
**Image-Based Nutritional Estimation for Comprehensive Daily Intake Monitoring – A Novel Approach to Include Singaporean Street Foods and Nutrition Labels**

National University of Singapore
BT5151 Advanced Analytics and Machine Learning
21/4/2024

Group Members:

| | |
|---|---|
| Anshumaan Phukan | E1148782 |
| Jared Cheang | E0560192 |
| Roy Yeo | E1148761 |
| Yi Kai Lim | E1148727 |
| Yuhe Xin | E1148671 |

**Executive Summary**
This report details the development and implications of a nutritional analysis tool designed for the Singapore market. By leveraging machine learning technologies, this solution aims to provide and record real-time, accurate nutritional data for the consumption of food through processing food images.By incorporating both made-to-order local dishes and prepackaged foods, our solution aligns with Singapore's Healthier SG initiative and addresses the growing discrepancy between perceived and actual dietary habits as highlighted in recent nutritional surveys. Our solution will suggest the daily nutrition intake based on user demographic, such as Age, Weight, Gender and Level of Activity, and indicates the percentage of calories consumed from the food recorded by users.

## 1. Introduction

### 1.1. Background and Context

In recent years, the Singaporean government has intensified its efforts to promote a health-conscious lifestyle through various national initiatives, including Healthier SG. Despite these efforts, a gap remains between perceived healthy eating and actual dietary habits. The 2022 National Nutrition Health Survey indicates an increase in daily calorie intake among adults, from 2,360 kcal in 2019 to 2,410 kcal in 2022 and a rising percentage of individuals exceeding their recommended calorie intake, from 55% to 61% of the population. Even top tech firms like Apple have yet to create smartwatches that can directly measure nutrient intake, which highlights the complexity and the need for innovative tracking solutions, particularly for Singapore's vibrant variety of freshly prepared foods. As such, this underscores the need for enhanced dietary monitoring and management tools.

Food consumption typically falls into two categories:

1. Freshly prepared food – e.g. fish soup, laksa, satay from a hawker center
2. Packaged food – e.g. milk carton, cereal

### 1.2. Problem Statement

The current methods for tracking nutritional intake are inadequate for several reasons. Firstly, freshly prepared foods often lack nutritional labels, making it challenging to accurately assess their nutritional content. Additionally, these methods are incomplete because food consumption typically includes a combination of freshly prepared items and prepackaged foods, which do have nutritional labels. Therefore, a more complete nutrient intake solution is needed to effectively monitor nutritional intake. This gap not only persists in local hawker centers but also in home-cooked and restaurant meals. Without accessible and reliable nutritional data, consumers face challenges in making informed dietary choices, potentially leading to health risks associated with over or under-eating.

### 1.3. Project Objectives

The primary objective of this project is to develop an application that uses advanced image recognition and machine learning technologies to estimate the calorie content of food from images. This tool will cater to both key categories of food consumption:

- Freshly Prepared Food: Utilize image recognition to analyze and estimate the calorie and nutritional content of meals typically found in local eateries. This is our major function of the application, which solves the current challenge of tracking calories of fresh local food consumption.
- Packaged Food: Utilize the OCR pipeline and customized object detection algorithm to extract and interpret nutritional labels from packaged products. While it does not involve a training process, OCR fits in the gap of recording packaged food to aid in comprehensive dietary tracking.

### 1.4. Value Proposition

Our proposed solution stands to benefit businesses and consumers alike by:

- **Enhancing Nutritional Transparency**: Providing immediate nutritional insights for a wide array of food items, supporting individuals in achieving their dietary goals aligned with national health objectives.
- **Fostering Health-Conscious Decisions**: Empowering users with data to adjust their eating habits, potentially reducing healthcare costs related to diet-associated ailments.

- **Enriching Culinary Experience**: Offering educational insights into the nutritional content of local Singaporean cuisines, thus enhancing the eating experience for tourists and residents.

## 2. Data Acquisition & Preprocessing

### 2.1. Data Collection

The foundation of our solution involves compiling a robust dataset, encompassing a wide variety of food items typically found in Singapore. We sourced images from existing databases (e.g., FoodSG-233), and selected 20 different local food categories consisting of 18,000 (18K) images in total. We also acquired nutrition label images from online for our OCR text extraction feature. Additionally, we integrated structured nutritional data, which we have obtained from the Health Promotion Board's Energy & Nutrient Composition of Food database to provide a comprehensive calorie and nutrient profile for local food items.

### 2.2. Data Integration

Combining structured nutritional information with unstructured image data involves aligning disparate data sources. We will align our prediction labels with corresponding nutrition values stored in the separate dataset. This integration was essential for enriching our dataset, providing a dual-layer of information (visual and quantitative) that enhances the predictive capabilities of our models.

### 2.3. Data Cleaning and Transformation

Initial processing steps included image resizing and normalization to ensure consistency across the dataset. This standardization is crucial for effective model training and performance. In our nutrition label extractor, we begin by preprocessing the image to remove unimportant regions before passing the processed image into the OCR for text extraction.

Do note our project dataset consists of 18,000 (18K) images in total with a slight imbalance as seen from the image, in which we balance the classes out by obtaining an equal number of samples across all categories. Due to the massive nature of the dataset, we have sub-sampled the dataset for our model testing while ensuring equal class representation. In particular, we sampled 10% of the dataset yielding 1.8K samples overall, corresponding to an equal class representation of 90 samples across the main 20 food categories. We believe an equal class representation is sensible as we treat all food categories as having equal importance to the final classification. Also, to enforce a balanced dataset, we have stratified the labels for equal representation among all food classes throughout the training, testing, and validation datasets.

