape, and color are well-defined. This might not always be true, as there can be local variations of the same dish in the ingredients used, and the preparation method could be different.

The most frequently used techniques [3] [4] were based on age-old classification algorithms like SVM, CNN, RNN, Naive Bayes, and Adaboost. These techniques were found to be highly inefficient when high-resolution images were provided. In addition, accuracy significantly dropped when light conditions were varied within the dataset. Also, the algorithm's inability to handle more data classes was highlighted when the accuracy dropped from 95% to 80%.

In another research, Bag of Texton (BoT) method [5] is used, which is just an extended version of the Bag of Words (BoW) [6] in the NLP Domain. BoW can extract commonly occurring words in a given text. Similarly, BoT catches the dataset's most frequently and distinctively appearing visual patterns. This technique, however, has been found inadequate for any dataset involving many classes.

A Deep Convolutional Neural Network was developed by [7]**.** The algorithm was devised in a multi-headed manner by integrating subnetworks of trained AlexNet, GoogLeNet, and ResNet. This technique of EnsembleNet achieved a final result of 97.6%. As concluded, the results of this model will outperform any individual Neural Network.

Using hardware to measure the resistance of food items is another technique researchers have developed [8] [9]. In this case, an Arduino UNO measures resistance within a food dish from different angles. This range of resistance values is then compared with the ones present in the predefined database to find the name of the food dish.

Some research includes updating the existing "FoodLog" System with new classifier to improve accuracy from 89% to 92% [10]. They have proposed a method to incrementally update the classifier based on user's correction. For this, the research has compared SVM and Bayesian Networks to send the user feedback to the classifier. It was found that the Bayesian Networks outperforms SVM due to its multi-class classification capabilities and tolerates contexture features.

Smartphone-based applications have also been developed to recognize food dishes [11]. A region of interest is marked using a Histogram of Oriented Gradients (HOG), which is then split using Grab Cut. The research used a SURF-based Bag of Features (BoF) to extract a color histogram fed to an SVM classifier for recognition. This technique achieved an accuracy of 81.5%.

Scale Invariant Feature Transformation (SIFT) is used to form a visual codebook of texture patterns within a food dish, as highlighted by [12]. This texture pattern is further represented as spatial distribution and then encoded using shape context. Using the Pittsburgh Fast-Food Image (PFI) dataset, accuracy on different dishes range from 82% to 67%.

III. Proposed Flow Diagram

978-1-6654-9291-1/22/$31.00 ©2022 IEEE

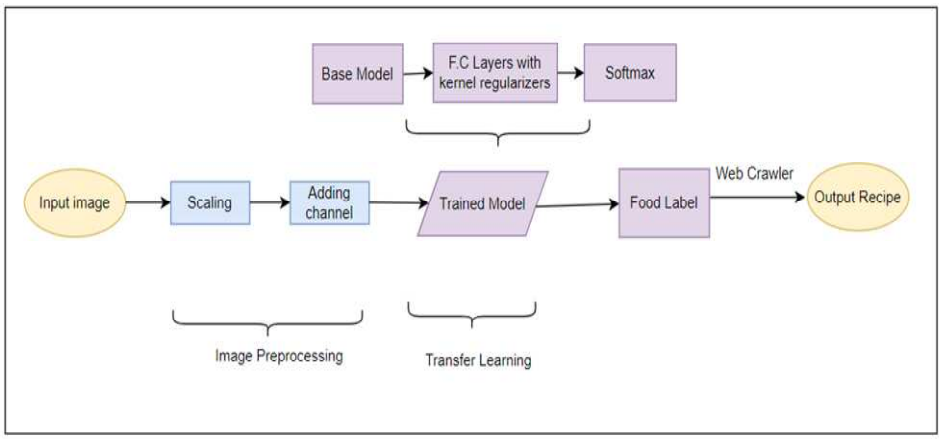

Fig. 1. Process Flow of the System

The proposed flow diagram is represented in Fig. 1. To initiate, the dataset is subjected to pre-processing, which includes splitting them into testing, training, and validation sets. Then images are scaled into 224X224 pixels, followed by adding an RGBA channel. This channel is necessary to make the overall dataset uniform and preclude the model from giving biased results. We trained our model using the base models, which are trained on the ImageNet dataset. Further layers are stacked on top of the base layer with the l1_l2 kernel regularizer [13] and Softmax as a classifier. By adding l1_l2 as a regularizer, we are reducing the chance of overfitting by introducing penalty to the model for every wrong output. Once the dish name is identified, we ran a web crawler to extract the recipe from the internet. For crawling, BeautifulSoup has been used, which returns a parsed HTML page from which we extracted meaningful terms. Each block is explained in detail under section V - Implementation.

