Classification of Edible and Poisonous Mushrooms

Hades Li, Yige Luo, Jibo Shen, Ming Zhao

06/06/2022

Introduction

As the back-to-nature and organic food movement proceeds, interests in wild mushroom hunting have risen substantially in recent years [3]. In the United States, over 7000 incidents per year of mushroom poisoning have been reported since 1999 [2]. According to a 2014 review, identified fungus species have reached approximately 100,000 worldwide, and about 100 of them are poisonous to humans [4]. Although toxic fungi seem to be rare overall, with 600 newly discovered fungi annually, the number of toxic mushrooms steadily increases [1]. It is possible to differentiate between poisonous and non-poisonous mushrooms with proper training, yet the public access to those training is still highly limited.

Traditional wisdom believes that toxic mushrooms are colorful, but many toxic ones look plain at first sight [Figure 1], making the toxicity inference very challenging to human eyes. On the other hand, as toxicity of mushrooms comes from certain chemicals, with the most well-known and deadly toxin called α-amanitin [6], a natural and more informative approach would be deciphering the underlying chemical compositions of fungi. However, such information is often infeasible to obtain, while many images are readily available from a single mushroom hunting trip. With help from domain experts and the development of computer vision, identified mushrooms could be then used to train supervised machine learning models, which in turn can serve as a handy tool for future mushroom amateurs and field scientists.

Figure 1. **Selected fungi images of all four fungi classes**, showing bright coloration is not a reliable indicator of toxicity. The fourth fungi example in Edible MS and the first example in Edible S are colorful but edible. The third fungi example in Poisonous MS and the first example in Poisonous S have ordinary yet deceitful appearances. Abbreviations are MS: mushroom sporocarp; S: sporocarp.

The emergence of AlexNet in 2012 heralds the revival of the field computer vision [8], which is shown to be superior to human eyes in object recognition. Over the past decade, more sophisticated computer vision models have been proposed and deployed with increasing sensitivities to detect subtleties that human eyes fail to capture. Among the widely-used deep learning modes, the 2015 ImageNet detection winner ResNet has excellent performance (3.57% error on 150,000 test set images with 1000 classes) in pattern recognition [7], and therefore demonstrates great potential to be applied in poisonous fungi recognition. As the consequence of misidentifying a poisonous fungus to be edible is more fatal than the other direction, we focused on improving the recall metric given fixed precision level (80%), instead of optimizing the overall prediction accuracy.

In this study, we show that a precise, reliable (~95% recall rate with 80% precision) and real-time identification is feasible by leveraging the power of pre-trained computer vision models ResNet, which are also uniformly more powerful than traditional machine learning methods. Among deep learning models, we also found that increasing model complexity does not translate to better precision-recall performance, despite considerable gain in overall accuracy.

Proposed Methods

Transfer learning with ResNet

Although training deep neural networks is usually computationally expensive, unlike traditional machine learning models, deep learning models could be easily applied to novel datasets with decent performance [9]. Often dubbed as transfer learning (TF), this phenomenon obviates training models from scratches and makes computer vision models appealing candidates for many tasks including image classification, especially when the data collection is non-trivial and sample sizes are consequently small [10]. In this study, we conducted transfer learning on fungi data with the widely used model ResNet, where well-curated beginners' guides and tutorials can be found at Pytorch. Like many other computer vision models, at the base of ResNet lies the convolutional neural network (CNN), which we briefly describe below.

Convolutional Neural Network (CNN)

The precursor of CNN - neural network (NN) was originally introduced in the 1980s to model complicated output [11]. Inspired by the architecture of the human brain, neural networks consist of a single input and output layer, between which often lie many hidden layers, each harboring an array of neuron-like hidden units. Feature extraction is accomplished through a series of nonlinear transformations of the weighted sum of input data. Due to the flexibility of model specifications (number of hidden layers, number of hidden units, choice of non-linear transformation), neural networks could easily expand the feature space for models to learn. Yet, with the number of features far exceeding the sample size, these models pose challenges on optimization and subsequently give birth to high-dimensional analysis (e.g. in image classification).

As a specialized version of NN to cope with images, CNNs consist of two types of layers: 1) convolutional layers, which use diverse convolutional kernels as filters to extract local features; and 2) pooling layers, which can reduce image pixel sizes and therefore model complexity. By using multiple convolutional kernels, local features with varying degree of neighborhood can be extracted and learned by the classifier. Similar to the way humans perceive and process images, multiple convolutional layers are usually designed to capture higher-level assemblies (edges, shapes, parts and eventually objects) from image pixels, as implemented in ResNet.

ResNet

The ResNet proposed by He et al. is a series of deep CNN models with multiple convolutional layers ranging from 18 to 152 [7]. To train a poisonous fungi classifier transferred from ResNet, we chose 3 ResNets: the simplest 18-layer (ResNet18), intermediate 34-layer (ResNet34) and the most complex 152-layer (ResNet152). Adapted from He et al., the architecture of selected models

is summarized in Table 1[1]. Starting from images of size 224*224, all ResNet share the basic design with 5 convolution stages, with output sizes decreasing by a factor of 2. Instead of simply increasing network depth, stacks of convolutional layers are used in convolution stages 2 through 5 to avoid the vanishing gradient problem [12]. As mentioned earlier, these packed convoluted layers can learn higher-level features (Figure 5). The key difference between the ResNet designs is the number of stacked layers and size of kernels. We changed the fully connected layer from 1000 dimensions (as ResNet was used to solve 1000-class classification) to 2 dimensions and re-trained model weights accordingly. As the last step, the predicted logits for each class are converted to assignment probabilities by a softmax function.