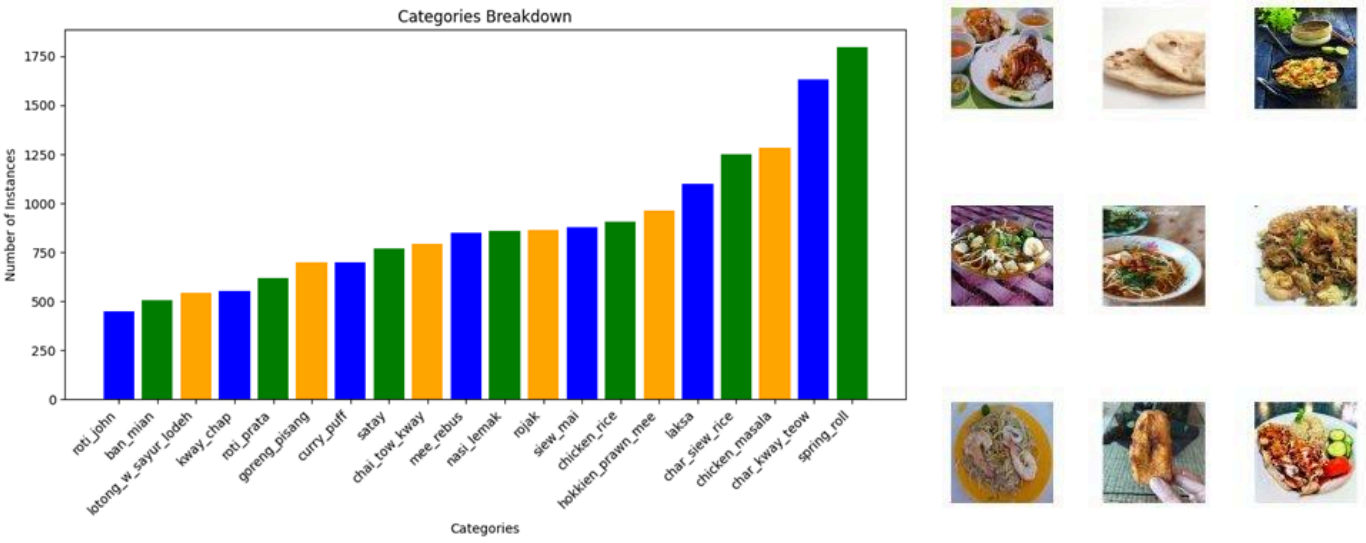



Fig 1,2: Breakdown of food categories and a snapshot visualization of food image samples

## 3. Machine Learning Models & Training Procedures

We investigate various machine learning models for image recognition tasks to select the best performing model, where our desired metric for this image classification task is based on accuracy. Accuracy prioritises overall correctness in this classification project where it measures the proportion of correctly classified instances out of the total instances evaluated. Additionally, after the preliminary data processing, the datasets contain a balanced representation of different food types, which makes accuracy a meaningful metric for evaluating the performance of the classification model in our food nutrition management solution.

Meanwhile, to improve our accuracy of the existing OCR pipeline, we design an algorithm to identify, extract and enlarge the nutrition label segment from the entirety of the image input as part of our integrated training process. Detailed processes are described under the OCR section.

### 3.1. Selected Machine Learning Models & Algorithms

Due to the nature of our project where multiple models were experimented and evaluated, we perform detailed documentation on the best performing model while giving a quick overview on the other models, with greater details placed in the Appendix.

#### 3.1.1. VGG16

The VGG16 model is a convolutional neural network (CNN) architecture, and is primarily used for image recognition tasks, such as identifying objects within an image. It has been widely adopted for various applications in computer vision, including image classification, image segmentation, and feature extraction.

Advantage & Disadvantage of the Model

VGG16 is known for its high accuracy in image recognition tasks. VGG16 has been pre-trained on the ImageNet dataset, which includes a range of object categories. This pre-training can be leveraged through transfer learning, where the learned features can be applied to a specific task like food recognition, even if the exact categories differ.

However, there are some limitations of the model. More recent architectures like ResNet, Inception, or EfficientNet are often more efficient than VGG16. These newer models provide similar or better performance with fewer computational resources. Meanwhile, VGG16 typically requires a fixed input size of 224x224 pixels. This can be a limitation if the original food images vary significantly in size or scale, requiring potentially distorting pre-processing steps.

#### 3.1.2. ResNet50

ResNet50 is primarily used for image classification, which has 50 layers that contain trainable parameters.The core idea of ResNet is introducing a residual learning technique to help with the vanishing gradients problem in deep neural networks, enabling the model to be effectively deeper for better performance without the training becoming inefficient.

Advantage & Disadvantage of the Model

The use of residual connections helps alleviate the vanishing gradient problem that can occur with very deep networks. This ensures that the model can be trained deeply enough to capture intricate details without the training process becoming inefficient, which is beneficial for distinguishing subtle differences in food items.

However, each convolutional layer in ResNet applies filters that capture patterns within small, localized regions of the input image; it struggles to integrate information across distant parts of the image effectively.

### 3.1.3. Hybrid CNN Transformer Models

Researchers have developed hybrid models that combine the strengths of CNNs and Transformers. By leveraging CNNs for local feature extraction and Transformers for capturing global context, these models can achieve a more comprehensive feature representation of images, potentially leading to superior performance and better generalization, especially with smaller datasets. Thus, we have experimented with CNN-Transformer hybrid models alongside using pretrained ResNet models for effective performance.

Advantage & Disadvantage of the Model
Hybrid CNN-Transformer models offer several advantages over using either architecture alone. By combining a pre-trained CNN (like ResNet50) for local feature extraction with a Transformer encoder for capturing global context, these models achieve improved feature learning. This can lead to better performance on image classification tasks, especially when dealing with smaller datasets. Additionally, the flexibility of choosing different pre-trained CNNs and adjusting Transformer hyperparameters allows for adaptation to specific tasks and data characteristics.

However, the integration of CNN and Transformer architectures requires careful design and can increase the overall complexity of the model. Furthermore, these models have more hyperparameters to tune, which can be challenging for optimal performance.

