Fresh and Spoiled Food Image Classification Using Transfer Learning with EfficientNetB0

Graidvel Vladen Gomar Gama

LB01 – Deep Learning

***Abstract*— This article provides an overview of advanced technologies for determining food spoilage, with a focus on fruits, vegetables, bread, and diary. Though traditional inspection remains fundamental, technologies such as Machine Deep Learning or Computer Vision offer potential real-time monitoring and analysis. This study covers a practical implementation of a Deep Learning technology, using image-based classification system to distinguish fresh from spoiled food using transfer learning with EfficientNetB0. This model uses a dataset composed of 2 perishable foods, each containing 4 categories, such as fruits, vegetables, bread, and dairy, and these data are processed into binary labels. The model was trained in transfer learning, data augmentation, and partial fine-tuning. Results showed that the model can achieve reliable accuracy in determining food freshness despite limited training dataset. This work demonstrates the feasibility of lightweight deep learning models in a real-world food quality inspection.**

Keywords*—Perishable Foods; Deep Learning; Computer Vision

I. INTRODUCTION

1.1. Background

Food spoilage is the result of changes of physicochemical properties of the food product and the microbial actions of microorganisms. Microbial spoilage accounts for 25% of crops globally, resulting in inadequate shape, appearance, and value. Food spoilage can be reduced or eliminated through efficient cold chains, though some microorganisms are still active under refrigeration temperatures and can induce spoilage, such as meats or their derivatives.

With the rapid advancement of Computer Vision, Deep Learning, etc., machine learning models have become more capable of detecting and utilizing visual cues, in this case, color change, texture degradation, and mold growth, all that indicate spoilage. CNN (Convolutional Neural Networks), particularly modern model architecture like EfficientNet, offers great performance with image classifications even with a lack of training data.

1.2. Problems

Despite modern advancements in food detection technologies, most existing systems still rely on expensive sensors, require controlled environments, or need technical expertise to operate, giving no simple, low-cost, image-based solutions that allows classification whether food is fresh or spoiled by only using a standard camera. With this statement, This project mainly addresses the question:

- Can a deep learning model classify fresh and spoiled food images effectively and accurately only using visual features?

1.3. Objectives

The objectives focused in this project are:

- Developing a Deep Learning image classification model that is capable to distinguish fresh and spoiled food items using image data.
- Preprocessing and refining a dataset consisting images of fresh and spoiled foods across multiple food times.
- Designing and training a CNN-based model, focusing on EfficientNet as the feature extractor.
- Evaluating the model performance using accuracy, loss, and other validation scores.
- Preparing a deployable system prototype with this model, enabling users to upload images and get their freshness prediction.

1.4. Significance

This project contributes to practical and academic values, such as:

- Public Health: reduces foodborne illness risks by early food spoilage detection.
- Waste Reduction: Avoids unnecessary food disposal by misjudgments.
- Low-Cost AI Application: A Demonstration of computer vision that can be used in real-world problems, such as a food spoilage detector.
- Educational Value: Provides real-time experience in dataset preparation, transfer learning, model fine tuning, and deployment.

II. RELATED WORK

Food spoilage detection has been explored in previous research using computer vision and deep learning. Research by Helen Bongard et al. concluded that AI enables the early detection of spoilages. Besides detecting spoilages early, AI also optimizes inventory management and significantly reduces waste through more accurate demate forecasting. Their research shows that companies like IKEA, Shelf Engine and Afresh has already implemented these Ais, demonstrating significant food waste reduction per store and preventing thousands of CO_2 emissions.

https://www.sciencedirect.com/science/article/pii/S2666154325002662

Other studies focus on freshness classification of fruits and vegetables using CNNs. Research by Mukhridddin et al. shows that they have built a CNN model using YOLO models to classify between fresh fruits, rotten fruits, fresh vegetables, and rotten vegetables. Using a dataset containing 12000 images, they have achieved high precisions in classifying these fruits and vegetables.

https://www.mdpi.com/1424-8220/22/21/8192

Some researchers apply transfer learning with pretrained models EfficientNet for food image classification. Additionally, EfficientNet has been shown to outperform traditional CNN architectures in image recognition tasks due to its compound scaling strategy. A research by Mingxing Tan and Quoc V.Le has tested all 8 EfficientNet models and recorded their accuracies. Comparing to other models, it is proven that EfficientNet excels other models they've tested.