Table 1. ResNet architecture

layer name	output size	18-layer	34-layer	152-layer
conv1	112×112			
conv2_x	56×56	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax		

Logistic Regression, K-Nearest Neighbors, Random Forest

Besides CNN, we also propose three supervised learning methods, Logistic Regression (LR), K-Nearest Neighbors (KNN), and Random Forest (RF) for model comparison. For the LR method, it uses the logit function to limit prediction values between 0 and 1 and uses maximum likelihood estimation to estimate the parameters of the model. For KNN, it calculates the distance of a new data point to all other training points, then selects the K-nearest points, and finally assigns the data point to the class of which the majority of the K data points belong to. For RF, it is an ensemble method consisting of decision trees and bagging, where bagging is defined as bootstrapping plus aggregation and is designed to decrease variance by introducing randomness.

[1] Note that 2-dimensional convolution kernels are applied to each of 3 RGB-channels separately.

Real Data Study

Data

Our project dataset is "Edible and Poisonous Fungi" from Kaggle [5]. The dataset contains 3401 fungi images and has 4 identified classes. Specifically, there are 2 types of fungi, mushroom sporocarps (MS) and sporocarps (S); in each type, they are further divided into 2 groups, edible and poisonous. We display the summary about the 4 classes in Table 2 below. Beyond the given image classification, no additional features are provided. As we are interested in whether the fungi are poisonous or not, we do not distinguish the types of fungi in this project. This results in 1181 edible-mushroom images and 2220 poisonous-mushroom images.

Table 2. Summary of image data

	Edible		Poisonous	
	MS	S	MS	S
count	715	466	860	1360

Preprocessing

We start by splitting the 3401 mushroom images into 64% train, 16% validation and 20% test set, stratified by 4 respective class sizes. As the central goal is to predict fungi toxicity, we combined edible MS and edible S into edible fungi, and the other two classes into poisonous fungi. Under the PyTorch deep learning framework, we imported, transformed the image data to 3-dimensional RGB tensors in a batch of 16, and normalized each color channel using channel means and standard deviations of the >1M training images from ImageNet. For our training set (756 edible and 1420 poisonous fungi), we performed data augmentation by randomly resizing and cropping the image to a size 224*224 pixels, and introduced random horizontal flips (as mushrooms are horizontally symmetric). These steps ensure that the images are on the same scale and learning robustness. For the validation (189 edible and 355 poisonous fungi) and test (236 edible and 445 poisonous fungi) set, we only resized images to 256*256, followed by a center crop to a final size of 224*224 pixels.

To train other proposed methods, we split the image data into training and testing sets stratified by the number of total edible and poisonous mushrooms with the proportion of 80%, and 20%, respectively. The size of the training set increases to 945 edible and 1775 poisonous mushrooms, and the size of the testing set stays unchanged as used for CNN. We then transform the two data sets as the approach we did for CNN as well. For training LR, KNN, and RF, we need to reshape the data from 4 dimensions to 2 dimensions. For instance, the dimension of the training set is (2720, 3, 224, 224), where the four values represent the sample size, color, height, and weight,

respectively. After reshaping, the dimension becomes (2720, 3*224*224). It is important to note that the number of parameters is therefore much larger than the number of observations.

Training Models

We select ResNet18, ResNet34, and ResNet152 (with increasing model complexity) to train our fungi classifier. Instead of using these pre-trained models as feature extractor, we adopted its architecture and fine-tuned model weights from their pre-trained values. For loss function, we tried cross-entropy loss, a common loss for classification, and its extension focal loss to accommodate the class imbalance. Although the focal loss should perform better theoretically, we found that cross-entropy loss is a good selection for our model [Table S1]. For the optimizer, we used stochastic gradient descent (SGD) with several starting learning rates (0.01, 0.001 and 0.0001) and fixed momentum. We also tried another optimizer Adam to adaptively change the momentum during optimization, yet the performance is not as good as SGD [Table S2]. We decreased the learning rate by a factor of 10 once every 7 epochs. The best model is returned as the one with the highest accuracy on the validation set. For the training process, we also try several different numbers of epochs (starting from 25), each representing a random configuration of training images after augmentation. Although more epochs are generally advisable, no more significant improvements are found after 25 epochs are used, justifying our final choice of 25. On an Intel i7 4-processor CPU, it takes ~3 hours to train the ResNet18 model, and the computation time can be reduced to ~8 minutes on a CUDA compatible GPU. As a result, the highest training accuracy is 0.966, 0.938, and 0.955 for ResNet18, ResNet34, and ResNet152, respectively, and their validation accuracy is 0.827, 0.840, 0.844, accordingly.

