Università degli Studi di Milano - Bicocca
Dipartimento di Informatica, Sistemistica e Comunicazione
Corso di Laurea Magistrale in Data Science

Food Recognition with Vision Transformers

Relatore: Prof. Paolo Napoletano

Correlatore: Prof. Simone Bianco

Relazione della prova finale di:
Gaetano Chiriaco
Matricola 882638

Anno Accademico 2022-2023

Abstract

Food recognition is a major challenge in the field of computer vision, requiring models that can effectively handle the wide variability and complexity of food images. In this thesis, vision transformers, a category of models based on self-attention mechanisms, are used to address the task of food recognition. The work focuses on training and fine-tuning different vision transformer architectures on Food2K, a large-scale dataset of food images with 2,000 categories. The performance of vision transformers are compared with convolutional neural networks (CNNs) on Food2K and Food-101. The effects of pre-training on Food2K are studied, comparing the performances obtained by using ImageNet weights and Food2K weights. Food-101 is used to assess the generalization capabilities of transformers, in order to test if these models are robust enough to be used in a real-world scenario. In addition, state-of-the-art explainability techniques are used to highlight the regions of interest that vision transformers take into account when performing a prediction. The relevancy maps obtained are compared to a ground-truth mask, to verify if transformers trained for recognition can achieve competitive results on a segmentation task. The results show that vision transformers can achieve competitive results on food recognition tasks, with the added benefit that pre-training on Food2K improve their generalization capabilities and interpretability. This study highlights the potential of vision transformers in food computing, paving the way for future research in this field.

Contents

Abstract **i**

1 Introduction **1**
1.1 Motivation . . . 1
1.2 Problem statement . . . 3
1.3 Contribution . . . 5
1.4 Outline . . . 6

2 Background & Related Work **9**
2.1 Proposed models for recognition . . . 9
2.2 Most used food image datasets . . . 14

3 Transformer Models **17**
3.1 Introduction and fundamentals . . . 17
3.2 Vision transformers . . . 21
3.3 Explainability in Vision Transformers . . . 28

4 Data **33**
4.1 Food2K . . . 33
4.2 Food-101 . . . 36
4.3 UECFoodPix . . . 39

5 Proposed Approach **43**
5.1 Experiments and objectives . . . 43
5.2 Data preprocessing and regularization . . . 44
5.3 Models and training scheme . . . 48

6 Results **53**
6.1 Performance on Food2K . . . 53
6.2 Generalization on Food-101 . . . 55
6.3 Segmentation and explainability . . . 58

Conclusions **65**

Bibliography **67**

1 Introduction

Contents

1.1 Motivation 1
1.2 Problem statement 3
1.3 Contribution 5
1.4 Outline 6

1.1 Motivation

Over the past few years, food-related studies have garnered increasing attention due to their significant impact on people's lives and well-being. Many individuals have expressed the need to understand and record their dietary choices in order to improve their personal diet management and nutritional analysis. The improvements in comprehending food and automating its analysis have paved the way for a variety of applications, such as mobile visual food diaries, smart restaurants and shops and the identification of ingredients and macronutrients.

As interest in automating food analysis has increased, so has the distribution and use of tools that generate data on people's nutritional and physical habits. The widespread adoption of smartwatches, sensors, and smartphones has vastly expanded our capacity to collect data, while the advent of social networks has facilitated the sharing of countless food images. This has fueled the development of cutting-edge computer vision applications focused on food detection and recognition. This phenomenon has given rise to a field known as *food computing*, where the analysis of this vast volume of information offers compelling and valuable insights on what we eat.

In Min, Jiang, et al. 2018 food computing is defined as the "computational approaches for acquiring and analyzing heterogeneous food data from disparate sources for perception, recognition, retrieval, recommendation and monitoring of food to address food related issues in health, biology, gastronomy and agronomy".

Among the different food computing tasks, *food recognition* stands out as one of the most popular ones in literature. Food recognition involves the creation of applications or models that can automatically classify and recognize foods or main ingredients. These application can be used in a variety of contexts, here's a few notable mentions:

- *nutrition tracking and dietary analysis*: by taking pictures of their meals, users can receive detailed nutritional information such as calorie counts, macronutrient composition and vitamin content;
- *health and wellness apps*: food recognition is integrated in various mobile applications designed to promote healthy eating habits. These apps can provide personalized recommendations, meal planning assistance and recipe suggestions based on users' dietary preferences and nutritional needs;
- *menu analysis and food labeling*: by scanning menu items or food packages, users can obtain nutritional information, allergen warnings or ingredient lists;
- *food waste management*: by monitoring and analyzing food items in real-time, it can provide suggestions for utilizing leftover food or how to recycle a particular type of waste.

Traditional approaches to food computing have relied on rule-based systems and simplistic algorithms, which often fail to capture the intricacies and complexities of the culinary world. As a result, there is a pressing need for a more sophisticated and data-driven approach that can learn from vast amounts of culinary data and derive valuable patterns and knowledge.

This thesis aims to develop a *deep learning-based* framework for food recognition, using cutting edge techniques such as *transformers* applied to computer vision. By training these models on extensive culinary datasets, this work aspires to contribute to the vast field of food computing, paving the way for the use of innovative models on increasingly large food-related datasets. The models and methods proposed in the following sections are designed for generic food recognition tasks, but can be tuned to obtain tools suitable for any of the particular fields described earlier.

