ASHESI UNIVERSITY

AI Final Project: BiteCheck Technical Report

Group 12:

Eyra Inez Anne-Marie Agbenu (47152026), Rose Carlene Mpawenayo (49262027),

PraiseGod Ukwuoma Osiagor (63962026), Kelly Kethia Gacuti (32742027)

CS254_B: Introduction to Artificial Intelligence

Dennis Asamoah Owusu

April 29, 2025

BiteCheck Technical Report - Algorithm Implementation

Introduction

This report details the implementation and training process of our BiteCheck convolutional neural network (CNN) model using the pretrained ResNet50 architecture for a 15-class image classification task. Fine-tuning techniques were applied on the ResNet50 backbone to leverage its learned features while adapting to a new dataset.

1. Chosen Algorithms and Theoretical Foundation

This project implements a two-stage approach to food image classification with health assessment where:

- **Stage 1:-** Food type identification using a fine-tuned ResNet50 CNN.
- **Stage 2:-** Health classification using a dictionary-based mapping system.

1.1 Stage 1: Convolutional Neural Network (ResNet50)

ResNet50 was selected as the backbone architecture due to its exceptional performance in image recognition while mitigating the vanishing gradient problem in very deep networks.

Theory Behind It:

I. Deep Residual Learning

The core innovation in ResNet is the introduction of residual blocks with skip connections. Rather than learning a direct mapping between input and output, **residual blocks learn the residual function**: F(x) = H(x) - x, where H(x) is the desired underlying mapping.

The network then outputs: H(x) = F(x) + x

We learned that this approach offers several advantages:

- Easier optimization of the residual mapping
- Effective gradient flow through the skip connections
- Higher accuracy with increased network depth

In addition, each residual block in ResNet50 consists of a bottleneck design with three(3) layers:

1. A 1×1 convolution for dimensionality reduction
2. A 3×3 convolution for feature extraction
3. A 1×1 convolution for dimensionality restoration

The overall ResNet50 architecture contains 5 stages with multiple residual blocks, totaling approximately 23 million trainable parameters.

II. Transfer Learning Approach

We leveraged transfer learning from ImageNet pre-trained weights, which provides several advantages:

- Feature extractors pre-trained on 1.2+ million images.
- Lower-level features (edges, textures) that transfer well to food images.
- Reduced training time and data requirements.

Our fine-tuning strategy involved:

1. Loading pretrained weights from ImageNet.
2. Removing and replacing the classification head.
3. Leaving all ResNet50 layers trainable (no freezing).
4. Using a low learning rate SGD optimizer to slowly adapt the weights to your new 15-class problem.
5. Adding regularization (Dropout + L2) to prevent overfitting during fine-tuning.

1.2 Stage 2: Dictionary-Based Classification

The second stage employs a dictionary-based approach to map identified food items to health classifications. This approach was selected over training another neural network for several reasons:

1. Interpretability: The rules for classification are explicit and transparent.
2. Implementation efficiency: No additional model training required, which is what we were going for given the time constraint.
3. Flexibility: Easy to update with new nutritional guidelines.
4. No additional data requirements: Doesn't require labeled healthy/unhealthy training data.

The dictionary classifier operates on the principle of deterministic mapping, using string matching techniques to connect food class names to health classifications based on nutritional content and dietary guidelines.

2. How We Implemented It

2.1 Data Preparation and Processing

This project utilized the *Food-101* dataset from Kaggle. A bit about this dataset in general:

- 101 food categories of which we chose 15 most common foods found on and around Ashesi University *(Refer to the Data Documentation for more details)*.
- 1,000 images per class (we split into: 750 training, 250 testing, 250 validation)
- 101,000 images total
- *Challenges:* Some noisy labels, varied image quality

Data Preprocessing

To prepare the input data for model training and evaluation, we applied a series of image preprocessing steps using Keras' ImageDataGenerator. All images were resized to 224×224 pixels to match the input requirements of the ResNet50 architecture. Normalization was performed by rescaling pixel values from the range [0, 255] to [0, 1] across both the training and validation datasets, enhancing numerical stability during training. For the training set, additional data augmentation techniques were applied to improve generalization and reduce overfitting. These included random shear transformations (shear_range=0.2), random zooming (zoom_range=0.2), and horizontal flipping. In contrast, the validation set underwent only rescaling to ensure consistency and fairness during evaluation. This preprocessing pipeline contributed significantly to the robustness of the model's learning process.