In addition, we applied the 5-fold cross validation for LR, KNN, and RF using default hyperparameters[2]. We chose the default mode because we assumed CNN would perform significantly better than the three models, and the default mode combined with cross validation could serve the purpose of model comparisons. Among the three models, LR performed the worst, with the mean validation accuracy based on the 5 iterations being only 0.535, compared with KNN (0.612) and RF (0.636). The poor performance of LR is probably due to the failure of convergence in this high-dimensional case.

[2] In "sklearn" package, some of important default parameters of LR is l2 for penalty, 1 for inverse of regularization strength, L-BFGS for solver, l00 for the maximum number of iteration, and 0.0001 for the tolerance error; default parameters of KNN include 5 as the number of neighbors and Minkowski as the distance metric; default parameters of RF include 100 as the number of trees in the forest, Gini impurity as the measure of the quality of a split, square root of total features as the number of features when looking for the best split, and nodes that expanded until all leaves are pure or contain less than 2 samples as the maximum depth of the tree.

Testing Models

To evaluate the model performance, we applied the models to the test data and compared the predicted results to the ground truth. We calculated accuracy, precision, and recall based on the confusion matrix. Table 3 shows the three measures of the proposed models. We can see that all the CNN models perform well. In particular, every measure of the three CNN models is around or above 80%. In contrast, LR and KNN perform generally worse on all three measures. For RF, accuracy and precision is also low, but the recall is higher than the CNN models. This characteristic suggests that the selected RF model is overly liberal in assigning fungi as being poisonous. in which it is more than 90%. Among the three ResNet models, Resnet152 performs the best in terms of all measures based on the results shown below, followed by ResNet 34 and ResNet18. Although modeling training could take up to 40 minutes on GPU, prediction only takes a couple seconds by loading saved models, therefore enabling a real-time identification on a mobile app with live-streamed fungi images.

Table 3. Model evaluations on testing set

	ResNet18	ResNet34	ResNet152	LR	KNN	RF
Accuracy	0.794	0.816	0.827	0.545	0.606	0.655
Precision	0.886	0.886	0.89	0.663	0.669	0.669
Recall	0.787	0.825	0.838	0.618	0.787	0.935

Visualization

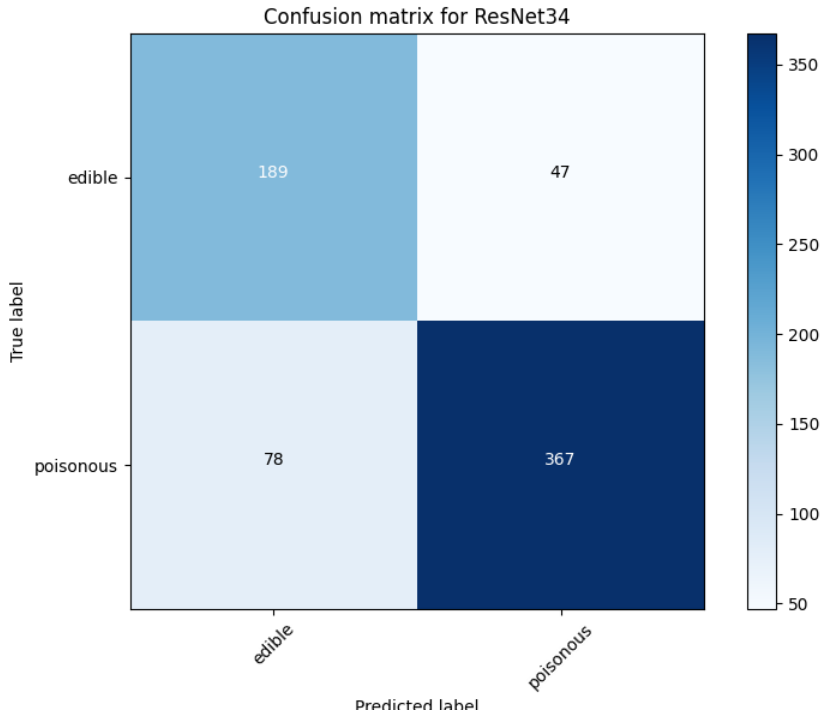


Figure 2. **Confusion matrix for Resnet34** classifying 2 groups from the test set. Ground truth labels are plotted row-wise and predicted labels column-wise. True Negative (TN) is 189, False Positive (FP) is 47, False Negative (FN) is 78, and True Positive (TP) is 367. Accuracy is the sum of TP and TN over the number of observations (0.816). Precision is TP over the sum of TP and FP (0.886). Recall is TP over the sum of TP and FN (0.825).