### 3.1.4. Squeeze and Excitation Block

The integration of Squeeze-and-Excitation (SE) blocks into a convolutional neural network (CNN) represents a significant enhancement over basic CNN architectures, particularly in the context of image classification tasks. The SE block refines the channel-wise feature responses by explicitly modeling interdependencies between channels, thereby improving the representational power of the network. In the squeeze phase, global spatial information is compressed into a channel descriptor by using global average pooling, reducing each channel to a single numerical value. This process summarizes the global distribution of feature responses across the spatial dimension. The excitation phase then follows, where this channel descriptor is passed through a two-layer neural network. This acts like a gating mechanism with a sigmoid activation at the output to learn a non-linear interaction between channels.

Advantage & Disadvantage of the Model
By focusing on useful features and suppressing less relevant ones, SE blocks help the network better capture the distinctive aspects of various food items, which can be particularly beneficial given the diverse appearance and textures in food classification. However, Incorporating SE blocks adds additional parameters and complexity to the network, which may lead to increased computational overhead.

### 3.1.5. VIT Transformer

Vision transformer adapts the traditional transformer architecture in a manner it can be converted to use in image data as to its usual text processing tasks. Instead of relying on progressively deeper layers of convolutional layers, this applies transformers to capture spatial dependencies within pixel values. The model considers images as a sequence of patches and uses a self-attention mechanism to provide weighted importance to each patch in relation to others, enabling it to capture both local and global contextual info. Therefore, this architecture performs better than traditional convolution architecture.

Advantage & Disadvantage of the Model
The Vision Transformer (ViT) offers a significant advantage in classifying 20 different food items due to its ability to capture global context and relationships between different parts of an image. Unlike conventional CNNs that focus on local features through small receptive fields, which does not allow us to understand broader and more complex visual patterns crucial for distinguishing varied food types. However, ViT's reliance on large amounts of training data to achieve optimal performance and its computationally intensive nature due to self-attention mechanisms can be a disadvantage.

### 3.1.6. Deit Transformer

Data-efficient Image Transformers (DeiT) was developed to address one of the main drawbacks of Vision Transformers (ViT): their heavy reliance on large-scale datasets for training. DeiT introduces a methodology that includes a distillation token, which mimics the role of a student learning from a teacher (typically a pre-trained CNN). This process not only speeds up the learning but also enhances the model's ability to generalize from smaller datasets. This approach significantly enhances the data efficiency of the transformer, making it viable to train on our subsampled food database. Similar to ViT, DeiT retains the hierarchical representations which allows the model to capture both local details and global context within an image, making it adept at understanding complex visual patterns that are crucial in detailed image classification tasks like food recognition.

Advantage & Disadvantage of the Model
By incorporating knowledge distillation directly into the training process, DeiT trains effectively on much smaller datasets than required by traditional transformers. Despite its reduced data requirements, DeiT achieves performance comparable to more data-hungry models, making it suitable for a wide range of image classification tasks, including food classification. The introduction of a distillation token adds complexity to the training process, requiring careful tuning of the distillation loss and the overall training procedure to achieve optimal results.

### 3.1.7. Swin Transformer *(Best performing model)*

The core concept of Swin transformer revolves around using shifted windows to perform self-attention computations, which will allow the model to efficiently scale to larger food images and deeper networks. This method breaks down the image into smaller, manageable patches and applies self-attention locally within these patches initially, and then between the patches in subsequent layers through window shifting.

Unlike ViT, which applies global self-attention and thus considers all parts of the image equally, Swin uses shifted window-based self-attention. This approach significantly reduces computational complexity by limiting self-attention to non-overlapping local windows. The windows are shifted in subsequent layers, allowing for cross-window connections and enabling the model to build a global understanding of the image incrementally. So this architecture performs better than most vanilla encoder-decoder architecture. This approach allows us to generalize better from smaller amounts of data compared to ViT, which is beneficial for our sub-sampling technique for data loaders.

Construction & Layers

1. Patch Partitioning: The image is first divided into patches (e.g., 4x4 pixels each), which are treated as tokens similar to words in NLP.
2. Embedding Layer: These patches are then embedded into a high-dimensional space.
3. Shifted Window Partitioning: To perform self-attention efficiently, Swin Transformer applies self-attention within local windows (e.g., 7x7 patches). In subsequent layers, these windows are shifted to cover the gaps left by previous window configurations, ensuring comprehensive coverage and interaction across the image.
4. Hierarchical Structure: The model employs a pyramid-like structure where the resolution is gradually reduced while increasing the feature dimensions, enhancing the model's ability to capture more abstract and global features at higher layers.
5. Self-Attention and MLP Blocks: Each layer consists of self-attention blocks and MLP blocks, with LayerNorm applied before each block and residual connections after each block.
6. Classifier Head: At the top of the network, global average pooling is followed by a linear layer for classification.

Advantage & Disadvantage of the Model
The Swin Transformer offers several compelling advantages for tasks like food classification, which requires handling varied scales and complex textures. Its ability to learn multi-scale representations is

crucial for accurately identifying diverse food items, from close-up shots of grains to wide images of entire meals. Additionally, the Swin Transformer's design is inherently flexible, allowing easy adaptation and scaling according to the task's requirements, such as adjusting the number of layers or window sizes. However, The introduction of new hyperparameters like window size and shift size also complicates the tuning process, requiring more meticulous adjustment to achieve the best results.

Training Process & Results

Initialization includes loading a pre-trained Swin Transformer and a feature extractor configured specifically for image classification. All parameters of the model are initially frozen to retain learned features from vast datasets, except for the last classifier layer, which is unfrozen and replaced to tailor the network for classifying 20 distinct food classes. During training, the model computes the loss using cross-entropy and tracks accuracy using the Accuracy metric from torchmetrics, optimized for these specific tasks with the AdamW optimizer targeting only the trainable parameters of the classifier.