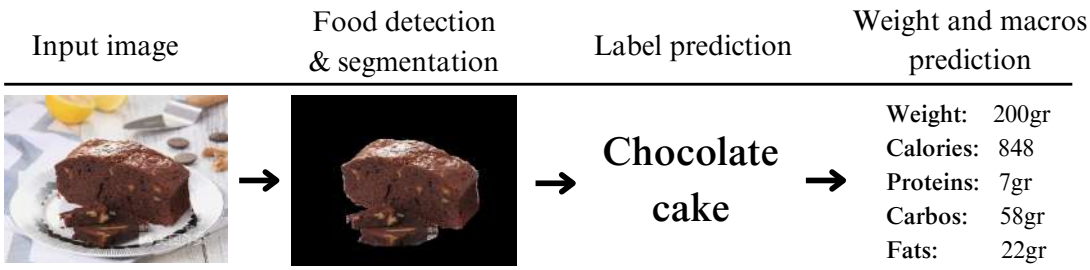

Figure 1.1: The dish is first detected and segmented, then classified. After that, further analysis can be carried out, like macronutrients or weight prediction

To be more specific, this thesis was conducted to answer the following research questions:

- *Are vision transformers better than convolutional neural networks for Food2K recognition?*
- *Are predictors trained on Food2K better then ImageNet-21K weights on food-related tasks?*
- *Can explainability techniques be applied to vision transformers to identify the key regions of a dish without explicitly training for segmentation?*

1.2 Problem statement

Food recognition is a task in which computer vision and machine learning techniques are used to identify and categorize different types of food based on visual information, such as images or videos. However, the analysis of food images often does not stop solely at the assignment of each dish to a class. With the development of deep learning algorithms, the accuracy of food detection and recognition methods have been drastically improved. Even simple models are capable of obtaining an accuracy close to the human eye, especially when the images are not chaotic and the food is clearly visible. This progress has opened doors to exploring more intricate and sophisticated recognition techniques.

In Subhi, Ali, and Mohammed 2019 food computing is thought like a pipeline, shown in Figure 1.1. Food in an image is first detected, than the image can be segmented, removing everything that is not identified as “food”. After that, the residual image is classified, associating a category to the picture. The predicted category can be used for deeper analysis such as estimation of volume and weight, or dish quality and ingredients. Finally, the nutritional characteristics of the dish can be calculated using the informations known a priori, the predicted category and its weight.

In Tahir and Loo 2021 e Y. Zhang et al. 2023 the food computing problem is represented as a composition of several categories, including:

- *representation of food images using deep or handcrafted features*: encode an image as an n-dimensional vector that can be used for other tasks;
- *classification of the food category*: use a model to assign to the image the probability of belonging to a specific class;
- *classification the ingredients*: extract a list of candidate ingredients that are likely to be found in the image;
- *food volume estimation*: estimate the weight and volume of the photographed dish.

These tasks are strongly related to each other and can be pursued simultaneously to achieve a more complex but more accurate model. For instance, the name of the meal can be used to extract some useful informations about its preparation or its main ingredients. If the label associated to an image is "mashed potatos", we already know without looking at the photo that the main ingredients are potatos and that probably the shape will be quite irregular. So the label can be used to build a model that manages to enrich the information derived from the image with such metadata. Alternatively, some foods may be immersed in a background that is too chaotic, which may undermine the predictive capabilities of the chosen method. Performing food detection and segmentation as an assist to classification methods is a frequently used and working practice.

Recognition models currently allow to tackle problems more complex than classification, but as it will be explained in section 2.2, the problem is often in the lack of data. In order to train a model to segment an image or to predict macronutrients and ingredients, it's necessary to have not only the image, but also the annotated ground-truth, like the list of ingredients or a matrix that indicates if a pixel is part of the foreground or background. There are few datasets on food in the literature, and those that make such specific information available are even fewer.

Food recognition, as well as many others computer vision tasks, can have a major impact on society and improve or automate processes. However, even when tackling a relatively simple task like classification, a number of complexities must be overcome to make algorithms and applications work properly. Food exhibits a wide range of variability in appearance, shape, color, texture and presentation. Additionally, factors such as lighting conditions, angles and occlusions further complicate the recognition process. Handling this variability and capturing the intricate details

of different foods pose significant challenges. Strong inter-class diversity is a very common characteristic in various datasets and the fact that each class is visually very different can make them more easily distinguishable. However, in the field of food recognition some classes can be considered "different" but appear really similar. For example, "chicken cutlet" and "beef cutlet" are two different foods, but visually can be indistinguishable even for the human eye. In addition , it's also common to have strong diversity within the same class. Food items within the same category can exhibit significant variance, caused by differences in cooking styles, ingredients and portion sizes. Lastly, visual food recognition needs fine-grained information to work correctly. Distinguishing between different types of pasta or meat only by using visual characteristics can be challenging due to subtle differences. For these and many other reasons, automating the recognition of food requires overcoming many obstacles, but if solved it can lead to great results with a strong impact on society and everyday dietary style.

The following research will be focused on the basic food recognition task, integrating the used models with methods commonly adopted in deep learning to improve performances and generalization capabilities.

1.3 Contribution

This work aims to address the existing gaps in the literature by integrating new models and benchmarks in the field of food recognition. Specifically, it aims to address the lack of research on training and evaluating vision transformers on Food2K (Min, Wang, et al. 2023), which is currently the most comprehensive publicly available dataset of food images. The most cited and used vision transformers, namely Vision Transformer (Dosovitskiy et al. 2020), Swin Transformer (Liu et al. 2021) and DeiT (Touvron, Cord, Douze, et al. 2021) are fine-tuned on the Food2K dataset and their performances are compared with those obtained by popular convolutional models, like ResNet (He et al. 2015), Inception (Szegedy et al. 2014) and others.

Additionally, a training method for neural networks is proposed, which utilizes food names embeddings, generated using BERT (Devlin et al. 2019), to guide the training and estimation of weights. This approach leverages both vision transformers and large language models to estimate a more robust and accurate model, that uses both text and images. In addition, to improve the predictive performance of chosen models, the most relevant image augmentation and regularization techniques will be studied and applied to avoid over-fitting and improve the generalization

capabilities.