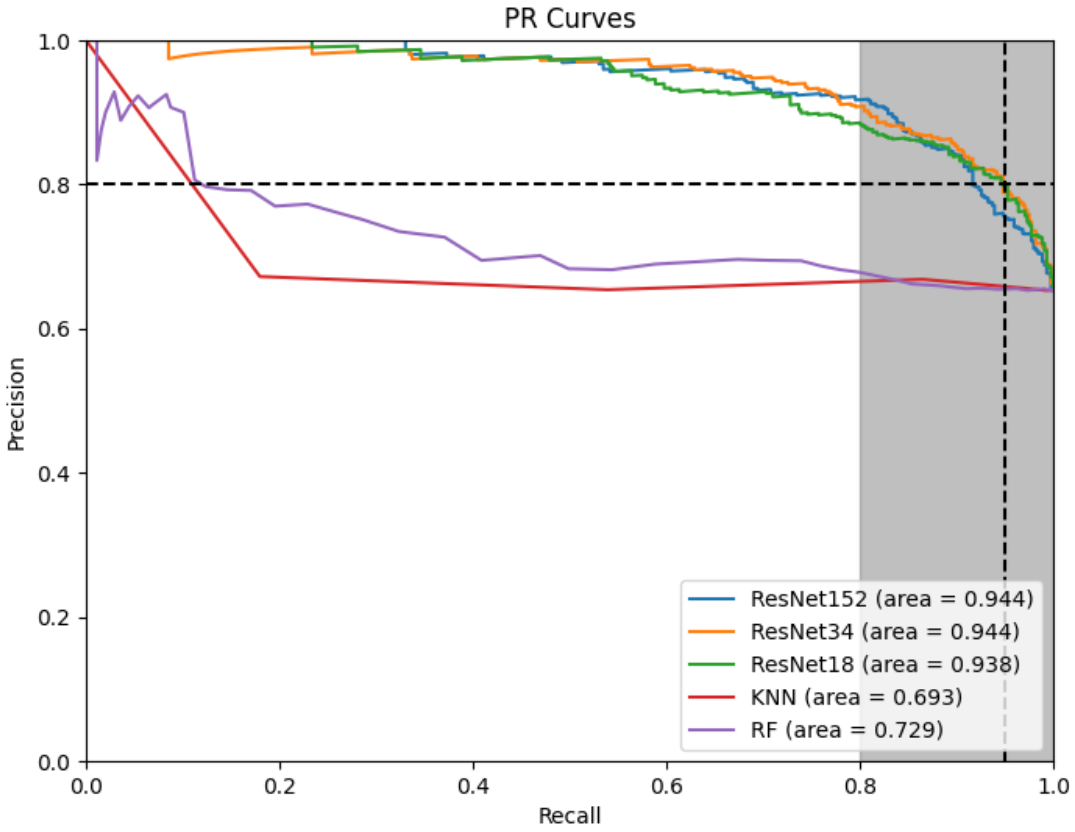


Figure 3. **Precision-Recall Curve** of 5 models, showing that ResNets are uniformly better than other machine learning models in this project. Fixing precision at 0.8 (horizontal dotted line), we can see that there is no clear difference in recall between ResNet18 and ResNet 34, both reaching a recall rate at 0.95 (vertical dotted line). The area with recall rate greater than 0.8 is shaded in gray.