Compared to these works, our study focuses specifically on binary classification (fresh vs. spoiled) across multiple food categories (bread, dairy, fruits, vegetables). We also use EfficientNetB0 with partial fine-tuning to improve generalization on a small dataset.

III. METHODOLOGY

The approach to this research is to create a model that can distinguish the difference between spoiled and fresh food. The creation process is as follows:

3.1. Datasets

The dataset used to train this model consists of approximately 1900 different images from Kaggle. These images are classified into eight classes:

- fresh_bread
- fresh_diary
- fresh_fruits
- fresh_vegetables
- Spoiled_bread
- Spoiled_diary
- Spoiled_fruits
- Spoiled_vegetables

Then, these images were mapped into 2 labels:

- Fresh (0) — All classes starting with "fresh_"
- Spoiled (1) — All classes starting with "spoiled_"

Each class was trimmed to exactly 100 images each, achieving balanced training, helping with computation time and prevent overfitting.

3.2. Data Preprocessing

The dataset then is preprocessed for model training. The steps applied for this data preprocessing are as follows:

1. Image Load

Images from the dataset were loaded using "image_dataset_from_directory" with 224 * 224 resolution to match EfficientNet's input size.

2. Label Conversion

Each image is given a label based on their folder origin. Therefore, these images were converted to binary labels by checking whether their class (folder origin) starts with "fresh" or "spoiled."

3. Normalization and EfficientNet Preprocessing

Using a built-in preprocess "preprocess_input" from EfficientNet's library, the images were rescaled into the range expected by the model.

4. Dataset splitting

The dataset is split into these proportions:

- 70% Training
- 15% Validation
- 15% Testing

The split used "train_test_split()" from "sklearn.model_selection" to maintain a balanced proportions within the dataset.

5. Data Augmentation

A data augmentation was used to increase the data's diversity. The augmentations used are as follows:

- Random Horizontal Flip
- Random Rotation
- Random Zoom
- Random Contrast

These augmentations help the model generalize better.

3.3. Model Architecture

The model use is based on EfficientNetB0, a pretrained CNN that's known for strong performance and optimized parameter efficiency. This model was further fine-tuned partially by only setting the last 20 layers as trainable. It is further enhanced by adding more layers, such as a Global Average Pooling layer (GlobalAveragePooling2D()), a dropout layer (Dropout()), and a Dense layer (Dense()) with

sigmoid activation. This enables binary classifications.

3.4. Training Setup

The setups used to train the model were as such:

- Optimizer: Adam with learning rate = 0.0001
- Loss Function: Binary Cross-Entropy
- Batch Size: 16
- Epochs: Up to 20
- Early Stopping: 5 times of validation loss monitoring
- Model Checkpoint: used to save the best model based on validation loss

These settings ensure stable training and prevent overfitting.

3.5. Evaluation Metrics

The metrics used are the following:

- Accuracy — Model classification performance
- Loss — Binary Cross-Entropy across epochs
- Accuracy — Classification accuracy
- Recall — True positive rate over (true positive rate + false negative rate)
- Precision — True positive rate over (true positive rate + false positive rate)
- F1 score — Harmonic mean between precision and recall

These values will represent the total performance of this model.

IV. IMPLEMENTATION AND RESULTS

4.1. System Implementation

The model was developed, trained, and tested locally using the following environment:

- Processor: 11th Gen Intel® Core™ i7-1165G7 @ 2.80GHz
- RAM: 16 GB
- Operating System: Windows 11
- Programming Language: Python 3.11
- Frameworks and Libraries:
  - Tensorflow/Keras
  - NumPy
  - OpenCV
  - Scikit-learn
  - Streamlit

4.2. Dataset Description

The dataset contained 800 images, evenly divided into 400 “Fresh” classified images and 400 “Spoiled” classified images. The dataset was stored in 8 folders and loaded using “image_dataset_from_directory” from “tensorflow.keras.utils.”