Furthermore, an evaluation of the proposed models on Food-101 (Bossard, Guillaumin, and Van Gool 2014) is carried out to verify the effectiveness of pre-training on Food2K. Food-101 has been one of standard benchmark dataset in computer vision, commonly used to compare the proposed implementation with the state of the art. The aim is to assess if weights learned from Food2K can serve as a better starting point then ImageNet (Deng et al. 2009) weights for food-related task.

Finally, explainability techniques are employed on the trained Vision Transformers. These methods allow to gain insights into the inner workings of the transformer models and can provide a better understanding on how the attention mechanisms are employed. By visualizing the relevancy maps, the aim is to provide a more transparent and interpretable explanation on how transformers make their decision, on which ingredients they focus and if this type of models can be applied for image segmentation.

1.4 Outline

The first part of the study provides a comprehensive review of image classification and food recognition, with a specific focus on the application of deep learning methods for image processing. The current state of the art will be thoroughly analyzed, particularly highlighting the use of deep learning and the lack of research on transformers trained on food image datasets.

Moving on to the second part, the Food2K dataset is employed to evaluate the predictive performance of various vision transformers on food images. Furthermore, the generalization capabilities of these models is examined training the proposed models on Food-101, a popular benchmark for food recognition. Additionally, popular explainability techniques are presented, shedding light on how vision transformers operate and make their classifications.

Besides this introduction section, the research is structured into five chapters. Chapter 2 presents how classification methods applied to food images evolved in the last years, showcasing key papers, commonly used models and datasets. Chapter 3 offers an in-depth examination of deep learning transformer models. In this part the theoretical aspects behind transformers will be explained, discussing their advantages and limitations, and comparing their features to other type of deep learning architecture used in computer vision, convolutional neural networks. Particular emphasis is placed on three vision transformers used in this study: DeiT, Vision Transformer, and Swin Transformer. In chapter 4, the three datasets employed in this research are presented. The

primary focus is on Food2K, the dataset used for training and fine-tuning the vision transformers. Additionally, an overview of the characteristics of Food-101 and UECFoodPix is provided. Food-101 is employed to assess the generalization capabilities of vision transformers across different food-related datasets. On the other hand, UECFoodPix is utilized to evaluate the models' ability to identify the most significant parts of the image when classifying a dish. In chapter 5, all aspects related to the experiments and training procedures will be thoroughly explained. Moving on to chapter 6, the results obtained from the conducted experiments will be presented.

2

Background & Related Work

Contents

2.1 Proposed models for recognition 9
2.2 Most used food image datasets 14

2.1 Proposed models for recognition

Research about food recognition covers a wide range of skills and approaches, making it a diverse and complex field. However, it's possible to simplify the problem by reducing it to a basic classification task. In a classification problem a typical setup involves an input and a classifier that transforms an input into the desired output. When it comes to food image classification, the input is an image depicting a dish, while the classifier used can vary. An ideal classifier should be able to recognize any food category that was included in the learning phase. The accuracy of the used method depends on several factors, such as the quantity and quality of pictures used during training, the representation given to the images, and the complexity of the classification method employed. Additionally, an effective classifier should exhibit robustness, meaning it can handle situations where the dish deviates from its typical presentation.

To effectively process an image for classification, it's necessary to convert it into a set of features that are suitable for the classifier. By "feature" is meant a sequence of N numbers (a vector) that encodes an image. Images that are visually similar should be represented by vectors that are close in a N-dimensional space. The extraction of image features can be done in two ways:

computing handcrafted features or using deep learning approaches (Y. Zhang et al. 2023). Both the handcrafted and deep approaches seek to give a representation in features that capture different aspects of the image such as color and shapes, or more complex and latent characteristics such as color combinations and peculiar patterns. The extracted features can then be used as input to classification algorithms such as Support Vector Machine (Boser, Guyon, and Vapnik 1992), K-Nearest Neighbor and Random Forest (Breiman 2001), or used as input for a simple fully-connected neural network.

Non-deep classifiers require features with high informativeness as input to function properly. For this reason, it's common practice to segment the image before extracting these features, highlighting only the foreground, as explained in section 1.2. After the image has been properly pre-processed, features can be extracted. When using a non-deep classifier (SVM, KNN,RF), a class of methods called descriptors are used to extract features from images. The most used traditional descriptors are:

- *bag-of-features* (Csurka 2004): the basic idea of the bag-of-features representation is to obtain some "features" that refer to local descriptors or key points in an image. These features can be extracted using techniques such as Scale-Invariant Feature Transform (SIFT), Speeded-Up Robust Features (SURF) or Histogram of Oriented Gradients (HOG). Local descriptors are computed at salient points in the image, like edges or corners. The extracted local descriptors are encoded to create a compact representation of the image in visual words. Also, an histogram is built to represent the frequency of occurrence of each visual word in the image and the resulting feature histogram serves as the image representation used to classify the image;

- *color histogram* (Smeulders et al. 2000): a color histogram represents the number of pixels that have colors in each of a fixed list of color ranges. An histograms can serve as feature vectors, and then be used as the input for a classification method;

- *gabor features* (Granlund 1978): gabor features are vectors calculating by passing one or more gabor filters on an image via the convolution operation. A gabor filter is a two-dimensional array that is described by a number of parameters that regulate its orientation and wavelength, used for extracting useful features from an image, like edges and patterns. By convolving a set of gabor filters on the original image a set of response matrix are obtained, that can be further transformed to produce a set of features usable for classification or other tasks. An example of a set of gabor filters is shown in Figure 2.1.