IV. Dataset Description

We have compiled the dataset from two Kaggle sources. First, the "Indian Food Classification" consists of 20 classes of Indian food [14] . In all, there are 6271 total images in the provided repository. Secondly, The-massive-Indian-Food-Dataset consists of a total of 4770 images in PNG format split into 22 labels [15] . All photos are by default resized to 300 X 300. After pre-processing and filtering, we are left with 8189 images combined from both sources. These images are then fed into the training module. An excerpt from the training dataset is shown in Fig. 2.

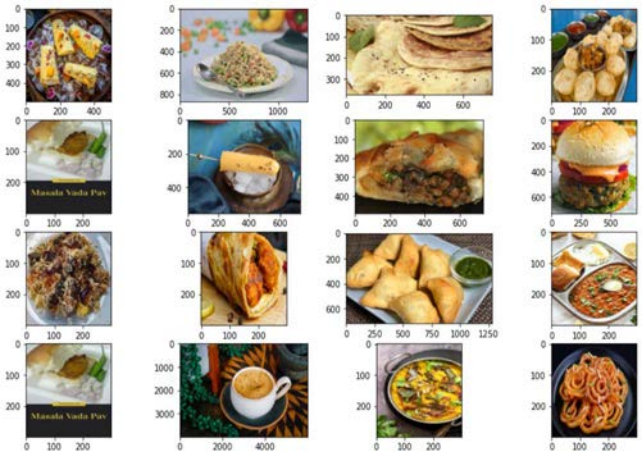

Fig. 2. An excerpt from the Dataset

V. Implementation

A. Image Processing

We have consolidated three Kaggle datasets and also scrapped images from google to increase the dataset as much as possible. The prepared dataset has been sub-grouped into three parts, namely Test, Train, and Validation in the ratio of 1:8:1. Images were resized and normalized appropriately, to 224x224 pixels and in RGBA format for the initial model implementations when using Transfer Learning. Image data was augmented through rotation, shifting, and horizontal flipping to make the model robust and avoid overfitting. This process makes the pre-trained model accommodate our dataset and normalizes the data.

B. Classification Algorithms

Modern Transfer learning [16] has been incorporated by loading the ImageNet weights into each model and freezing the inbuilt layers of each model while the top layers are replaced with our custom-made layers, which are meant to learn the classification of food dishes. We have used base models of ResNet, VGG19, EfficientNet, and DenseNet.

VGG19 is a deep CNN having 19 layers viz., 16 convolutional layers and three fully connected layers used for object classification [17] [18] . Using transfer learning on VGG19, we have added dense layers and modified the output layer to cater to our requirements.

We have used pre-trained ResNet50 since it focuses on the edges and provides more accuracy compared to VGG, as presented by [19]. It has fewer number of parameters and memory, making it computationally effective, as shown in the original paper [20]. The pooling layer reduces the parameters by two, and after the Convolutional layer and Relu, an identity of block input is added to the output block, which will reduce the training time.

DenseNet provides a solution to vanishing gradients, as shown in the original paper [21]. It has efficient back-propogation, strengthening propagation of features and concatenation to reduce feature map size in deeper layers and reuse it at the same time [20]. Each layer has access to gradients of all the previous layers [22]. The input size of our model was changed to 224 x 224. We have frozen the initial layers to minimize the training computational time and to

prevent it from modifying the existing weights of pre-trained DenseNet.

EfficientNet, as described in [23], is another network that improves accuracy by upscaling width, depth, and resolution by adding more feature maps at each layer, more layers to the neural network, and increasing the resolution of the image instead of downsizing images. Scaling is balanced since width, depth and resolution are increased by a constant ratio which requires more computational resources. We have used EfficientNet B0 to train our model. After extracting features, we added additional layers to provide a solution to our problem.

All the above networks were trained for 50 epochs with an early stopping condition having a patience of 10 to prevent overfitting [24]. The learning rate was initialized to 1e-4, which will be reduced by a factor of 0.1 when the loss metric has stopped improving. Softmax is used as an activation function in the case of multi-label classification problems, as it gives mutually exclusive outputs.

C. *Web Crawling*

Web Crawling has been integrated to suggest recipes once the Food Item is recognized. We have used BeautifulSoup [25] and Selenium [26] for this purpose. They work in a dynamic scraping mode, allowing us to utilize web browser content using Python Language. Thus, data extracted by JavaScript can be made available by triggering auto-click using Selenium, and Scraping could be achieved using BeautifulSoup. Once the JavaScript content is available, we can extract meaningful terms from the HTML Page tags.