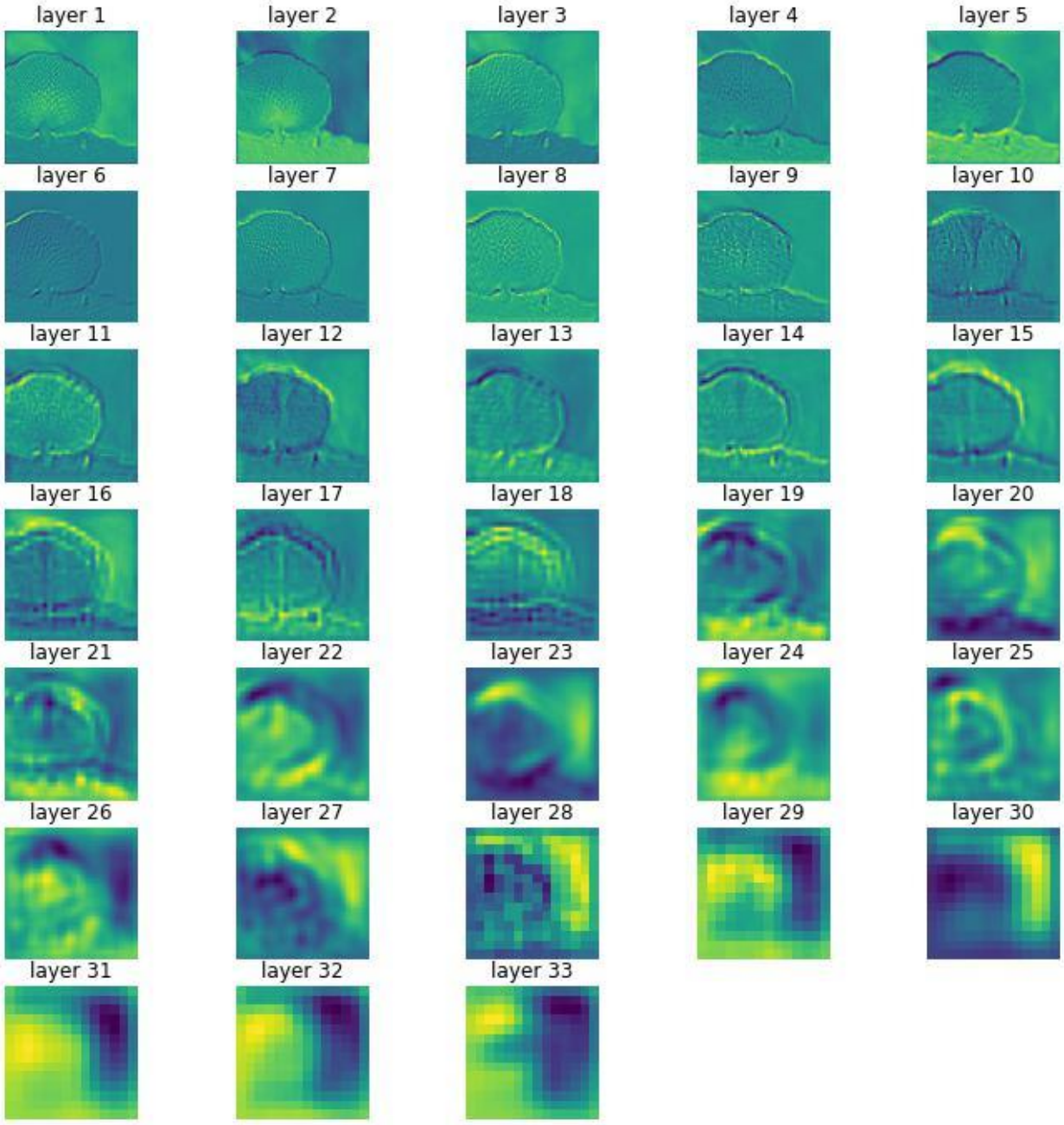


Figure 4. **Feature Map of ResNet34 for Intuition.** We notice that each layer catches different information. The early layers can catch some higher-level features like position and silhouette, while the later layers catch some lower-level features, such as color and shadow. In this way, the machine can learn much more detailed information from the picture than humans' eyes. It is the reason why we concentrate on using machine learning techniques for distinguishing edible fungi in this project.

Discussion & Conclusion

ResNet consistently outperforms traditional machine learning methods

As expected, all three fine-tuned CNN classifiers from ResNet have better performance in fungi identification (Figure 3, Figure S5), all capable of recognizing >90% from all poisonous fungi in the test set with satisfactory precisions. Since misclassification when fungi is actually poisonous is more problematic, attention should be paid to models with high-recall rate regions (shaded area in Figure 3). We found that increasing model complexity does not necessarily lead to gain in recall rate, as ResNet152 suffers from a steeper decrease in precision when entering >90% recall region, probably due to overfitting. We also found that "robust" machine learning methods like random forest and k-nearest neighbors perform poorly, resulting in a merely ~15% recall rate, given 80%, a common threshold for precision. To put in another way, blindly predicting all fungi to be poisonous will reach 65.3% prediction precision (there are ~65.3% poisonous fungi images across all dataset), learning through KNN and RF has little improvement on precision at the high-recall-rate region. Further feature engineering procedures, such as principal component analysis or using ResNet as feature extractors, could be implemented before training to further test the performance of KNN and RF.

Fine-tuning hyperparameters of ResNet do not lead to model improvement

To improve model prediction performance, we tried various ways to finetune model weights according to the update rule during optimization, yet none of our attempts lead to a noticeable improvement, regardless of targeting on learning rate or gradient calculation (Table S1, S2). To our surprise, the seemingly better optimizer alternative to SGD, Adam, as well as the seemingly better loss function alternative to cross-entropy, focal loss, both turned out to be inferior. In the case of loss function, it might be that poisonous fungi are more represented (65.3% vs. 34.7%) than edible ones, opposite to the common use case when the label of interest is rarer. As the training loss levels off in the first few epochs, our attempts of increasing the number of epochs did not improve our model. Since a large learning rate will forfeit the pre-trained model weights as if training from a scratch (comparing Figure S4 and S5), we limited the search scope of initial learning rates to be no larger than 10 times the tutorial recommended value (0.001), and did not find any effect. The only improvement we found is when fitting a more complex ResNet model (ResNet152), where higher training accuracy is observed despite that it cannot be generalized to test sets. Given the modest sample size of fungi data, our results suggest that little improvement could be made within the transfer learning framework of ResNet, while we acknowledge that our evaluation of hyperparameter combinations is not exhaustive. A few further directions include performing cross-validation, investigating misclassified cases, as well as transfer learning from more state-of-the-art computer vision models on ImageNet.

4-group vs. 2-group classification

Although we decided to focus on 2-group classification in this study, there are practical reasons in favor of including the omitted dimension, i.e. mushroom sporocarps (MS) and sporocarps (S). In fact, these two types of fungi are easily distinguishable by human eyes. For example, mushrooms have a stalk and cap, while sporocarps like dietary black fungus lack such structure. However, due to the class imbalance between MS and S and the modest sample size in the dataset, predicting with 4 classes suffer from per-class recall rate (Figure S3), especially in the smaller classes (edible MS and poisonous S). As a result, caution should be made when reading the edible fungi prediction from our 2-group models. The conditional probability of it being misclassified as edible could differ drastically on whether it is MS or S, since the per-class recall rate for poisonous MS is only 0.552 compared with 0.787 when combined with poisonous S. We also note that if pooling is performed after 4-group classification, the test accuracy and precision remain roughly the same as the 2-group classification, while a slight gain is observed in recall rate (0.813 vs. 0.787). Besides exploring new methodological aspects, this might open up a new perspective for better model predictions.