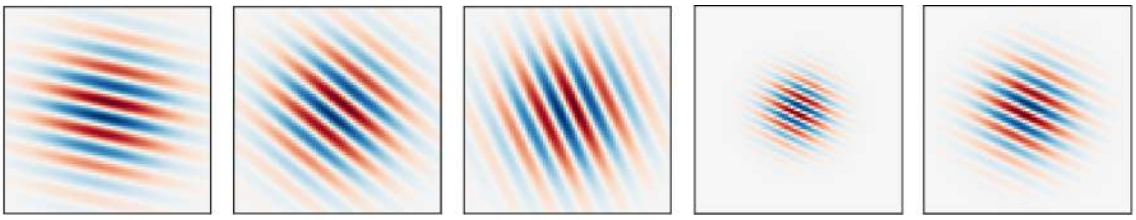

Figure 2.1: Examples of a set of gabor filters at different orientations and wavelengths. These filters can be convoluted on an image to extract the desired patterns.

Approaches like histograms or linear filters usually can struggle with handling variations in scale, rotation or occlusion. At time of writing, hand-crafted features are no longer used to classify a large set of food images since more complex and recent models perform much better. One example of the application of the handcrafted features-approach on a large scale food-related dataset can be found in Martinel, Piciarelli, and Micheloni 2016. Using a combination of SIFT, Color, Shape and Texture features, the Support Vector Machine used as classifier scored an accuracy of 55.89% on Food-101 dataset, which is one of the best results obtained with this kind of approach.

With the continuous progress of computer vision methods, convolutional neural networks (CNNs) have emerged as the dominant model for food recognition and image processing. Their effectiveness in image processing has been well-established, leading to a widespread adoption across various fields. In recent years, researchers have increasingly employed CNNs for food recognition, capitalizing on their ability to unveil higher-level features and hierarchical relationships through the addition of layers. However, it's important to note that as the depth of neural networks increases, training time and computational costs also escalate. Also, problems like vanishing or exploding gradients can emerge when training deep networks. Filters estimated in deep CNNs can be extremely complex and adapt to the field of study and type of images that one is trying to classify. In Figure 2.2 is presented an example of feature maps extracted at different depths from a convolutional neural network.

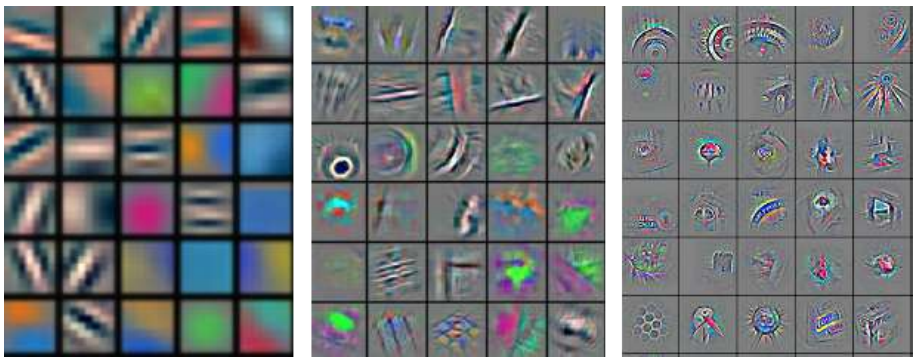

Figure 2.2: Filters estimated at different depths of a CNN. Moving from left to right, the depth of the CNN increased as well as the complexity of the patters captured by the estimated filters. The deeper the network, the higher the complexity of the features it can extract.

In the last few years, different types of convolutional networks have been crafted, improved and tested on food-related dataset. A popular CNN model called EfficientNet scores an accuracy of 92.98% on Food-101 with pre-training on ImageNet, and 96.18% pre-training on the enormous dataset JFT-300M (Foret et al. 2021). Also other types of popular CNNs have been used for food recognition, like ResNet (He et al. 2015) or Inception (Szegedy et al. 2014), scoring similar results. An infinite amount of results and experiments on food datasets using CNNs can be found in the literature, but two main aspects are common between all the studies: the accuracy is proportional to the complexity of the models and the size of the dataset used for pre-training or fine-tuning has a strong impact on the results.

In addition, several studies have tried to supplement or modify high-performance convolutional architectures to exploit the common aspects that all food photos share. In Qiu et al. 2022 the authors introduced PAR-Net, a convolutional network architecture that utilizes adversarial erasing (Wei et al. 2018) and class activation maps (B. Zhou et al. 2015) to integrate local and global features and highlight discriminative regions in food images. PAR-Net consists of three convolutional networks: P-Net,that operates on the complete image, A-Net, that operates on the segmented image containing the piece of food and R-Net that operates on the residual image (everything that is not considered the main dish). The proposed method enhances the network's ability to recognize specific food items by erasing regions and focusing on the remaining regions. Class activation maps are used to identify the regions contributing most to the predicted class label. It achieves state of the art performance on three popular food datasets and surpasses baseline methods consistently.

The previous examples underline the fact that CNNs have been the dominant architecture for

image classification tasks for many years. The reason behind the success of CNNs is the potential to obtain good performances even with relatively small architectures. CNNs are specifically designed to exploit the local spatial structure in images, making them highly effective at capturing local features and patterns. However, in the last years researchers began to explore the application of transformers in computer vision. Transformers were originally introduced for natural language processing tasks, but their architecture was recently changed and adapted to make these type of models applicable on different types of inputs. The idea was to leverage the self-attention mechanism of transformers to capture long-range dependencies and global context within images. The class of transformers applied to computer vision takes the name of vision transformers. Vision transformers have recently shown promising results in the domain of food recognition, even surpassing the performances of CNNs on various tasks.

At date of writing, vision transformers are starting to be used in food-related tasks. For example, in the task of semantic segmentation the best results on the most cited dataset (FoodSeg103) are obtained using a vision transformer called SEgmentation TRansformer (Zheng et al. 2021). However, the studies on the application of these type of models on food data are still few, even on a popular task like food recognition. An other example of the use of vision transformer on food can be found in Y. Zhou et al. 2022. The authors propose a so-called semantic center guided window attention fusion framework (SCG-WAFM) for food recognition. Food images, taken from Food-101, are fed into a pre-trained Swin transformer, which assigns a label to the image. Then, using the self-attention mechanism, the discriminative region that should contain the dish is extracted and fed into the same Swin Transformer. The two predictions are then combined by a linear layer that produces the final prediction.