4.3. Data Preprocessing

Several preprocessing steps were conducted to the dataset. The steps were as follows:

1. Image Normalization
   Images from the dataset were preprocessed using “preprocess_input” from “EfficientNet” to ensure compatibility with ImageNet-trained weights.
2. Label Conversion
   Since the dataset contains 8 different folders, the images were assigned binary labels, labeled 0 for images from “fresh_” and 1 for images from “spoiled_.”
3. Train-Validation-Test Split
   The dataset was split using “train_test_split” from the library “sklearn.model_selection” into 3 groups:
   - 70% Training
   - 15% Validation
   - 15% Testing
4. Data Augmentation
   The dataset is augmented with a “Sequential” block to improve

generalization. The augmentation layers used are as such:

- Random Horizontal Flip
- 0.2 Random Rotation
- 0.2 Random Zoom
- 0.1 Random Contrast

4.4. Model Implementation

The model was built using EfficientNetB0 as the feature extractor. The model is built as such:

- Pretrained on ImageNet
- Top layers removed
- Last 20 layers set as trainable for fine-tuning
- Global Average Pooling
- 0.2 Dropout
- Dense layer with sigmoid activation

4.5. Model Training

The model was trained using the following configurations:

- Optimizer: Adam
- Learning Rate: 0.0001
- Loss Function: Binary Cross-Entropy
- Batch Size: 16
- Epochs: 20

Two callbacks were used to improve training efficiency:

- EarlyStopping
  5 Patience and keeping best weights
- ModelCheckpoint
  Saving the best model as "best_model.h5"

4.6. Results

The model was trained and stopped at Epoch 14 due to EarlyStopping callback. The results are shown below.

4.6.1. Training Curves

The visualization between training and validation accuracy is shown below:

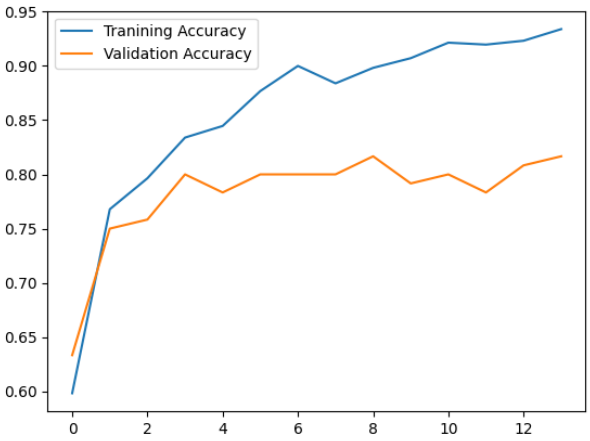

Figure 6.1 Accuracy Graph

With final Training Accuracy at 0.9339 and Validation Accuracy at 0.8167.

The visualization between the training and validation loss is shown below:

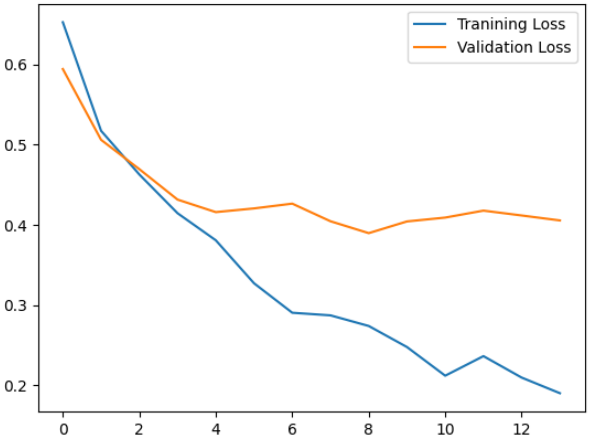

Figure 6.2 Loss Graph

With final Training Loss at 0.1899 and Validation Loss at 0.4054.

4.6.2. Evaluation Metrics

The metric scoring used are Accuracy, Precision, Recall, and F1. The computation results of these scores from the model are as follows:

- Accuracy : 0.5834
- Precision : 0.5781
- Recall : 0.6167
- F1 Score : 0.5968

4.6.3. Prediction Samples

The model is implemented to streamlit website. Below are some examples on the model prediction.