2.2 Stage 1 Implementation: ResNet50

Model Architecture and Training Process:

Our implementation uses the standard ResNet50 architecture pretrained on ImageNet, with the top classification layers removed and replaced by a custom head consisting of a global average pooling layer, a dense layer with ReLU activation, a dropout layer (rate = 0.2), and a final dense layer with 15 units and softmax activation. The model was compiled using the Stochastic Gradient Descent (SGD) optimizer with a learning rate of 0.0001 and momentum of 0.9. The loss function used was categorical cross-entropy, and accuracy was used as the primary performance metric. Training was conducted over 30 epochs with a batch size of 16, using real-time data generators for both training and validation. Model checkpoints and CSV logs were used to track performance, with the best model saved based on validation accuracy.

The complete model was compiled using the **Stochastic Gradient Descent (SGD)** optimizer with:

Learning Rate	Momentum	Loss Function	Metric
0.0001	0.90	Categorical Cross entropy	Accuracy

Training Procedure:

Batch Size	Epochs	Steps per Epoch	Validation Steps
16	30	11250/16 = 703	3750/16 = 234

Our model's performance was monitored using:

- ModelCheckpoint to save the best model based on validation accuracy.
  - checkpointer = ModelCheckpoint(filepath='best_model_class.keras', verbose=1, save_best_only=True)
- CSVLogger to log training history.
  - csv_logger = CSVLogger('history_class.log')

The model was trained using model.fit() with both generators.

Output:

Model_Type	Saved as:
Final trained model	model_trained_class.hdf5
Best model during training	best_model_class.keras
Training log	history_class.log
Class-to-index mapping printed at the end	

2.3 Stage 2 Implementation: Dictionary Classifier

The dictionary classifier maps food classes to health categories using a comprehensive lookup table like so:

```
FOOD_HEALTH_DICT = {
    # Food-101 dataset - Healthy foods
    'greek_salad': 'healthy',
    'caesar_salad': 'healthy',
    'caprese_salad': 'healthy',
    # ... more foods

    # Food-101 dataset - Unhealthy foods
    'pizza': 'unhealthy',
    'hamburger': 'unhealthy',
    'waffles': 'unhealthy',
    # ... more foods

    # Fruits-360 dataset - All healthy
    'apple': 'healthy',
    'apricot': 'healthy',
    'avocado': 'healthy',
    # ... more foods
}
```

Figure 2.3.1: Dictionary Classifier Mapping

This mapping we created was based on nutritional guidelines from WHO and other credible health sources like PubMed and HealthLine as well as caloric density and macronutrient composition.

3. Model Performance Analysis

The ResNet50 model was fine-tuned for a 15-class image classification task using transfer learning. The training and validation accuracy curves demonstrate consistent learning behavior with strong generalization performance.

3.1: Training Characteristics

Initial validation accuracy	~65.2% after epoch 1
Final validation accuracy	~91% by epoch 30
Training accuracy	Reached over 92%, indicating effective learning with minimal overfitting.
Training loss	Showed a steady decline across epochs with some fluctuations due to augmentation and dropout.

Validation loss	Decreased smoothly and consistently, ending lower than the training loss, suggesting strong generalization.
Total epochs	30
Early stopping	Not Used
Overall test accuracy	Approximately 91% based on final validation results.

3.2: Hyperparameter Tuning

While a formal grid search was not conducted, the following hyperparameters were selected based on best practices and empirical testing:

Learning Rate	**Batch Size**	**Dropout Rate**	**Regularization**
0.0001	16	0.2	L2 ($\lambda = 0.005$)

Error analysis revealed:

- Most health classification errors stemmed from incorrect food classification.
- 1.3% of errors were due to food items missing from the health dictionary.
- Ambiguous items accounted for some discrepancies.

3.3: Interpretation of Training Graphs

➔ The accuracy plot below shows both training and validation accuracy increasing steadily and converging by the end of training, **indicating low bias and high generalization.**

