Mushroom classification using Machine Learning

Sushmitha Kasimsetty Ramesh, Rohan Kanigere Umesh
CS530: AI for Radar and Remote sensing
Prof Haonan Chen
Colorado State of University
Fort Collins, CO, USA

‘

I. Abstract

Mushroom classification can be a time-consuming and labor-intensive task due to the vast number of mushroom varieties. However, recent advancements in machine learning and smart technology have made it possible to automate this process using deep convolutional neural networks. This technology can quickly and accurately identify the edibility of mushrooms based on their attributes such as cap color and structure, stem thickness and length, and habitat.

Here, we will delve into the application of machine learning in mushroom classification and its significance in identifying poisonous mushrooms. We will provide examples of both edible and non-edible mushroom types and discuss the importance of being able to differentiate between the two. Additionally, we will explain how a CNN-based classification tool has been trained on a dataset of segmented mushrooms to achieve better performance through hyperparameter tuning and adjusting pooling combinations. This tool can be used for real-time inference and has shown promising results in accurately detecting the edibility of mushrooms.

The integration of machine learning in mushroom classification can greatly improve the efficiency and accuracy of identifying different mushroom types, particularly in the context of mushroom cultivation, which is a popular source of income for many individuals. Furthermore, being able to quickly and accurately distinguish between edible and non-edible mushrooms is critical in ensuring public safety and avoiding potential health hazards.

II. Introduction

Mushrooms are a distinctive type of fungus that typically form umbrella-shaped fruiting bodies known as sporophores. They belong to the order Agaricales in the phylum Basidiomycota, but can also be found in other groups. Inedible or poisonous mushrooms are often referred to as ”toadstools,” while the two most commonly cultivated edible fungi are Agaricus Campestris and Bisporus. Given their popularity in everyday consumption and medical science, it’s crucial to identify and categorize mushrooms as edible, inedible, or poisonous.

When it comes to mushroom classification, there are a few factors to consider. Some mushrooms are unappetizing due to poor taste, while others are toxic. Poisonous mushrooms can even be identified by their flavor, and cooking may not always remove their harmful effects. For mushroom hunters, the challenge is to differentiate between edible, inedible, and dangerous mushrooms. Fortunately, humans are quite adept at categorizing objects based on their appearance and other characteristics.

Expert knowledge can also aid in mushroom classification. Factors such as the location of growth, whether the mushroom grows on the ground or on wood, the type of gills beneath the cap, the appearance of the surface, the shape of the stipe, geographic location, and even smell and touch can all provide important clues. In reality, there are 9 different classes of mushrooms, including ’Entoloma’, ’Suillus’, ’Hygrocybe’, ’Agaricus’, ’Amanita’, ’Lactarius’, ’Russula’, ’Boletus’ and ’Cortinarius’, which serve as the foundation for classification. (a). Entoloma: The Entoloma genus comprises a wide range of terrestrial mushrooms with pink gills, featuring over 1,000 species. The majority of these fungi have an unremarkable appearance, sporting olive, brown, or grey caps and pink gills attached to the stem. They also have angular spores and a smooth, thick cap. Some entolomas are mycorrhizal, while others are saprobic. The most well-known members of the genus are the livid agaric (Entoloma sinuatum) responsible for several poisonings in Europe and North America, and Entoloma rhodopolium found in Japan. Unusually colored caps can be found in some species from the southern hemisphere, such as Entoloma rodwayi and Entoloma viridomarginatum from Australia, and Entoloma hochstetteri from New Zealand.

(b). Suillus: The genus Suillus belongs to the Suillaceae family of basidiomycete fungus, which can be found in the Boletales order. These fungi are typically associated with pine trees of the Pinaceae family and are primarily found in temperate climates in the Northern Hemisphere. Some species have been introduced to the Southern Hemisphere.

(c). Hygrocybe: Hygrocybe is a type of agaric, or gilled fungus, that belongs to the Hygrophoraceae family. The fruit bodies, also known as basidiocarps, are vividly colored and have waxy to slimy caps, white spores, and smooth, ringless stems. They are commonly referred to as waxcaps in English, and waxcap grasslands in Europe are where many Hygrocybe species can be found. They are more commonly seen in forests elsewhere. These mushrooms are mainly ground-dwelling and thought to be companions to mosses. There are around 150 Hygrocybe species found around the world, with many of them being edible and occasionally sold in local markets.