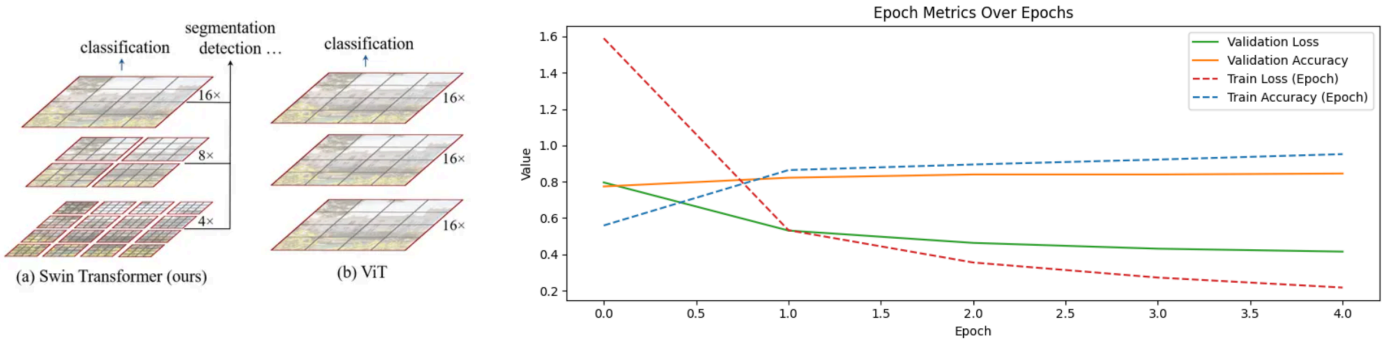



Fig 3,4: Visualisation of Swin Transformer architecture and model performance evaluation

### 3.2. OCR

This portion of implementation focuses on complementing our food recognition model to deliver a well-rounded solution to our users. We employ an OCR pipeline, EasyOCR, to extract information from our nutrition label images. While there is no training process, it is observed that simply passing the raw image input into the pipeline will extract unnecessary text outside of the nutrition label in food packaging, thus there is a need to perform image processing and we document our methodology as follows.

Nutrient Details Extraction Process

1. Convert to Grayscale: OpenCV loads images in BGR format instead of RGB, which is inconsequential for our use case. However, image input is converted into grayscale to reduce three color channels to one so that computational complexity is decreased.
2. Gaussian Blur: Used to smooth image input by reducing noise and unwanted details. This would help in minimizing false detections for edge detection. We used a Gaussian kernel of 7 with a corresponding sigma calculated from the kernel size.
3. Edge Detection: Identify significant differences in neighboring pixel values to detect edges. We made use of OpenCV's Canny detector as it provides good accuracy and is robust to noise. In Singapore's context where nutrition labels are mostly standardized and confined without a tubular boundary, it is easy for Canny to pick up the rectangle.
4. Define Contours: We identify the image contours to subsequently trace the edges. But not all contours found are rectangles, so we use an approximation technique to simplify the contours and identify the rectangles. We will finally detect the largest rectangle, which is the nutrition label and extract the label image.
5. Perform OCR: We implement an OCR pipeline customized to the English language and extract out the text and coordinates. We will finally pass the information into OpenAI and through prompting.

6. GPT 3.5: OpenAI has trained on large datasets nutritional label images, especially if these images are conforming to standards like that of the US Food and Drug Administration. There are two sets of prompts: to extract the nutrition value per serving and amount of calories per serving respectively.

How it works

Users input their information like height and weight and their target calories. For fresh foods our solution make use of our trained model, but for packaged food there can be countless new products released into the market and by adopting OCR techniques, they can ensure their daily calorie count can continue to be tracked by allowing them to take picture of the label, and nutrition information will be calculated into their daily intake needs.

Advantage & Disadvantage

Most food packages Singapore conform to the FDA's standard or adopt a variant of it, making Canny a suitable algorithm, since the nutrition facts are almost boxed up in a bordered table. So long the image contains the entire nutrition label, EasyOCR will be able to pick it up. Using openAI's GPT3.5, which has learnt images of nutrition labels of multiple standards depending on the country of origin, we should be able to get the paired data of nutrition and value.

However, if the labeling does not conform to any standard or is not put into a tabular format with borders, Canny may not be able to pick up where the table is. To ensure robustness, we can also employ image segmentation techniques by running images of all food packaging together with their mask to train a CNN model.

**4. Results: Model Result Comparison and Final Selection**

| Model Name | Test Accuracy (%) |
|---|---|
| ResNet50 | 77.8 |
| VGG16 | 74.9 |
| Hybrid (ResNet50) | 60.4 |
| Hybrid (ResNet18) | 68.1 |
| Squeeze and Excitation Block | 19.8 |
| VIT Transformer | 78.8 |
| DeiT Transformer | 79.9 |
| Swin Transformer | 85.9 |

Table 1: Sorted by order of model being mentioned

| Model Name | Test Accuracy (%) |
|---|---|
| Swin Transformer | 85.9 |
| DeiT Transformer | 79.9 |
| VIT Transformer | 78.8 |
| ResNet50 | 77.8 |
| VGG16 | 74.9 |
| Hybrid (ResNet18) | 68.1 |
| Hybrid (ResNet50) | 60.4 |
| Squeeze and Excitation Block | 19.8 |

Table 2: Sorted by model accuracy

Given the test dataset, the Swin transformer was the best performing model in successfully recognizing items from 20 food categories with the highest accuracy of 85.9%. Whereas, the Squeeze and Excitation Block yielded the lowest test accuracy of 19.8%. Most of the better performing models belong to the Transformer category, yielding about 80% which is slightly higher than the ResNet50 model yielding 78%,

which is a powerful and popular image classification model. The hybrid models, which were built on similar ResNet models as well, ironically performed worse than the ResNet50 standalone itself (68<78%). Furthermore, the hybrid model employing ResNet18 as a building block performed better than its ResNet50 counterpart (68>60%), which is counter-intuitive because the latter offers higher representational capability and is known to achieve higher accuracy than the former.

**5. Discussion: Experiment Insights & Business Implications**

With regards to the Swim based transformer architecture performing the best overall, we account it to the fact that Swim transformers are superior in terms of capturing spatial dependencies in between different food textures.