VI. Experimental Results

A. *Results from Testing and Validation of Neural Networks*

Table 1. Testing Accuracy and Loss

Model	Accuracy (in %)	Loss
ResNet	87.42	0.2
VGG19	72	0.85
DenseNet	85.65	0.44
EfficientNet	96.52	0.58

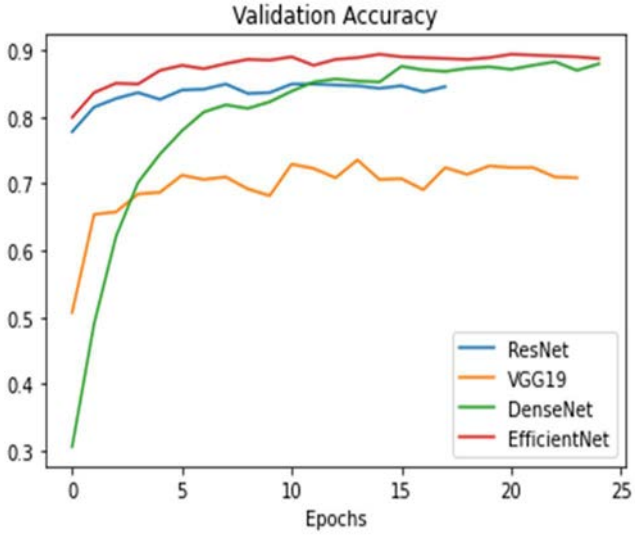

Fig. 3. Validation Accuracy of the trained models

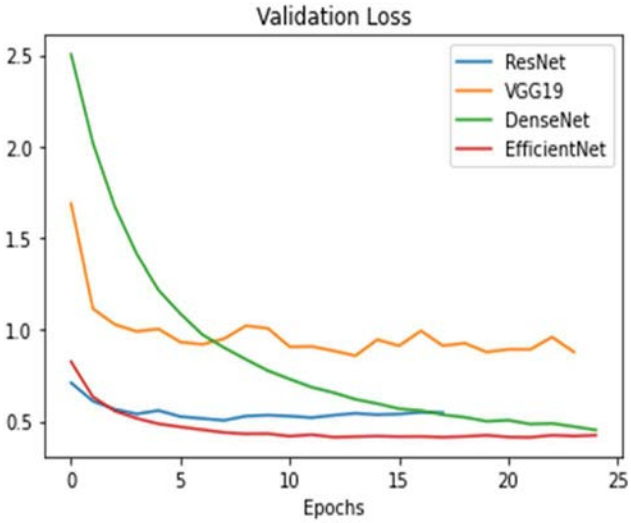

Fig. 4. Validation Loss of the trained models

Based on the plots and table of accuracy and loss on the training dataset and validation/real-world dataset shown in Table 1, Fig. *3* and *Fig. 4*, we can draw the following inferences-

VGG19 showed significant improvement in the first few epochs however had less improvement after ten epochs for both loss and accuracy. At epoch 13, the best results were found with an accuracy of 73.56% and a loss of 0.858. This behavior displays that the model was overfitted since training loss and accuracy improvement surpassed the validation loss and accuracy. Less amount of training data might be a reason for it.

ResNet ushered in with an accuracy of 77.78 % and 0.711 loss. It showed a similar pattern as that of VGG19 after the 7th epoch, and due to no improvement in accuracy for Ten epochs, the training was terminated after 17 epochs. The best results obtained by the model were having an accuracy of 84.93% and a loss of 0.504, which corresponds with the test results.

DenseNet showed an immense spike throughout the training process, starting with an accuracy of 30.59% and with a loss of 2.504 and reaching an accuracy of 87.98% and a loss of 0.4513 at epoch 24. More improvement in validation scores can be expected if the number of epochs were to be increased. Moreover, validation scores being better than testing scores suggests the model is more accurate than the testing results.

While EfficientNet commenced with 79.95% accuracy and 0.824 loss and bagged the highest accuracy of 89.39% with 0.412 loss. The stark difference between training and validation accuracy can be attributed to the model being trained to its best capacity. Thus, we can conclude that EfficientNet has the best results surpassing other models, though with training on a greater number of epochs, DenseNet might outperform EfficientNet for both metrics.

B. *Real-world Result*

We tested our best model (EfficientNet) on a real-world image, as shown in Fig. 5. The model applies prediction which results in a NumPy array having a separate index for each label in the dataset. The NumPy array would output "1" at the index of the identified dish. After applying keras.decoding_prediction [27], we get the desired output in a string format. A web crawler is then executed, which gives us the recipe to prepare the recognized dish.