The application of transformers in the food domain has some advantages and drawbacks. First of all, food recognition models are obviously thought to be used in real contexts. In a real scenario, images can strongly differ for lighting conditions, prospective or equipment used to take the photo. For these reasons, robustness is an important aspect to consider when deploying deep learning models into the wild. Robustness in computer vision is usually measured as the degradation in performances against common corruptions, perturbations and shifts. In Paul and Chen 2021 is concluded that vision transformers significantly outperform CNN counterparts in terms of robustness. It can be concluded that vision transformers could be a valid alternative to convolutional networks, especially when estimated to be used as tools in a real world context.

However, it's important to note that transformers usually require larger amounts of training data compared to CNNs (Dosovitskiy et al. 2020), as they tend to have a larger number of pa-

rameters and require more diverse and numerous samples to generalize well. The results obtained by vision transformers in such a short time have surely demonstrated that these types of models have the potential to surpass CNNs, but the absence of a huge dataset composed of food images is a strong limitation. Also, the computational resources needed to train large transformers models are not available to everyone.

Chapter 3 will explain in more detail transformer architectures, the various types of models and expand on the advantages and disadvantages of these architectures.

2.2 Most used food image datasets

As pointed out earlier, models composed of many parameters such as convolutional neural networks and transformers require a large amount of data to be trained. As seen in the case of EfficientNet, pre-training on a much larger dataset such as JFT-300M leads to a significant increase in accuracy. The same conclusions are valid for datasets used for food recognition. Furthermore, for a model to perform properly in the real world, it must be trained a large number of different foods in order to be able to distinguish them. In addition, it's also necessary for each class to consist of a sufficient number of distinguishable and informative images. Thus, a sufficiently advanced and complex model is not enough, and a good amount of high-quality data is necessary to achieve successful results. As a result, all the most popular and effective datasets used for food recognition are characterized by a large number of classes and enough images for each dish.

The first dataset public dataset dedicated to automated visual food recognition was the Pittsburgh Fast-Food Image Dataset (PFID), introduced by M. Chen et al. 2009. It is composed of 1098 food images from 61 different food categories. In today's literature, the dataset is not commonly used given the scarcity of images, making it unsuitable for training deep models.

Today, one of the most widely used food recognition datasets in the literature is Food-101 (Bossard, Guillaumin, and Van Gool 2014). Food-101 is a dataset commonly used for benchmarking and designed for fine-grained food recognition tasks. The images were collected from the FoodSpotting application, a social platform where people uploaded and shared food images. The Food-101 dataset contains a collection of 101 categories with $1,000$ images per category, resulting in a total of $101,000$ images. Food-101 is popular due to its large-scale nature, diversity of food categories and high-quality images. However, the number of images is far cry from datasets commonly used to train large neural networks such as ImageNet-21K and JFT-300M.

From this need, the Food2K dataset was assembled in 2019. It consists of $2,000$ different dishes mainly from the east and China. About 500 images are available for each dish, bringing the dataset to a total size of one million images. The dataset is quite recent and has only been tested on some of the most popular convolutional architectures. An in-depth analysis of Food-101 and Food2K is given in chapter 4.

While these datasets are rich in images and widely used in literature, they lack of fine-grained information about the shown dishes. While food labels provide essential information about the dish and main ingredients, a comprehensive food dataset should go beyond just the label to encompass a wider range of information, like ingredients, nutritional information or weight. Having a dataset rich in information and images could pave the way for the estimation of models that could predict much more than just the label. However, the job of tagging each images with a ton of detailed information is very tedious. An example of a new dataset that is taking this route is Nutrition5k (Thames et al. 2021). Nutrition5k is a novel dataset of $5,000$ diverse, real world food dishes with corresponding video streams, depth images, component weights and high accuracy nutritional content annotations. The dataset was collected from Google cafeterias using a custom scanning rig that captured each dish incrementally as ingredients were added. The dataset includes detailed visual and nutritional data for each plate of food, fine-grained list of ingredients, per-ingredient mass, total dish mass and calories, fat, protein, and carbohydrate macronutrient masses.

The next steps in the literature of food datasets for classification will likely follow two paths:

- using and integrating datasets with simple labels such as Food2K to obtain an increasingly rich and diverse set of images;
- enriching the information associated with each image with increasingly rich metadata, associating information on macronutrients, weight, depth and other aspects with each photo.

3

Transformer Models

Contents

3.1 Introduction and fundamentals **17**
3.2 Vision transformers **21**
3.3 Explainability in Vision Transformers **28**

3.1 Introduction and fundamentals

Deep learning is currently experiencing remarkable advancements, revolutionizing many fields of application like natural language processing, computer vision, time series analysis and many others. The availability of large-scale labeled datasets and the improvements to the architecture and optimization allowed researchers to train deep networks with millions of parameters, enabling them to learn complex representations of data and achieve state of the art results in tasks related to various tasks like recognition, generation and segmentation.

In a context of continuous progress and development, a new type of architecture called transformers was proposed to solve problems related to sequential and infinite data, like time-series and texts. The birth of transformers can be attributed to the groundbreaking paper titled “Attention is All You Need” by Vaswani et al. 2017. The proposed architecture revolutionized the field of natural language processing (NLP) and became the foundation for numerous subsequent advancements in deep learning. Before the emergence of transformers, recurrent neural networks (RNNs) were the dominant models for sequential data processing. However, RNNs have limitations, in-

cluding difficulties in parallelization and capturing long-range dependencies effectively. Through the introduction of the attention mechanism, transformer achieved a better understating of the context, access long-range relationships, basically solving some of the limits of RNNs.

The first transformer architecture proposed in Vaswani et al. 2017 is shown in Figure 3.1.