Whereas on the flip side with regards to the Squeeze and Excitation Block yielding the lowest accuracy by a substantial margin, we desire to mention this model is the only model among all we experimented with where we pretrain the model from scratch rather than apply transfer learning. Furthermore, given the limited data samples of 18K, we believe there is insufficient data for proper training from scratch and hence it is unable to yield satisfactory results.

Hybrid vs standalone models

The hybrid models were in general giving an accuracy of <70%, which is lower than traditional CNN models like VGG16 and ResNet which give an accuracy of 75%.

In theory, Hybrid models combine the spatial hierarchy learning capabilities of CNNs with the global receptive fields of transformers which theoretically offers comprehensive feature extraction (local and global contexts). However, we argue the underperformance of the hybrid models is due to an imbalance where neither component is optimized fully. Namely for food classification where features like texture, shape, and color are critical, the straightforward feature extraction by CNNs is more directly effective compared to the more generalized feature processing in transformers.

Given that simpler CNN architectures provide comparable or superior performance, businesses might reconsider allocating resources towards developing and training more complex hybrid models for this specific task. The higher computational and time costs associated with hybrid models may not justify the marginal gains (if any) in accuracy. Deploying simpler CNN models like VGG16 or ResNet can be more efficient, not just in terms of computational resources but also in ease of integration, maintenance, and scalability within production environments.

Hybrid ResNet investigation

We earlier mentioned the hybrid models employing ResNet18 and ResNet50 as respective building blocks yielded accuracies of 68.1% and 60.4% correspondingly, where the ResNet18-base model outperformed its ResNet50-base counterpart despite it having reduced model complexity.

Several factors could be influencing this outcome. ResNet-50, with its additional layers, inherently has a higher capacity for learning complex features compared to ResNet18. While this can be beneficial for larger and more diverse datasets, in our scenario where the dataset is not sufficiently large or varied, the increased model complexity is perhaps leading to overfitting. For a task like food classification where distinguishing features between classes does not require very deep hierarchical representations, the additional layers in ResNet50 may not be providing significant benefits to warrant its benefits.

If the increase in depth does not result in improved performance, it may not be cost-effective to use such models, especially in a production environment where efficiency and speed are crucial.

Overall Insights

The performance ranking of different models in the task of classifying 20 food categories highlights key insights into the training process and data suitability. Namely, we take note of hybrid models performed the least effectively, followed by CNNs with transfer learning like VGG16 and ResNet, and transformer-based models achieving the highest accuracy. This suggests that while hybrid models might

be too complex for the available data, traditional CNNs benefit significantly from transfer learning but still fall short of the more dynamically capable transformer models, which excel due to their advanced handling of global and local context within images.

For businesses, this implies that investing in transformer technologies, particularly in environments where understanding intricate details and variations is crucial, could yield better performance and offer a competitive edge. This shift would necessitate a focus on acquiring high-quality, diverse datasets and possibly more computational resources, but the payoff in terms of accuracy and user satisfaction could justify the investment, especially in sectors like culinary tech or food services where precise image classification drives core functionalities.

**6. User Flow**

To accomplish the objective of accurately tracking caloric intake from both freshly prepared local dishes and packaged foods, our application seamlessly merges two core functionalities to enhance the user experience. Initially, users will enter their personal details and fitness objectives. When having fresh foods, the user will take photos for image recognition and calorie estimation, which will be immediately recorded by our application. When having packaged foods, the user will take photos of the corresponding nutrition labels for our OCR pipeline to recognize and record. Following the image processing, the application provides users with a tailored daily calorie intake recommendation. Additionally, it displays the accumulated daily caloric intake as a percentage, assisting users in monitoring their remaining caloric allowance.

**7. Conclusion**

**Achievement of Objectives**

This project set out with the goal of developing a solution that leverages advanced machine learning techniques to provide accurate, real-time nutritional information of food that can be commonly found in Singapore. Through our systematic model exploration and innovative business solution, we have successfully achieved the goal of helping users identify, record and track their calorie intakes from both freshly prepared local food and packaged food.

**Difficulties & Limitations**

- Computing Resource Constraints: Training models such as transformer requires substantial computing power. Due to lack of computing resources, our model training process appears to be relatively slow. We eventually use subsampling mentioned in the data preprocessing section to increase our training efficiency.
- Limited Food Categories: Due to time and computing resource constraints, we demonstrate 20 local food categories in our solution. The solution may not support other local fresh food recognition at the moment.
- Variability in Food Preparation Process: Even within the same food category, the actual calorie may differ from our estimation due to the difference in preparation methods, recipes, portions. Therefore, the solution can provide only approximate estimation at the moment.

**Future Vision & Outlook**

Based on our current achievement and current challenge, we have determined our future development strategy:

- Expand the Food Categories: To enhance the user experience, we will explore more food options for our image recognition task, by incorporating additional local dishes and commonly eaten fresh foods that extend beyond the existing local categories.
- Enhanced Customization Options: We will offer users the ability to select portion sizes, ingredients, and cooking preferences when they upload photos of local foods. This enhancement will allow the application to provide more precise calorie estimates.

# APPENDIX A

**References**

Foodlg (2021). http://www.foodlg.com/.

Derricklty, Singapore-Hawker-Food-Classifier, (2021), GitHub repository, https://github.com/Derricklty/Singapore-Hawker-Food-Classifier

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., & Guo, B. (2021, March 25). Swin Transformer: Hierarchical Vision Transformer using Shifted Windows. arXiv.org. https://arxiv.org/abs/2103.14030

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., & Houlsby, N. (2020). An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. arXiv (Cornell University). https://doi.org/10.48550/arxiv.2010.11929

Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., & Jeǵou, H. (2020). Training data-efficient image transformers & distillation through attention. arXiv (Cornell University). https://doi.org/10.48550/arxiv.2012.12877

Hu, J. F., Shen, L., Albanie, S., Sun, G., & Wu, E. (2017). Squeeze-and-Excitation networks. arXiv (Cornell University). https://doi.org/10.48550/arxiv.1709.01507