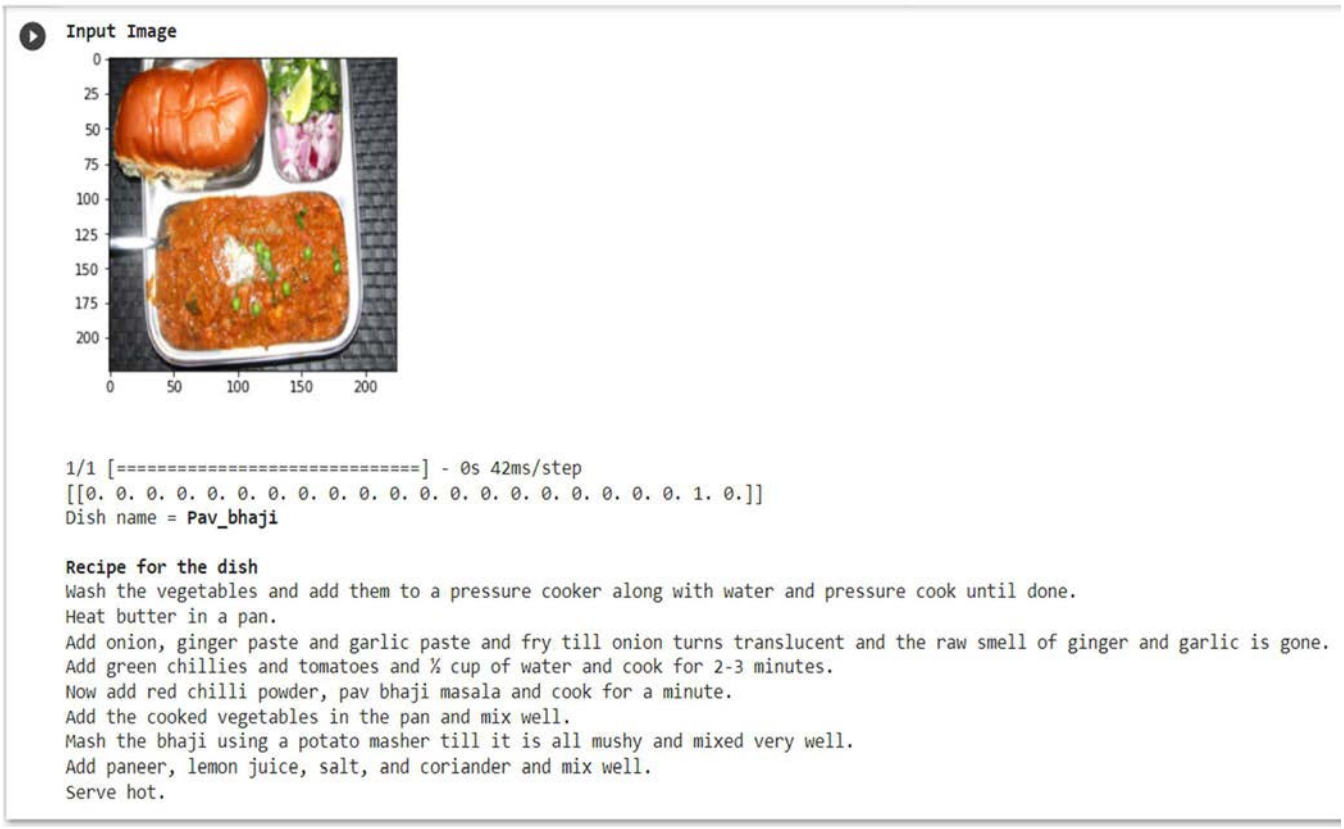

Fig. 5. Sample output on the Console

VII. Conclusion

In this project, we have used Transfer Learning technique to train models to recognize food images. For training, we employed several algorithms like VGG19, ResNet and DenseNet. The experimental results show that EfficientNet gave the highest validation accuracy (89.39%) among all other models. In the case of DenseNet, due to limited hardware bandwidth, we could train DenseNet only for 24 epochs. Had it been trained more, it could have given better results than EfficientNet. Finally, we implemented Web-Crawler in the form of BeautifulSoup and Selenium to suggest a recipe to the end user. In future, we aim to expand the scope of our project by building a mobile application and train on diverse dataset.

VIII. Future Scope

Due to limited hardware accessibility, we trained the model on only 22 classes of data. How the accuracy flares, when trained on a vast dataset is to be seen. As evident from the paper [7], the accuracy of the model will drastically improve by using an ensemble networking approach. Additionally, we can expand the scope of our model by building a mobile application and embedding a voice assistant.