Conclusion

The CNN model outperformed the traditional classification methods we proposed, i.e. logistic regression, KNN, and random forest. Inside the CNN models, hyperparameter tuning did not show great improvement. Adding layers indeed brought higher accuracy and recall rates, but also increased model size and complexity. ResNet34 is selected as the best pre-trained model for our project, which generated 0.95 recall given a fixed 0.8 precision on the test dataset. As known toxic fungi only account for 3% of all existing fungi diversities, future efforts should focus on further improving the recall rate (>97%) to make our model applicable in a natural setting.

Acknowledgement & Reference

We are grateful to Prof. Bo Ning and Xiner Zhou for course materials, instructions and suggestions for data analysis methodology. We also thank Pu Li for exchanging thoughts on using models from computer vision.

[1] Lima , A. D. L., Fortes, R. C., Novaes, M. R. C. G., & Percário, R. (2012). Poisonous mushrooms: a review of the most common intoxications. *Nutricion Hospitalaria*, *27*(2), 402–408. https://doi.org/10.1590/S0212-16112012000200009

[2] Brandenburg, W. E., & Ward, K. J. (2018). Mushroom poisoning epidemiology in the United States. *Mycologia*, *110*(4), 637–641. https://doi.org/10.1080/00275514.2018.1479561

[3] Eren, S. H., Demirel, Y., Ugurlu, S., Korkmaz, I., Aktas, C., & Güven, F. M. (2010). Mushroom poisoning: Retrospective analysis of 294 cases. *Clinics*, *65*(5). https://doi.org/10.1590/s1807-59322010000500006

[4] Graeme, K. A. (2014). Mycetism: A review of the recent literature. *Journal of Medical Toxicology*, *10*(2), 173–189. https://doi.org/10.1007/s13181-013-0355-2

[5] https://www.kaggle.com/datasets/marcosvolpato/edible-and-poisonous-fungi

[6] https://en.wikipedia.org/wiki/A-Amanitin

[7] K. He, X. Zhang, S. Ren and J. Sun. (2016) Deep Residual Learning for Image Recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770-778, doi: 10.1109/CVPR.2016.90.

[8] Krizhevsky, A., Sutskever, I. & Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. In F. Pereira, C. J. C. Burges, L. Bottou & K. Q. Weinberger (ed.), *Advances in Neural Information Processing Systems 25* (pp. 1097--1105) .

[9] Sze, Vivienne & Chen, Yu-Hsin & Yang, Tien-Ju & Emer, Joel. (2017). Efficient Processing of Deep Neural Networks: A Tutorial and Survey. Proceedings of the IEEE. 105. 10.1109/JPROC.2017.2761740.

[10] https://builtin.com/data-science/transfer-learning

[11] Lecun, Yann & Bengio, Y.. (1995). Convolutional Networks for Images, Speech, and Time-Series. The Handbook of Brain Theory and Neural Networks.

[12] Philipp, George & Song, Dawn & Carbonell, Jaime. (2017). Gradients explode - Deep Networks are shallow - ResNet explained.

Appendix

Github link for code is provided to reproduce visualizations and statistical analysis from the study.

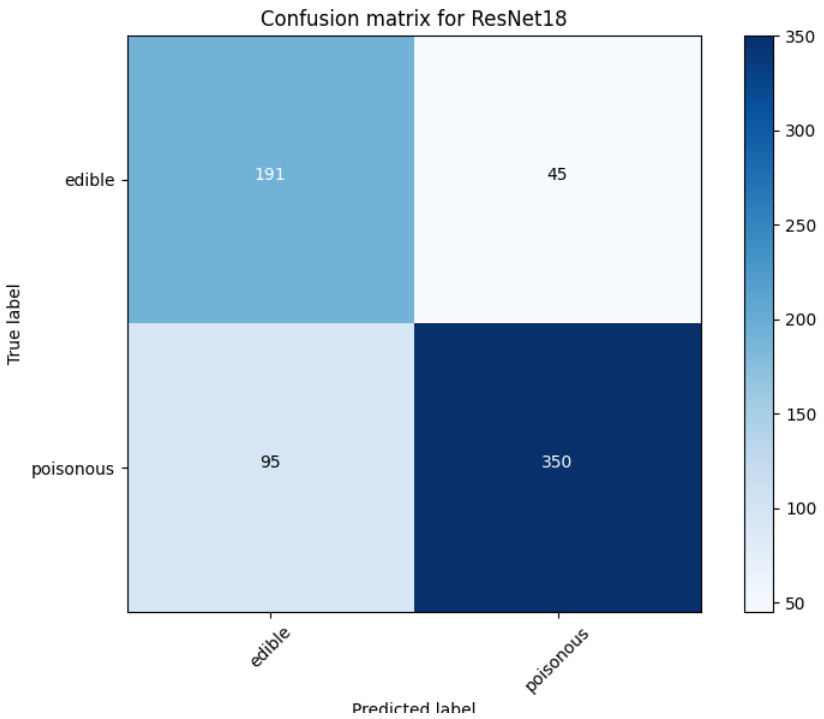
Supplemental Materials

Supplemental Table 1[3]. Loss function comparison table		
	CrossEntropy	Focal Loss
Accuracy	0.794	0.398
Precision	0.886	0.760
Recall	0.787	0.703