(d). Agaricus: The Agaricus genus includes over 300 mushroom species, some of which are edible, while others are toxic. The two most commonly farmed mushrooms in the Western world are the common or "button" mushroom (Agaricus bisporus) and the field mushroom (Agaricus campestris). The distinguishing feature of Agaricus species is their fleshy cap, or pileus, with radiating plates or gills on the underside where the naked spores are formed. These mushrooms also have a stem or stipe that raises them above the substrate they grow on and a partial veil that shields the developing gills, forming a ring or annulus on the stalk. They are set apart from other members of the Agaricaceae family by their chocolate-brown spores.

(e). Lactarius: Lactarius is a genus of fungi that produces mushrooms and is ectomycorrhizal. Many species of this genus are edible, and they are often called milk-caps due to the milky fluid or "latex" that they release when cut or injured. Their flesh is similar to that of closely related species Russula and is fragile. The genus comprises around 500 species, mainly found in the Northern Hemisphere. The Lactifluus genus was recently separated from Lactarius based on molecular phylogenetic data.

(f). Russula: Russula is a genus of mushrooms that are ectomycorrhizal, meaning they form a mutually beneficial relationship with the roots of certain plants. There are over 750 species of Russula found all over the world. These mushrooms are easily identifiable by mycologists and mushroom collectors due to their abundance, large size, and vivid colors. Their caps are brightly colored, their gills are brittle and adhere to the cap, and there is no latex or partial veil or volva tissue on the stem. Under a microscope, Russula can be distinguished by amyloid decorated spores and spherocysts in the flesh. While some members of the Lactarius genus share similar traits, they release a milky latex when their gills are damaged.

(g).Boletus: Boletus is a genus of mushrooms that produce fruiting bodies with hymenial holes rather than gills. There are over 100 species of Boletus, and Carl Linnaeus named and characterized the genus in 1753. Since then, new genera have been discovered, such as Tylopilus, while old names like Leccinum have been reinterpreted. Some mushrooms that were once classified as Boletus have been moved to other genera, such as Leccinum scabrum. The majority of boletes are ectomycorrhizal, forming a mutually beneficial relationship with the roots of certain plants. Recent research has shown that Boletus is not a monophyletic group, with only a small percentage of the approximately 300 species belonging to the genus.

(h).Cortinarius: Cortinarius is a genus of mushrooms in the Cortinariaceae family found all over the world. It is believed to be the largest agaric genus, with around 2,000 widely distributed species. All young specimens in the Cortinarius genus have a cortina, or veil, between the cap and the stem, which is why the genus is named after curtains. While some species may have remnants of the cortina on the stem or cap edge, it usually disappears without a trace. All species in this genus have a reddish-brown spore print. Cortinar and webcap are common names for members of this genus. Non-experts are advised against consuming mushrooms from this genus due to the high toxicity of certain species, such as Cortinarius orellanus, and the difficulty in distinguishing between numerous species.

III. Dataset

The mushroom dataset available on Kaggle contains a diverse set of images that represent various common genera of mushrooms. This dataset is an essential resource for researchers, mycologists, and anyone interested in studying or identifying different types of mushrooms.

[7]The dataset includes images of edible, non-edible, and poisonous mushrooms, each belonging to a different genus. The images have been meticulously categorized and labeled to ensure that users can quickly and accurately identify the different types of mushrooms represented in the dataset.

This mushroom dataset is particularly valuable as it allows researchers to analyze and compare the different physical attributes and features of mushrooms from various genera. This information can be used to improve our understanding of the different types of mushrooms and their properties, such as taste, texture, and nutritional content. The availability of this mushroom dataset on Kaggle provides an excellent opportunity for researchers to conduct various analyses and experiments. It enables them to train machine learning models for mushroom classification and identification, as well as to explore different aspects of mushroom biology and ecology.

Overall, the mushroom dataset on Kaggle is a highly valuable resource for anyone interested in studying mushrooms, whether for research, education, or personal interest. Python by default only displays the top 5 records; the head method returns the top records in the data set. The dataset's shape attribute includes a number of observations and variables, and it is used to validate the data's dimension. The info() function is used to check the data and datatypes for each individual attribute. Last but not least, using the describe() method will allow you to view how data has been distributed for numerical values such as the minimum, mean, various percentiles, and maximum.

IV. Approach

A. Convolutional Neural Networks

Artificial Intelligence has been experiencing significant growth in the last few years, as researchers and enthusiasts work on improving various aspects of the field. One such area is Computer Vision, which aims to enable machines to view the world and perceive it like humans d [6]. Convolutional Neural Networks (CNNs) have emerged as the primary algorithm for advancing Computer Vision with Deep Learning. CNNs are a type of deep feedforward network that consists of one or more convolutional layers, pooling layers, complete connection, and an output layer. These networks assign learnable weights and biases to various objects in an image to differentiate one from another.