The proposed model is characterized by an encoder-decoder structure, suitable to solve sequence-to-sequence problems like machine translation. Both the input and output are tokens (words) and all the intermediate operations are made on a numeric vector of prefixed size. The main elements of these architecture are:

- *input embedding*: input data is fed into the network, but not in its original form. Each token (word) is first converted into an embedding, which is a numerical vector of a fixed length d_{model}. In its first implementation, the conversion was performed by an embedding layer, which is a linear layer of trainable parameters. The d_{model} hyperparameter can be chosen arbitrarily and its value is directly proportional to the complexity of the network;

- *positional encoding*: an additional component that provides information about the order or position of tokens in the input sequence. Since the Transformer architecture does not inherently encode the positional information of tokens, the positional encoding helps the model understand the sequential nature of the data. The parameters governing positional encoding are also trainable and are added to the tokens given as input;

- *multi-head attention* (Figure 3.2): a key component of the Transformer architecture that allows the model to capture different types of relationships and dependencies between tokens in an input sequence. The input embedding is transformed into three different linear projections, known as queries, keys, and values. The projection operations are learnable. After converting each of the initial embeddings into three vectors, the attention scores matrix is calculated using the following formula:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_{model}}})V$$

  An attention score measures the similarity between a query vector and all the other key vectors. The paper further refined the self-attention layer by adding a mechanism called "multi-headed" attention which allows the model to jointly attend to information from different representation sub-spaces at different positions replicating the same attention mechanism

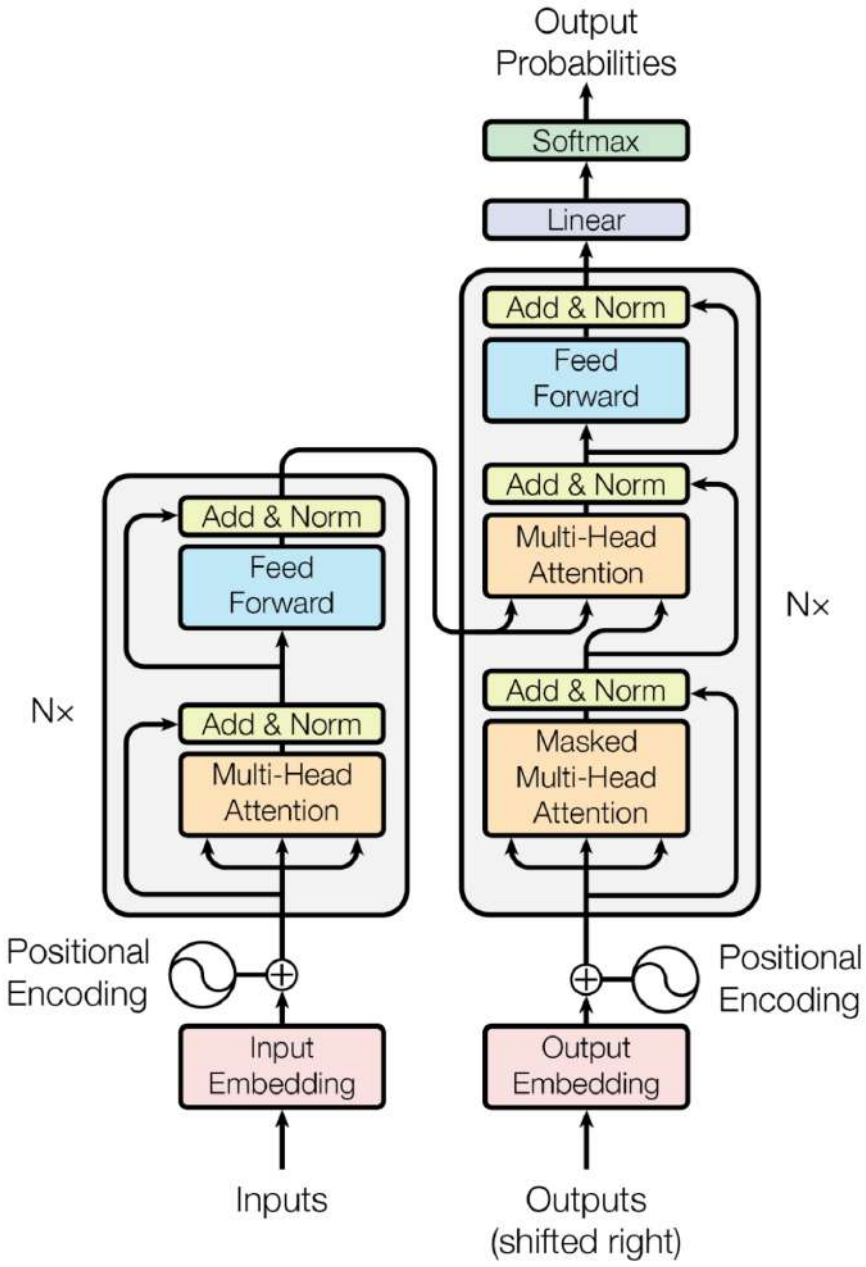

Figure 3.1: The architecture proposed in Vaswani et al. 2017 for natural language processing tasks. It follows an encoder-decoder structure, being composed of $N\times$ encoder blocks and $N\times$ decoder blocks.