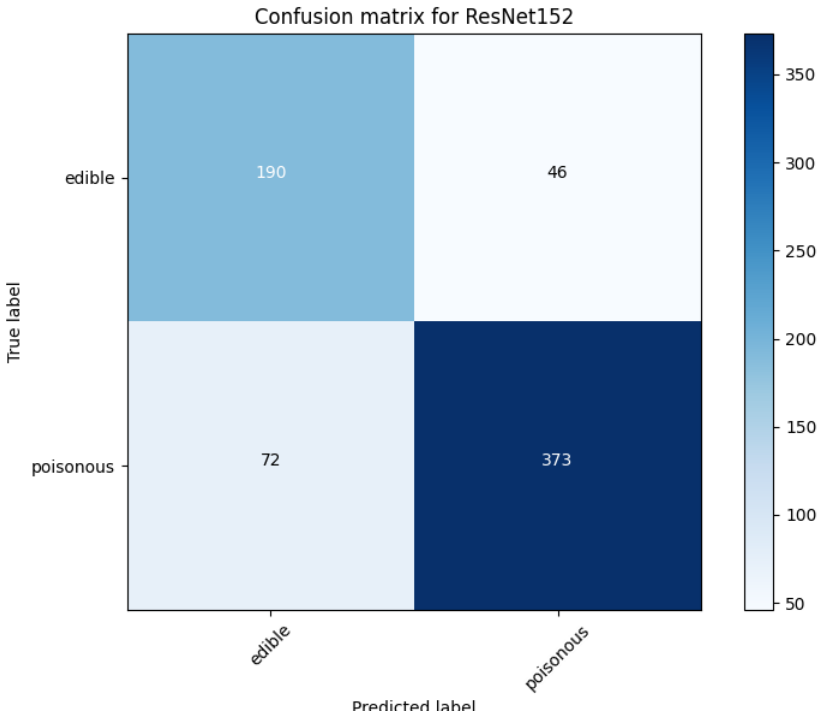
Supplemental Table 2[4]. Algorithm comparison table		
	SGD	Adam
Accuracy	0.794	0.595
Precision	0.886	0.827
Recall	0.787	0.739

[3] The results in Table S1 are from the ResNet18 model with different loss functions for classifying the 4 classes of fungi. It is calculated on the test dataset.

[4] The results in Table S2 are from the ResNet18 model with different optimization algorithms for classifying the 4 classes of fungi. It is calculated on the test dataset.

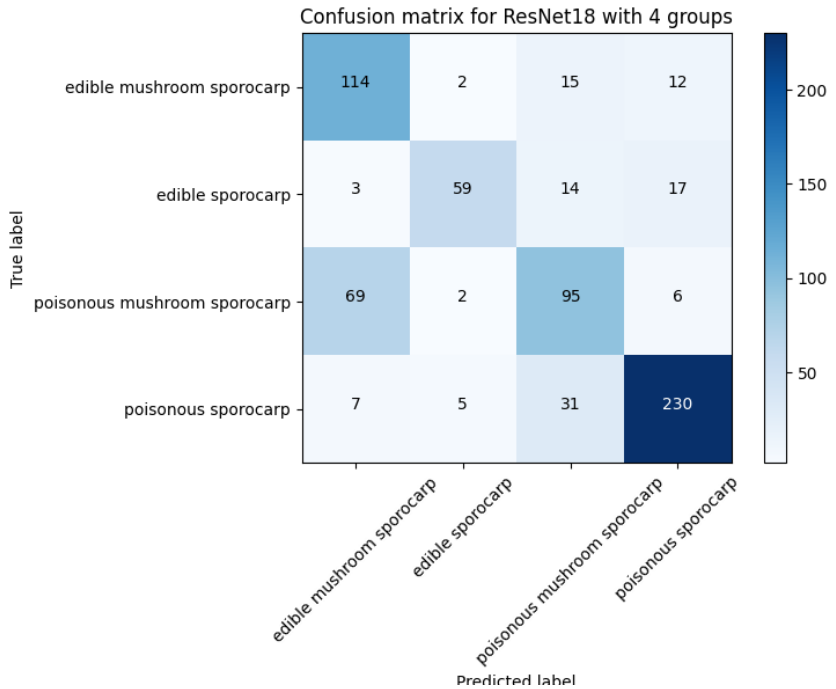


Supplemental Figure 1. Confusion matrix for Resnet18 classifying 2 groups[5]