The structure of CNNs is similar to the connection pattern of the neurons in the human brain, which makes them effective in capturing spatial and temporal relationships in an image through the application of relevant filters. CNNs require less preprocessing compared to other classification algorithms, and they can extract picture features automatically without manual selection, labeling, or identification of image characteristics. With enough training, CNNs can learn filters and properties to understand the complexity of an image better.

The first convolutional layer collects low-level details, such as gradient direction and hue, and the architecture adapts to high-level features as the layers increase. The convolution operation extracts high-level characteristics such as edges from the input picture. The pooling layer shrinks the convolved feature's spatial size, reducing the computer power required to process the data. Max Pooling performs better than Average Pooling because it discards the noisy activations and performs de-noising along with dimensionality reduction. The Convolutional Layer and the Pooling Layer form the i-th layer of a Convolutional Neural Network, and the number of such layers can be increased to capture low-level details further.

For our project, we use a pre-trained machine learning classifier based on a convolution neural network architecture, which is currently the state of the art in image recognition. We collect many photos of previously categorized mushrooms with a verified edibility tag to train the classifier. We normalize the size and form of the images we use because most classification algorithms require predetermined input sizes. Our classifier only uses photos as input and ignores factors like scent and location. We anticipate that our model will be able to infer some important features related to appearance based on its convolutional layers' implicit feature extraction. We use CNN because of its efficiency, ease, quantification, self-adaptation, manual selection, labeling, and, most importantly, picture recognition features. We demonstrate the correctness of our classifiers using summary statistics that express the model's accuracy and predictive capacity.

B. Confusion Matrix

[3]After going through the various stages of preparing the data, including cleaning and pre-processing, the first thing we typically do is apply the data to a robust machine learning model to obtain a probability output. At this point, we need to assess the performance of the model, and that's where the confusion matrix comes into play. The confusion matrix is a classification performance metric in machine learning, and it comes in the form of an N x N matrix where N is the number of target classes. It allows us to evaluate how well our classification model is performing by comparing the actual target values with the model's predicted values. This metric is commonly used to address issues in both binary and multiclass classification problems.

To calculate the confusion matrix, we require either a test or validation dataset that has the expected outcome values. Then, for each row in the dataset, we make a prediction and compare it with the expected outcome. We then record the number of correct predictions for each class and the total number of incorrect predictions for each class. This information is then organized into a matrix, where each row represents the predicted class and each column represents the actual class.

We populate this matrix with the counts of correct and incorrect categorizations, recording the total number of correct predictions for each class in the expected row and predicted column for that class value. Similarly, the total number of incorrect predictions for each class is entered into the anticipated row and predicted column for that class value. Although the confusion matrix is commonly used in binary classification problems, it can be extended to multi-class classification issues by adding additional rows and columns to the matrix.

C. Training, Testing And Validation Of Data

[2]When training a machine learning algorithm, we utilize a training dataset that contains both the input data and the expected output. This dataset is mainly comprised of training sets, which account for about 60% of the overall data. During the testing phase, we adjust the model's weights and biases to fit the parameters. The training dataset is critical in training the model, particularly in neural networks where it is used to train the weights and biases that the model uses to learn from the data.

The validation dataset is a subset of the data used to provide an unbiased evaluation of the model's fit on the training dataset. This is useful in fine-tuning the model's hyperparameters, as the model's performance on the validation set is included in the model setup. However, as the evaluation becomes more biased, it is essential to be careful not to overfit the model to the validation set.

The test dataset is a separate subset of data used to assess the model's fit on the training dataset without any bias. This is achieved by using data that the model has not previously seen during the training or validation phases. Typically, about 20% of the data is used for testing sets. The test dataset is only used once the model has been fully trained using the training and validation sets. It is important to use a well-curated test set that includes properly collected data from various classes that the model may encounter in the real world. This enables a fair comparison between competing models and provides a gold standard for evaluating the model's performance.

D. PyTorch and PyTorch Lightning

PyTorch is a popular Deep Learning framework that allows users to build and train standard models with ease, as well as create complex AI models. However, as the research becomes more involved and requires multi-GPU training, 16-bit precision, and TPU training, users are more likely to make errors. To address this issue, PyTorch Lightning was developed as a high-level framework for PyTorch that abstracts away implementation details. This allows users to focus on building great models and saves them time by eliminating trivial details.