Lin, T., Dollár, P., Girshick, R., He, K., Hariharan, B., & Belongie, S. (2016). Feature pyramid networks for object detection. arXiv (Cornell University). https://doi.org/10.48550/arxiv.1612.03144

**Contribution**

| Name | Contribution |
|---|---|
| Anshumaan Phukan | Model investigation and implementation of majority of models |
| Jared Cheang | Model implementation for Hybrid; code for data cleaning, implementation, dataloader |
| Roy Yeo | Final combined solution integration and implementation, OCR implementation, demo |
| Yi Kai Lim | Image files and class selection, nutrition data preparation, label detection, OCR implementation |
| Yuhe Xin | Model implementation for VGG, ResNet; Presentation and slides organiser |

# APPENDIX B: Model Details

### 3.1.1. VGG16

The VGG16 model is a convolutional neural network (CNN) architecture, and is primarily used for image recognition tasks, such as identifying objects within an image. It has been widely adopted for various applications in computer vision, including image classification, image segmentation, and feature extraction.

Construction & Layers

1. **Input Layer**: Takes an input image of fixed size 224x224 pixels, with three color channels (RGB).
2. **Convolutional Layers**: The network uses a total of 13 convolutional layers. These are equipped with 3x3 filters with a stride of 1 pixel. The padding is designed to keep the spatial resolution of the image unchanged after the convolution (same padding).
3. **Activation Functions**: Each convolutional layer is followed by a ReLU (Rectified Linear Unit) activation function, which introduces non-linearity into the model helping it to learn more complex patterns in the data.
4. **Pooling Layers**: There are 5 max-pooling layers scattered throughout the architecture, each using a 2x2 pixel window, with a stride of 2 pixels. These are used to reduce the spatial dimensions of the input volume for the layers that follow, which reduces the number of parameters and computation in the network.
5. **Fully Connected Layers**: Following the convolutional layers, there are 3 fully connected layers. The first two have 4096 channels each, and the third performs 1000-way classification (matching the number of classes in the ImageNet dataset), ending with a softmax activation.
6. **Softmax Activation**: The final layer uses a softmax activation function that outputs a probability distribution for 20 class labels.

Training Process & Results
The VGG16 base layers are frozen to preserve learned features, while additional custom layers include global average pooling and dense layers equipped with dropout and L2 regularization to prevent overfitting. After training with the training dataset, we reached accuracy of 74.9% in the test dataset.

### 3.1.2. ResNet50

ResNet50 is primarily used for image classification, which has 50 layers that contain trainable parameters.The core idea of ResNet is introducing a "residual learning" technique to help with the vanishing gradients problem in deep neural networks, enabling the model to be effectively deeper for better performance without the training becoming inefficient.

Construction & Layers

1. **Input Layer**: The network takes an input image typically of size 224x224 pixels, with 3 color channels (RGB).
2. **Initial Convolution and Pooling Layer**: The image first passes through a convolutional layer with a 7x7 kernel and a stride of 2, followed by a 3x3 max pooling layer, also with a stride of 2. This step reduces the spatial dimensions while increasing the depth.
3. **Residual Blocks**: The core component of ResNet50 is its use of residual blocks. These blocks consist of a few convolutional layers followed by batch normalization and ReLU activation. Each block has a "shortcut" or "skip connection" that adds the input of the block to its output, which helps to mitigate the problem of vanishing gradients as the network depth increases.
4. **Stacks of Residual Blocks**: ResNet50 contains 4 stages of residual blocks, each stage having a varying number of blocks and filters. The stages use convolutional layers with 3x3 and 1x1

kernels. The number of filters in these layers doubles as the spatial dimensionality decreases, typical of deeper convolutional networks.
5. **Ending Layers**: After all residual blocks, the network uses a global average pooling layer to reduce each feature map to a single vector. This is followed by a fully connected layer which acts as the classifier part of the network.
6. **Output**: The final output layer uses a softmax activation function to output a probability distribution over the class labels.

Training Process & Results

The ResNet50 model is employed with pretrained ImageNet weights but without its original top layer, allowing for the addition of new layers tailored for a 20-class classification task. The additional layers include a Global Average Pooling layer, a dense layer with 512 neurons for deep feature processing, followed by a dropout layer to prevent overfitting, and another dense layer with softmax activation for class predictions. After training with the training dataset, we reached accuracy of 77.8% in the test dataset.

### 3.1.3. *Hybrid CNN Transformer Models:*

Traditional image classification models often rely on either Convolutional Neural Networks (CNNs) or Transformers. While CNNs excel at capturing essential visual details like textures and shapes, they struggle to grasp the relationships between these features across the entire image. Conversely, Transformers are adept at understanding long-range dependencies but might require substantial data for effective local feature learning. To address these limitations, researchers have developed hybrid models that combine the strengths of CNNs and Transformers. By leveraging CNNs for local feature extraction and Transformers for capturing global context, these models can achieve a more comprehensive feature representation of images, potentially leading to superior performance and better generalization, especially with smaller datasets. Thus, we have experimented with CNN-Transformer hybrid models alongside using pretrained ResNet models for effective performance.

Construction & Layers

1. **Input Layer**: The network takes an input image typically of size 224x224 pixels, with 3 color channels (RGB).
2. **CNN Feature Extraction**: A pre-trained CNN ResNet is employed to extract feature maps from the image where these feature maps capture low-level visual features like textures and edges. Here, 2 types of ResNet models were explored as the base CNN model, which are pretrained ResNet18 and ResNet50.
3. **Patching and Embedding**: The extracted feature maps are divided into patches, where each patch is flattened and embedded into a high-dimensional vector using an embedding layer.
4. **Transformer Encoder**: A Transformer encoder receives the embedded patches where it utilizes self-attention mechanisms to learn long-range dependencies and relationships between features across different patches. Here, there are 512 or 2048 nodes corresponding to the base model being ResNet18 or ResNet50 respectively.
5. **Classification Head**: The final output layer uses a softmax activation function to output a probability distribution over the class labels.