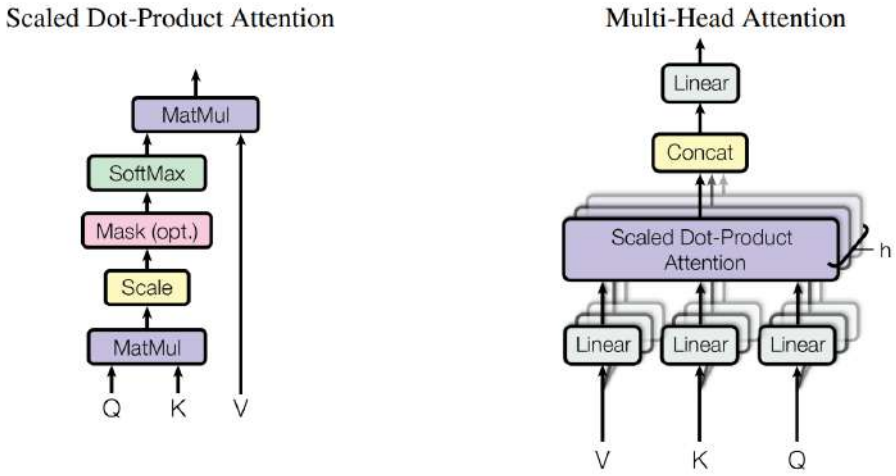

Figure 3.2: (left) Scaled Dot-Product Attention. (right) Multi-Head Attention consists of several attention layers running in parallel.

and calculation N different times. The attention scores obtained from different heads are then concatenated using the following formula:

$$MultiHead(Q, K, V) = Concat(head_1, \ldots, head_h)W^O$$

- *feed-forward*: a type of layer where each neuron or unit is connected to every neuron in the previous layer. It's applied to each token separately and identically.

The elements that make up this architecture follow a logic of their own, not necessarily related to the machine translation task or natural language processing. The flexibility of its component parts and the success of the transformer model in text-related tasks has spurred researchers to explore its structure and modify it to make it suitable for other problems. An example of this development is BERT (Bidirectional Encoder Representations from Transformers), introduced in Devlin et al. 2019. BERT is a state of the art language model that was introduced by researchers at Google AI. It revolutionized the field of natural language processing by demonstrating exceptional performance in various tasks such as text classification, named entity recognition, question answering and sentiment analysis. The architecture of BERT, shown in Figure 3.3, is made by stacking together only encoder layers, meaning it is an encoder-only model. The general usage of BERT involves taking advantage of its pre-trained language understanding capabilities and fine-tuning it on specific tasks. The input text is tokenized, embedded and then processed by the model.

In section 5.3 will be explained how BERT is used to give a vectorial representation to food

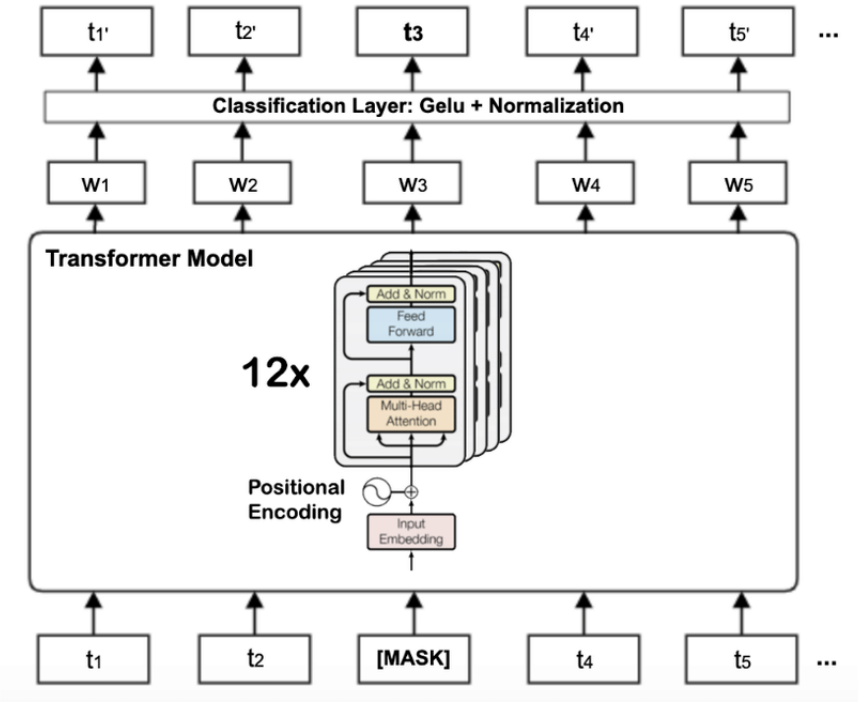

Figure 3.3: An example of the architecture of BERT proposed in Devlin et al. 2019. In this example, it is used to classify which word is most suitable for the phrase to be substituted in place of the token [mask].

labels, so that the resulting vector can be used to enrich the information associated to the image inputted in the model.

3.2 Vision transformers

The extensive use of transformers in natural language models inspired researchers to explore their applicability in computer vision tasks. The first transformer specifically designed for image processing is called Vision Transformer (ViT) and was introduced in "An image is worth 16x16 words" by Dosovitskiy et al. 2020. Until 2020, the convolutional models were completely dominant in the computer vision field. CNNs are designed to exploit the spatial structure of images and are inherently translation invariant. This made convolutional architectures really effective in all computer vision tasks. In the proposed Vision Transformer architecture, traditional "convolutional approaches" were replaced with self-attention layers and positional encoding, allowing it to capture spatial relationships and long-range dependencies in images. The first Vision transformer architecture intentionally followed the original Transformer as closely as possible. The architecture proposed in the paper is shown in Figure 3.4, while the introduced models and their specifications are presented