PyTorch Lightning is a Python package that provides a high-level interface for PyTorch. It organizes PyTorch code into a structure that makes it reusable and shareable, making

deep learning experiments easier to understand and replicate. It allows users to build scalable deep learning models that can operate on distributed hardware while being technology agnostic.

The Lightning framework enforces a specific code structure, which includes research code (the LightningModule), engineering code (handled by the Trainer), non-essential research code (logging, etc. handled by Callbacks), and data (organized using PyTorch DataLoaders or a LightningDataModule).

PyTorch Lightning offers several advantages over PyTorch's unstructured version. Models become agnostic to hardware, and because engineering code is abstracted away, the code is easier to read and recreate. It also helps users make fewer mistakes since Lightning takes care of the tough engineering. Additionally, Lightning maintains full flexibility, but eliminates a lot of boilerplate code.

The Lightning Module is made up of six components: the model, the optimizer, the training loop, the validation loop, the testing loop, and the prediction loop. Each component has its respective classes/functions, with only necessary characteristics defined. This reduction in boilerplate allows for simpler code and a decreased chance of making minor mistakes, while still preserving flexibility.

E. Model Training

[4]The project aims to use a pre-trained machine learning classifier with a convolutional neural network architecture to classify images of mushrooms based on their edibility. To achieve this, a training dataset is used to teach the algorithm how to learn and generate outcomes using neural networks. The dataset is then divided into three subsets: a training set, which comprises 70% of the data, a validation set consisting of 20% of the data, and a test set which makes up 10% of the dataset.

The validation set is used to provide an unbiased evaluation of how well the model fits the training dataset when adjusting model hyperparameters. To ensure the model is set up correctly, competence on the validation dataset is included in the model setup, which is then used to verify the model and adjust higher-level hyperparameters.

The test dataset is used to provide an unbiased assessment of the final model's fit on the training dataset. It's crucial to avoid using the training dataset during the testing stage of AI projects since the algorithm would already know the expected outcome, leading to inaccurate evaluations. As a result, the test dataset comprises 20% of the overall dataset and is considered the gold standard for evaluating the model. The test set includes properly collected data from a variety of classes and is typically used to compare and contrast competing models.

V. Experimental Analysis

A. Source Code

[5]The ImageClassifier class has been designed to facilitate the training, validation and testing of data using a variety of methods. To begin with, the image classifier is capable of

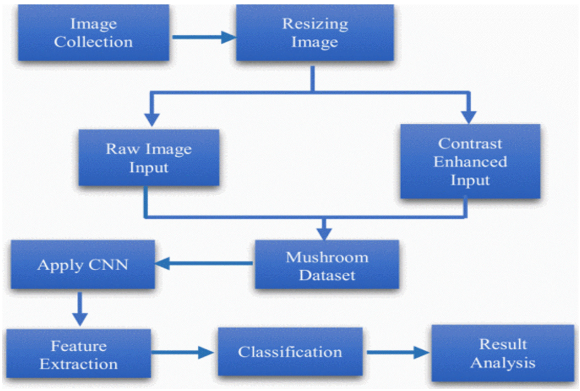

Fig. 1. Flowchart

accepting unprocessed image datasets for training, validation, and testing. These datasets are subjected to several transformations including resizing to 255 x 255 pixels, center cropping to 224 x 224 pixels, and normalization.

The training method of the Image Classifier makes use of PyTorch lightning, where only the last layer is trained for a specified number of epochs before the whole backbone model is unfrozen and fine-tuned. In order to perform inference on testing data, the forward function is used, as it is required by the framework. The configure optimizers function is then used to set up the optimizer for the model, while the train_dataloader() method is used to optimize the loading of training data into the model. Similarly, the val_dataloader() and test_dataloader() methods are used to optimize the loading of validation and test data respectively.

Moreover, the environment variable CUDA_LAUNCH_BLOCKING=1 is set up to disable asynchronous kernel launches at runtime, in accordance with the CUDA programming guide. This setting proves to be useful for debugging when there are concurrent kernels and transfers involved. It is important to note that the CUDA_LAUNCH_BLOCKING setting will not affect the streams API at all. However, when additional debug code is added to force all streams code to use stream 0, all calls other than kernel calls will revert to synchronous behavior.