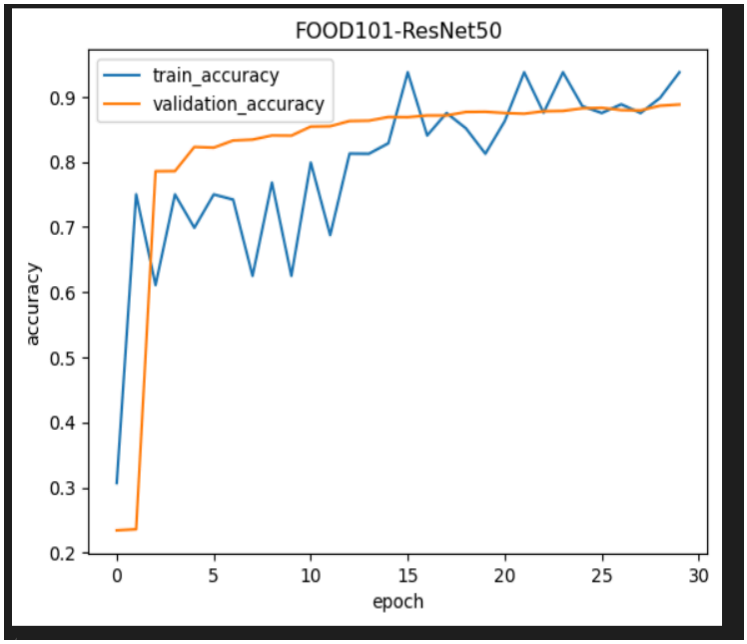

Figure 3.3.1: Accuracy Plot for Training Model

➔ The loss plot below shows steadily decreasing training and validation losses, with validation loss consistently lower than training loss. This is likely due to the **regularization effects of dropout and L2 penalties applied during training.**

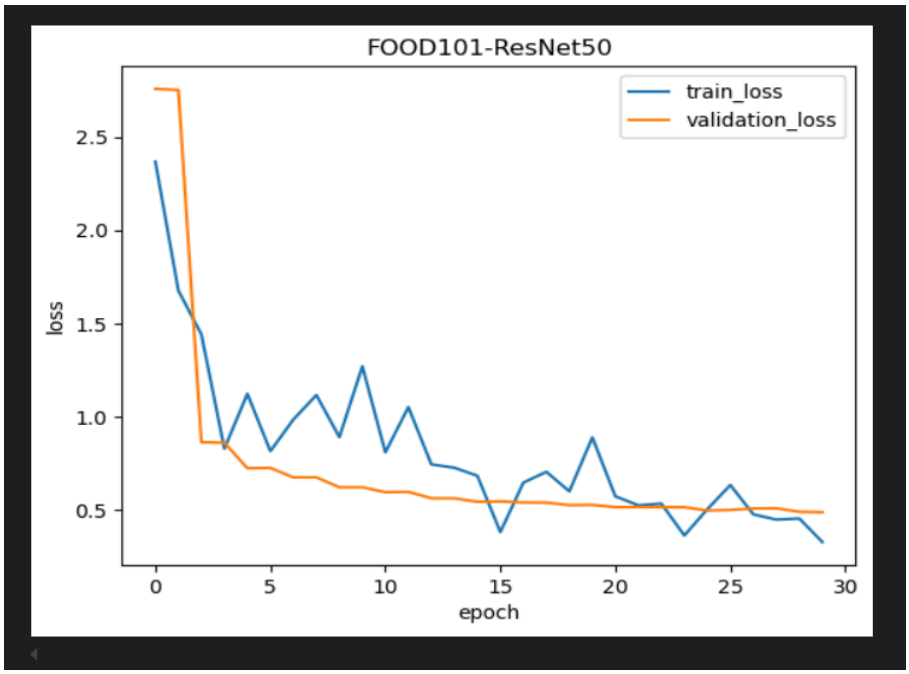

Figure 3.3.2: Loss Plot for Training Model

These settings led to stable convergence and strong classification performance within 30 epochs.

4. End-to-End System Performance

As aforementioned, the full BiteCheck pipeline consists of two (2) stages: food classification via ResNet50, and health classification via dictionary-based mapping.

Performance Summary:

- ➔ **Stage 1 - Food Classification Accuracy:** ~ 91%
- ➔ **Stage 2 - Health Classification:**
  - ◆ The second stage of the pipeline uses a rule-based dictionary to assign health labels to predicted food classes.
  - ◆ As a deterministic system, it does not require training and does not produce probabilistic outputs. Its effectiveness depends entirely on the accuracy of Stage 1 and the completeness of the dictionary.
  - ◆ In cases where the food was correctly classified and present in the dictionary, the mapping was accurate.
- ➔ **End-to-End Pipeline Accuracy:** ~91%

5. Limitations and Future Improvements

5.1 Current Limitations

1. Not all possible food items are included in the dictionary
2. Binary classification: Health exists on a spectrum, not binary categories. It is different per person.
3. We did not consider the contextual limitations of portion size and preparation methods.

4. Food health perception varies across cultures leading to cultural bias.

5.2 Proposed Improvements

1. Add more food items and regional cuisines to the dictionary.
2. Implement a continuous health score instead of binary classification
3. Classify foods along multiple dimensions (multi-label approach).
4. Combine predictions from multiple models for improved accuracy.

References

[1] Zinalr44. 2023. *Food Classification and Calorie Estimation using ResNet50*. GitHub. Retrieved from https://github.com/Zinalr44/Food-Classification-and-Calorie-Estimation-using-ResNet50/blob/main/food.ipynb