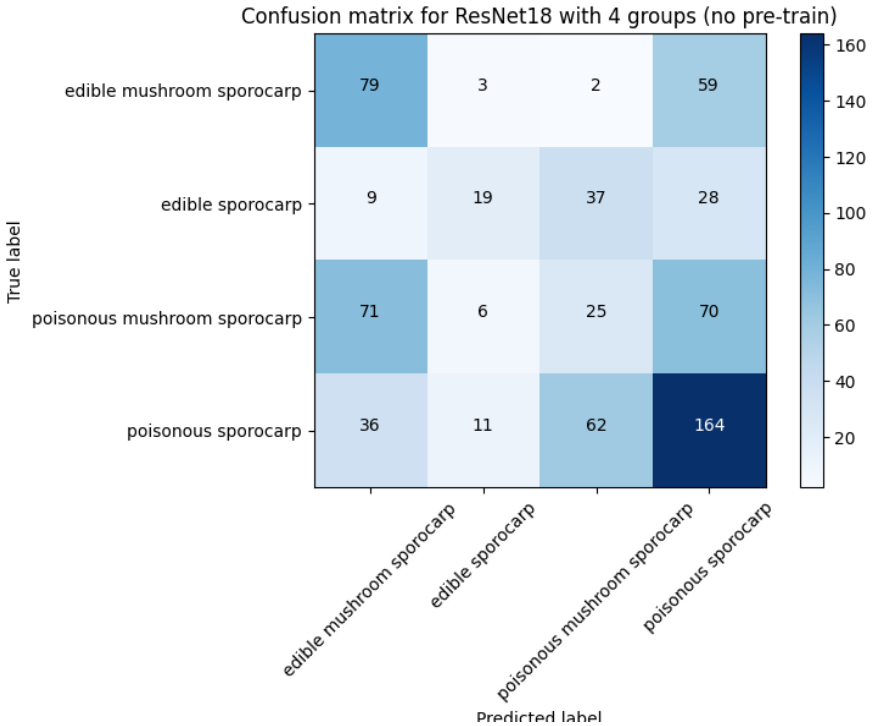


Supplemental Figure 2. Confusion matrix for Resnet152 classifying 2 groups[6]

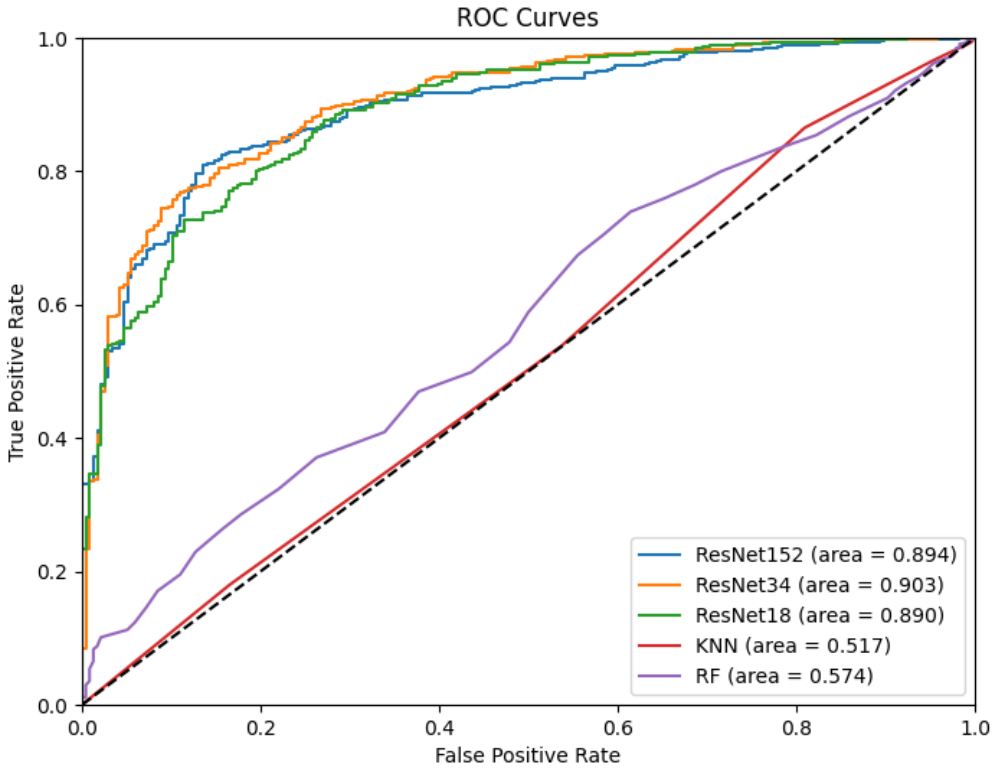
[5] For the ResNet18 confusion matrix, TN is191, FP is 45, FN is 95, and TP is 350, based on the test set.
[6] For the ResNet152 confusion matrix, TN is190, FP is 46, FN is 72, and TP is 373, based on the test set.



Supplemental Figure 3. Confusion matrix for Resnet18 classifying 4 groups.



Supplemental Figure 4. Confusion matrix for Resnet18 classifying 4 groups without pre-trained model weights, showing inferior performance compared with pre-trained model weights.



Supplemental Figure 5. ROC comparison of 5 models, showing that ResNets are uniformly better than other machine learning models in this project. Among three ResNets, the intermediate complex ResNet34 has slightly better performance than the others