B. Libraries

The framework we are working with requires a number of different libraries to be imported in order to access the necessary functionalities. These libraries include TensorFlow, Keras, seaborn, and pandas. TensorFlow is a powerful open-source library that allows for the development and training of end-to-end machine learning models, with a flexible ecosystem of tools that can be used to create and deploy models. Keras, on the other hand, provides an intuitive, high-level API with eager execution, serving as an interface for TensorFlow that can help reduce the cognitive load and streamline the process of model iteration and debugging. Meanwhile, pandas is an open-source data analysis library that also functions as a manipulation tool, and Seaborn integrates with pandas to

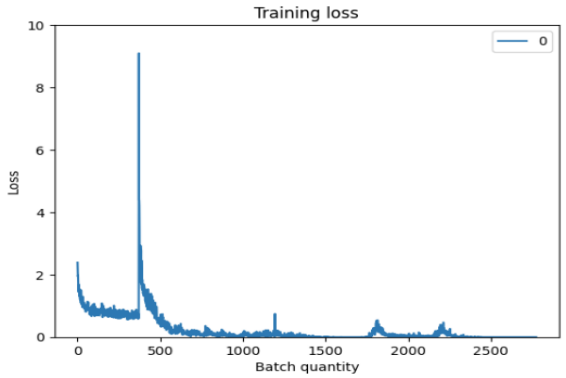

Fig. 2. Training Loss

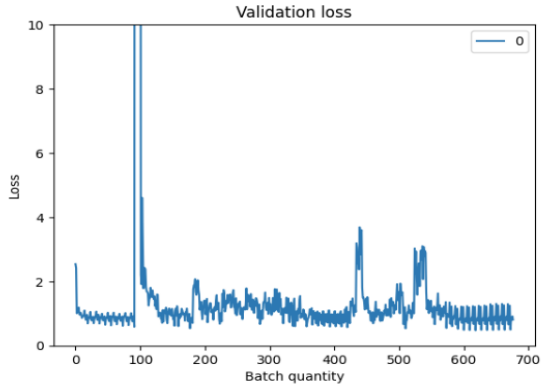

Fig. 3. Validation Loss

offer statistical graphics in Python, enabling the plotting of functions, semantic mapping, and statistical aggregation to generate informative visualizations. Finally, we also make use of the PyTorch lightning framework, which is a high-level interface for PyTorch, a Python package that provides both tensor computation and strong GPU acceleration. PyTorch lightning abstracts away much of the deep learning boilerplate, while still allowing for complete control and flexibility over the code.

VI. Results

[1]In terms of results, the training dataset for this project comprises almost 70% of the overall data, which equates to 4699 images. The validation dataset accounts for approximately 20% of the data, or 1342 images, while the test dataset represents about 10% of the data or 673 images. When analyzing the training process, the training loss increases significantly around the 1000 batch quantity mark, specifically at the 9th epoch. On the other hand, the validation loss becomes challenging to control when the batch quantity surpasses the 100 mark and reaches around the 200 mark. However, the training and validation losses experience a substantial decrease as the batch quantity increases, with the training loss becoming nearly zero and the validation loss approaching a value of 1. Ultimately, the model achieves an accuracy of around 72%.

References

[1] A. K. Agarwal, V. Khullar, and N. K. Agrawal. Scalable machine learning for mushroom dataset classification. In *2021 International Conference on Simulation, Automation Smart Manufacturing (SASM)*, pages 1–4, 2021.

[2] J. Dong and L. Zheng. Quality classification of enoki mushroom caps based on cnn. In *2019 IEEE 4th International Conference on Image, Vision and Computing (ICIVC)*, pages 450–454, 2019.

[3] S. Ismail, A. R. Zainal, and A. Mustapha. Behavioural features for mushroom classification. In *2018 IEEE Symposium on Computer Applications Industrial Electronics (ISCAIE)*, pages 412–415, 2018.

[4] N. Kiss and L. Czùni. Mushroom image classification with cnns: A case-study of different learning strategies. In *2021 12th International Symposium on Image and Signal Processing and Analysis (ISPA)*, pages 165–170, 2021.

[5] K. Tutuncu, I. Cinar, R. Kursun, and M. Koklu. Edible and poisonous mushrooms classification by machine learning algorithms. In *2022 11th Mediterranean Conference on Embedded Computing (MECO)*, pages 1–4, 2022.

[6] A. Wibowo, Y. Rahayu, A. Riyanto, and T. Hidayatulloh. Classification algorithm for edible mushroom identification. In *2018 International Conference on Information and Communications Technology (ICOIACT)*, pages 250–253, 2018.

[7] L. Wu and Y. Chen. Mushroom recognition and classification based on convolutional neural network. In *2022 IEEE 5th Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC)*, volume 5, pages 1430–1433, 2022.