References

[1] R. Chellappa, S. Theodoridis and A. van Schaik, "Advances in Machine Learning and Deep Neural Networks," IEEE, vol. 109, pp. 607-611, 2021.

[2] C. Pham and N. T. Thanh Thuy, "Fresh food recognition using feature fusion," International Conference on Advanced Technologies for Communications (ATC 2014), pp. 298-302, 2014.

[3] H. Kagaya, K. Aizawa and M. Ogawa, "Food Detection and Recognition Using Convolutional Neural Network," Association for Computing Machinery, pp. 1085-1088, 2014.

[4] Y. Lu, "Food Image Recognition by Using Convolutional Neural Networks (CNNs)," arXiv, 2016.

[5] G. M. Farinella, M. Moltisanti and S. Battiato, "Classifying food images represented as Bag of Textons," IEEE International Conference on Image Processing (ICIP), pp. 5212-5216, 2014.

[6] W. A. Qader, M. M. Ameen and B. I. Ahmed, "An Overview of Bag of Words," International Engineering Conference (IEC), pp. 200-204, 2019.

[7] P. Pandey, A. Deepthi, B. Mandal and N. B. Puhan, "FoodNet: Recognizing Foods Using Ensemble of Deep Networks," IEEE Signal Processing Letters, vol. 24, no. 12, pp. 1758-1762, 2017.

[8] crispylel, "https://create.arduino.cc/," 25 June 2016. [Online]. Available: https://create.arduino.cc/projecthub/crispylel/food-detector-6178cc.

[9] H. Prasaath, "engineersgarage," [Online]. Available: https://www.engineersgarage.com/arduino-based-smart-iot-food-quality-monitoring-system/.

[10] Y. Maruyama, G. C. de Silva, T. Yamasaki and K. Aizawa, "Personalization of food image analysis," Virtual Systems and Multimedia (VSMM), 2010 , 2010.

[11] Y. Kawano and K. Yanai, "Real-Time Mobile Food Recognition System," 2013 IEEE Conference on Computer Vision and Pattern Recognition Workshops, pp. 1-7, 2013.

[12] Z. Zong, D. T. Nguyen, P. Ogunbona and W. Li, "On the Combination of Local Texture and Global Structure for Food Classification," 2010 IEEE International Symposium on Multimedia, pp. 204-211, 2010.

[13] A. Bajpai, "Medium," 28 August 2021. [Online]. Available: https://medium.com/analytics-vidhya/implementation-of-regularization-techniques-l1-l2-in-keras-e0b39cb6a81d.

[14] P. Jain, "Indian Food Classification," October 2021. [Online]. Available: https://www.kaggle.com/datasets/l33tc0d3r/indian-food-classification.

[15] A. Mehta, "The-massive-Indian-Food-Dataset," January 2022. [Online]. Available: https://www.kaggle.com/datasets/anshulmehtakaggl/themassiveindianfooddataset.

[16] S. Bozinovski and A. Fulgosi, "Reminder of the First Paper on Transfer Learning in Neural Networks," Informatica (Slovenia), 2020.

[17] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," ICLR, 2014.

[18] J. Xiao, J. Wang, S. Cao and B. Li, "Application of a Novel and Improved VGG-19 in the Detection of Workers Wearing," Journal of Physics: Conference Series, Volume 1518, 2020 4th International Conference on Machine Vision and Information Technology, 2020.

[19] M. M. Leonardo, T. J. Carvalho, E. Rezende, R. Zucchi and F. A. Faria, "Deep Feature-Based Classifiers for Fruit Fly Identification (Diptera: Tephritidae)," SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI), 2018.

[20] K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," arXiv:1512.03385 [cs.CV], 2015.

[21] G. Huang, L. Zhuang, L. Van Der Maaten and K. Weinberger, "Densely Connected Convolutional Networks," IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2261-2269, 2017.

[22] P. Ruiz, "Understanding and visualizing DenseNets," 10 October 2018. [Online]. Available: https://towardsdatascience.com/understanding-and-visualizing-densenets-7f688092391a. [Accessed 2022].

[23] M. Tan and Q. V. Le, "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks," arXiv, 2019.

[24] "EarlyStopping," [Online]. Available: https://keras.io/api/callbacks/early_stopping/.

[25] L. Richardson, "Beautiful soup documentation," no. 2007, April.

[26] "The Selenium Browser Automation Project," [Online]. Available: https://www.selenium.dev/documentation/.

[27] Tensorflow.org, 26 October 2022. [Online]. Available: https://www.tensorflow.org/api_docs/python/tf/keras/applications/imagenet_utils/decode_predictions. [Accessed 17 November 2022].