V. DISCUSSION AND LIMITATIONS

5.1. Discussion

The experimentation results in the EfficientNet based classifier model capable of learning visual differences between fresh and spoiled foods, but its overall performance remains moderate. The model achieved approximately 58% accuracy and F1 score of 0.59, showing that the model is not consistently reliable despite its ability to identify spoilage characteristics. The cause of this varies, including but not limited to:

- Subtle visual differences between images and/or classes
- High intra-class variation dataset
- Small sized dataset size
- Various sample quality

Despite these limitations, the model still has the potential to serve as a food spoilage detection tool, mainly on simple environments, which is better than random guessing. On another note, the model proves that EfficientNet has the capability to extract relevant features despite the small dataset.

5.2. Limitations

The limitations mentioned above will be explained more thoroughly here. The limitations are as follows:

1. Limited Dataset Size
   While the dataset contains 800 images, this dataset also contains 8 different food categories (e.g. bread, fruits, vegetables, diary) with each fresh and spoiled type, which means that each category only contributes 100 images for the dataset, while EfficientNet requires thousands of images per class/categories to prevent underfitting.

2. High Variations Across Dataset
   With 8 categories used, this significantly increases the learning difficulty for the model. The model is forced to generalize spoilage pattern across the dataset instead of specific type of food spoilage, significantly decreasing model classification accuracy.
3. Environmental Noise inside Images
   Besides the class differences, each images also highly varies in terms of lightning conditions, camera distance, background, and color temperature, making noises that also increases learning difficulty.
4. Overfitting Risk
   Based on figure 6.2, the training loss decreases faster than validation loss, indicating that the model learns the dataset very well but struggles with new data.
5. Limited Computational Resources
   Originally, the dataset contains approximately 1900 images spread across these 8 classes, varying from 200 to 500 images per class. Due to computational limitations, the dataset is trimmed randomly down to 100 each class. Furthermore, other settings are also adjusted, such as lower batch size or fewer epochs to save computer resources but also affect the final performance and its ability to tune hyperparameters effectively.

5.3. Summary

In summary, the model still requires significant improvement in dataset size, its consistency, and its training configuration to reach deployment standards, But the current performance still aligns with the dataset constraints and experimental environment.

VI. CONCLUSION AND FUTURE WORK

This project has presented an image-based food freshness classification model using deep learning techniques. A CNN based by the Efficient-B0 architecture was implemented to the model, designed to distinguish fresh and spoiled food items, including bread, fruits, vegetables, and dairy products. The model was trained using a dataset consisting of 800 images and is evaluated by using standard performance metrics, such as accuracy, precision, recall, and F1-score.

Experimentation shows that the model to learn decent virtual features related to food spoilage, achieving moderate classification performance. Aside from its feasibility of using transfer learning for food freshness detection, it also highlight challenges associated with limited dataset and high variability across food categories. In overall, the main objective of this project were met by successfully developing, training, evaluating, and deploying a decent functional deep learning model.

Several improvements can be made in the future work to enhance the performance of the current system. Mainly, the dataset should be expanded with more images and higher variability to significantly improve model generation. Second, experimentation with higher epochs from better computational powers is highly recommended to improve the model and prevent it from overfitting.

Additionally, the model can be further developed not only from a streamlit page, but could also be developed inside mobile or cloud platforms for real-world usages as week as integrating AI techniques such as Grad-CAM to visualize detections. These enhancements to the model would contribute highly to making the system more robust, effective, and practical for real-world food quality monitoring.

REFERENCES

Mukhiddinov, M., Muminov, A., & Cho, J. (2022). Improved classification approach for fruits and vegetables freshness based on deep learning. *Sensors*, *22*(21). https://doi.org/10.3390/s22218192

Tan, M., & Le, Q. V. (2019). *EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks*.

Onyeaka, H., Akinsemolu, A., Miri, T., Nnaji, N. D., Duan, K., Pang, G., Tamasiga, P., Khalid, S., Al-Sharify, Z. T., & Ugwa, C. (2022). Artificial Intelligence in food system: Innovative Approach to minimizing food spoilage and food waste. *Journal of Agriculture and Food Research*, *21*, 101895. https://doi.org/10.1016/j.jafr.2025.101895