Training Process & Results

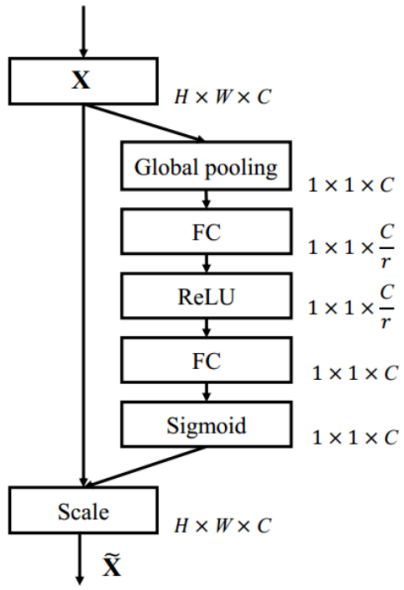
2 variants of the CNN-Transformer were employed, using ResNet18 and ResNet50 as the base CNN model where the corresponding transformers consist 512 or 2048 nodes respectively. After training with the training dataset, the ResNet18 and ResNet50-base hybrid models yielded accuracies of 68.1% and 60.4% respectively in the test dataset, where the ResNet18-base outperformed the ResNet50-base hybrid model. We further discuss this later under Experiment Findings.

### 3.1.4. *Squeeze and Excitation Block:*

The integration of Squeeze-and-Excitation (SE) blocks into a convolutional neural network (CNN) represents a significant enhancement over basic CNN architectures, particularly in the context of image classification tasks such as identifying 20 different food categories. The SE block, originally proposed in the Squeeze-and-Excitation Networks paper, refines the channel-wise feature responses by explicitly modeling interdependencies between channels, thereby improving the representational power of the network.

The block works in two phases: squeeze and excitation. In the squeeze phase, global spatial information is compressed into a channel descriptor by using global average pooling, reducing each channel to a single numerical value. This process summarizes the global distribution of feature responses across the spatial dimension. The excitation phase then follows, where this channel descriptor is passed through a two-layer neural network. This acts like a gating mechanism with a sigmoid activation at the output to learn a non-linear interaction between channels. In simpler words, instead of assigning equal importance to feature maps in a convolution operation, we assign weighted parameters to represent the feature map. This establishes a relationship between feature maps which is an improvement over traditional CNN architecture.

During the forward pass, the input feature map is first squeezed to produce a channel-wise descriptor. This descriptor is then passed through the excitation pathway, resulting in a set of modulation weights. These weights are reshaped and scaled up to the original dimensions of the input feature map and applied to each channel of the input by multiplication. This process effectively allows the network to perform a dynamic channel-wise feature recalibration.




Construction & Layers

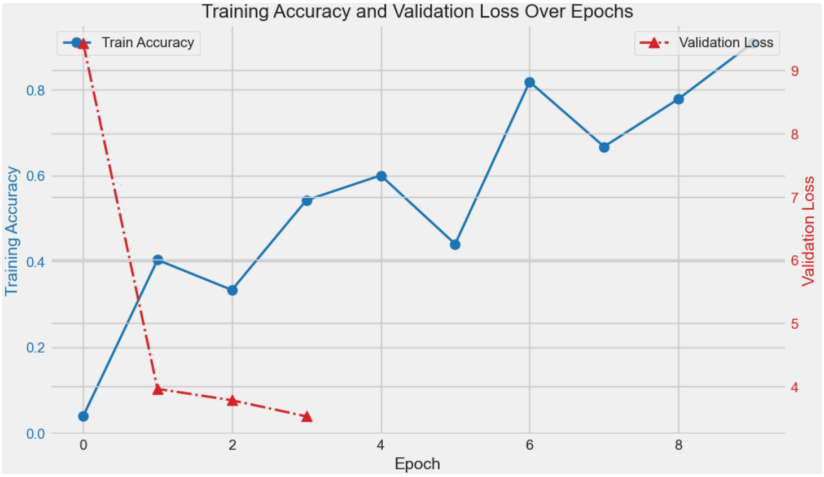
1. Adaptive Average Pooling (Squeeze): The first component of an SE block is the adaptive average pooling layer. This layer squeezes global spatial information into a channel descriptor, reducing each channel of the input feature map across spatial dimensions (height and width) to a single numerical value.
2. Fully Connected Layers (Excitation): The squeezed output is then processed through two fully connected layers. The first layer reduces the dimensionality of the channel descriptor by a factor specified by the reduction ratio, using ReLU activation for non-linearity. The second linear transformation projects the features back to the original channel dimension and applies a sigmoid activation to generate channel-wise weights.

**Forward Pass in SE Block**
During the forward pass, the input feature map is first squeezed to produce a channel-wise descriptor. This descriptor is then passed through the excitation pathway, resulting in a set of modulation weights. These weights are reshaped and scaled up to the original dimensions of the input feature map and applied to each channel of the input by multiplication. This process effectively allows the network to perform a dynamic channel-wise feature recalibration.

Integration within a CNN (CustomSENet)

1. Convolution and Batch Normalization: Each convolutional layer (nn.Conv2d) extracts spatial features, which are then normalized by a batch normalization layer (nn.BatchNorm2d).
2. Activation: A ReLU activation layer follows (nn.ReLU()), introducing non-linearity into the network.
3. Squeeze-and-Excitation Block: Post-activation, the SE block recalibrates the feature maps, emphasizing informative features and suppressing less useful ones dynamically.
4. Pooling: After recalibration, a max pooling layer (nn.MaxPool2d) reduces the spatial dimensions of the feature maps, preparing them for the next set of layers or for final classification.

Training Process & Results
This model begins by sequentially processing input images through convolutional layers, each followed by an SE block that adaptively recalibrates channel-wise feature responses, boosting important features while suppressing less useful ones. As the images pass through the network, features are pooled and flattened before being classified by a fully connected layer. During training, the model computes logits from which it derives the cross-entropy loss and measures accuracy, logging these metrics to monitor training and validation performance. The optimizer used is AdamW, chosen for its effectiveness in handling sparse gradients and adaptive learning rate capabilities, which together facilitate efficient convergence and robust generalization.




### 3.1.5. *VIT Transformer*

Vision transformer adapts the traditional transformer architecture in a manner it can be converted to use in image data as to its usual text processing tasks. Instead of relying on progressively deeper layers of convolutional layers, this applies transformers to capture spatial dependencies within pixel values. The model considers images as a sequence of patches and uses self-attention mechanism to provide weighted importance to each patch in relation to others, enabling it to capture both local and global contextual info. Therefore, this architecture performs better than traditional convolution architecture.

Construction & Layers

1. **Input Layer**: The resized image is taken and then split into smaller or patches. These patches are then flattened and linearly transformed into a dimension similar to the subsequent transformer layers.
2. **Patch Embedding**: The patches generated from the inputs are considered as a token similar to an NLP model. These patches are embedded into a higher dimensional space to provide meaningful input for the transformer. Positional embeddings are combined with patch embeddings to maintain the spatial context of each patch within the image grid. This is essential because the transformer model does not naturally recognize the position of data in a sequence.
3. **Transformer Encoder**: The architecture consists of several layers of transformer encoders. Each encoder layer contains a multi-head attention layer and a feed-forward network (FFN).
4. **Layer Normalization**: To facilitate faster training, LN is applied before and after each transformer layer.
5. **Classification Head**: After going through all the layers, the output corresponding to the aggregation of all patch outputs is passed to a final linear layer.

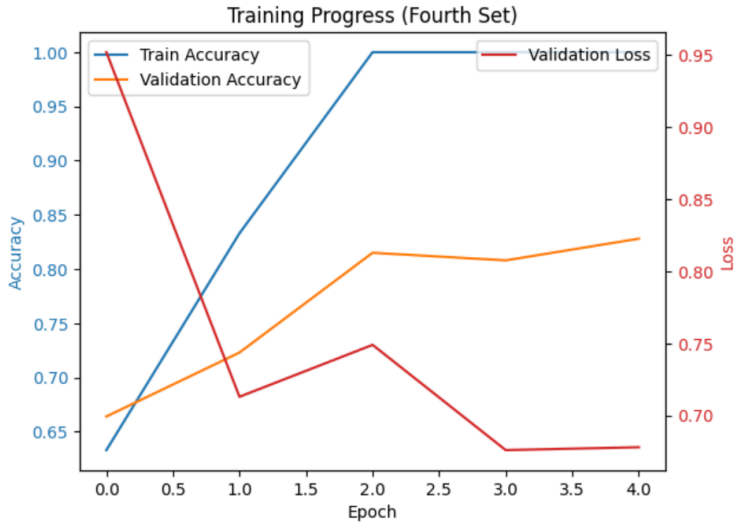
Advantage & Disadvantage of the Model

The Vision Transformer (ViT) offers a significant advantage in classifying 20 different food items due to its ability to capture global context and relationships between different parts of an image. Unlike conventional CNNs that focus on local features through small receptive fields, which does not allow us to understand broader and more complex visual patterns crucial for distinguishing varied food types. However, ViT's reliance on large amounts of training data to achieve optimal performance and its computationally intensive nature due to self-attention mechanisms can be a disadvantage.

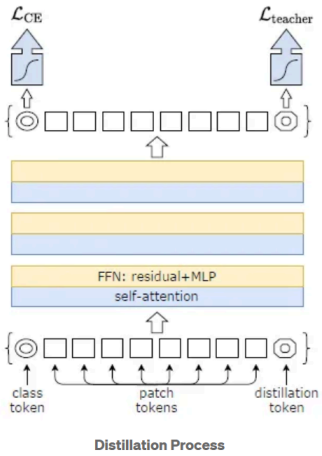
Training Process & Results

During initialization, We initiate a pre-trained transforme, adapting it for 20 categories. It sets up a training regimen where all parameters are initially frozen except for the last block, allowing for fine-tuning on our specific use case of recognizing 20 food items.

The Vision Transformer uses the GELU (Gaussian Error Linear Unit) activation function. Adam optimizer with weight decay regularization is applied to prevent overfitting, with cross entropy as our loss function.



### 3.1.6. *Deit Transformer*




Distillation Process

Data-efficient Image Transformers was developed to address one of the main drawbacks of Vision Transformers (ViT): their heavy reliance on large-scale datasets for training. DeiT introduces a methodology that includes a distillation token, which mimics the role of a student learning from a teacher (typically a pre-trained CNN). This process not only speeds up the learning but also enhances the model's ability to generalize from smaller datasets. This approach significantly enhances the data efficiency of the transformer, making it viable to train on our subsampled food database. Similar to VIT, DeiT retains the hierarchical representations. This feature allows the model to capture both local details and global context within an image, making it adept at understanding complex visual patterns that are crucial in detailed image classification tasks like food recognition.

Construction & Layers

1. Input Layer: DeiT processes the input image by dividing it into fixed-size patches, similar to ViT, which are then flattened and linearly projected.
2. Embedding and Positional Encoding: Each patch embedding is summed with a positional encoding to retain information about the original position of the patch in the image, as transformers inherently do not process sequential data with positional context.
3. Distillation Token: Unique to DeiT, a distillation token is added alongside the class token (used in ViT) at the input of the transformer. This token learns from the output of a pre-trained teacher model, improving the learning efficiency.
4. Transformer Encoders: Multiple layers of transformer encoders consisting of multi-headed self-attention and MLPs (multi-layer perceptrons) process the sequence of embedded patches.
5. Classification Head: The transformed class token (the first token in the sequence after passing through the transformer layers) is passed through a linear layer to produce the final classification output.

Training Process & Results

Initially, the model is configured with pretrained parameter that are mostly frozen except for the parameters in the last block, allowing fine-tuning on a specific task with 20 classes. During the training process, the model takes batches of images and labels, computes logits, and evaluates the cross-entropy loss and accuracy, logging these metrics for both training and validation steps. The model employs an AdamW optimizer with a learning rate scheduler to adjust the learning rate dynamically, optimizing training efficiency.