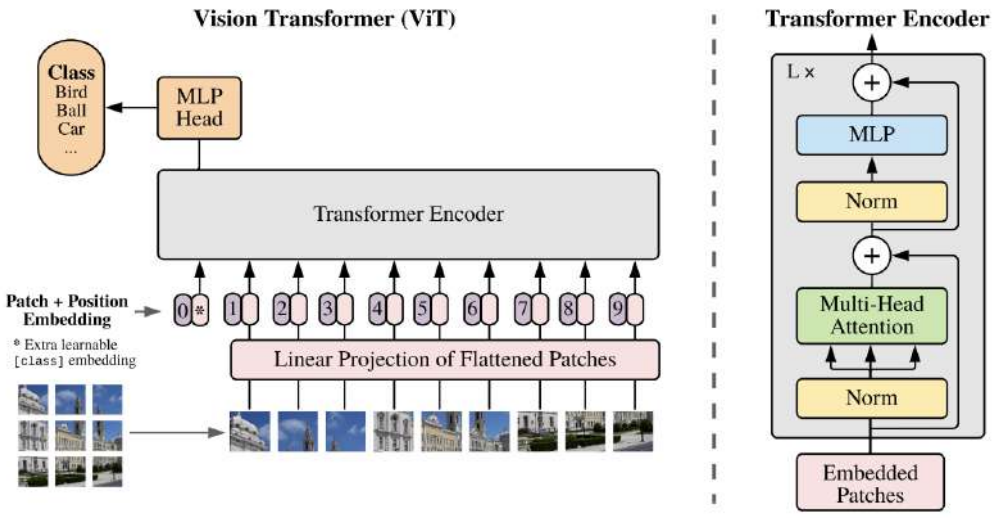

Figure 3.4: The architecture proposed in Dosovitskiy et al. 2020: (left) The encoder ViT takes in input an image, splits it in patches and is transformed by the encoder, (right) layers that compose a transformer encoder block.

Table 3.1: Details of Vision Transformer model variants introduced in Dosovitskiy et al. 2020 and their performances on ImageNet-1K (models are pre-trained on JFT-300M)

Model	Hidden size	Layers	Params	ImageNet Top-1 acc.
ViT-Base	768	12	86M	85.22%
ViT-Large	1024	24	307M	87.76%
ViT-Huge	1280	32	632M	**88.55**%

in Table 3.1.

The standard Transformer receives in input a 1D sequence of token embeddings. A Vision Transformer does basically the same thing, but the token in this context is not a word but a sub-portion of an image. A 2D image of dimension $X \in \mathbb{R}^{H \times W \times C}$ is reshaped into a sequence of flattened 2D patches $x_p \in \mathbb{R}^{N \times (P^2 \cdot C)}$, where (H, W) is the resolution of the original image, C is the number of channels, (P, P) is the resolution of each image patch. The Transformer uses constant latent vector size d_{model} through all of its layers. The parameters P, d_{model} and the number of layers are proportional to the complexity of the model. The value assigned to P regulates the patch size and directly affects the complexity of ViT. Each patch is treated as a token and processed individually by the transformer model. If the patch size is smaller, it results in a larger number of tokens in the input sequence which can be translated in a higher "training resolution". Consequently, a larger number of tokens can increase the computational and memory requirements of the model. Usually using smaller patch sizes results in a model that can capture fine-grained

details and local features more effectively. Increasing the size of the embedding increases the complexity of the model, as it must process larger inputs. However, a latent representation of higher dimensionality allows for more complex encoding. Finally, obviously increasing the number of layers leads to an increase in model parameters and model complexity.

Similarly to BERT's [class] token, a learnable embedding ($z_0^0 = x_{class}$) is added to the sequence of embedded patches. The hidden state at the output of the encoder part (z_L^0) serves as the image representation y. A classification head is attached to z_L^0, and uses only the embedding of the [class] token to classify. The classification head is implemented by a feed-forward block with one hidden layer . Also, positional embeddings are added to the patch embeddings to retain positional information.

Since the publication of the paper "An image is worth 16x16 words", research on Vision Transformers has increased exponentially. Several researchers have tested small modifications and optimizations to the original architecture that have led to improvements in classification performance. Here are a list of paper found some flat improvements to the the original ViT architecture:

- *"Going deeper with Image Transformers"*(Touvron, Cord, Sablayrolles, et al. 2021): the residual connection is modified by adding a so called "LayerScale", a learnable scalar weighting α on the output of residual blocks, and removing the pre-normalization:

$$\begin{aligned} x' &= x + \alpha SA(x) \\ x" &= x' + \alpha' FFN(x') \end{aligned} \tag{3.1}$$

- *"Scaling Vision Transformers"*(Zhai et al. 2022): the [class] token is removed and replaced by a Global Average Pooling of the embeddings of the image patches;

- *Scaling Vision Transformers to 22 Billion parameters* (Dehghani et al. 2023): the transformer encoder is parallelized. The multi-head attention and the MLP are not calculated sequentially but in a parallel fashion:

$$\begin{aligned} x' &= LayerNorm(x) \\ x'' &= x + FFN(x') + SA(x') \end{aligned} \tag{3.2}$$

As just mentioned, many of the trials on Vision transformers have tested some modifications to the original architecture to gain advantages in performance or training time. At the same time,

several researchers tried to analyze and find weaknesses in the ViT architecture and looked for alternative solutions that could solve them.

Here are some of the key limitations of the basic Vision Transformer:

- *computational complexity*: for a given image with $h \times w$ patches, a regular multi-head self-attention module has a computational complexity of $4hwC^2 + 2(hw)^2C$. Since the term hw is squared, the complexity of the regular self-attention module is quadratic with respect to hw. In general, quadratic complexity is not optimal and is avoided when possible because it undermines the scalability capabilities of an algorithm;
- *data-hungry*: Vision Transformer are extremely data-hungry. To be able to correctly generalize and achieve acceptable performances it's usually needed an huge dataset. In Dosovitskiy et al. 2020, the authors claim that the ImageNet dataset with 1.3M images and 1k classes, relatively big for standard computer vision tasks, is too small for transformers. Their suggestion is to pre-train transformers with the JFT-300M dataset, which contains 303M images and 18k classes and distill this knowledge for further tasks. Yet, not everyone has the computational resources needed to train on large dataset and JFT-300M is not publicly available;
- *unsuitable for fine-grained tasks*: traditional ViTs divide the input image into fixed-size patches and process them independently. This approach limits the ability of the model to effectively capture fine-grained spatial information, as in some cases the patches are relatively large compared to the details present in the image. This limitation is extremely severe in tasks such as semantic segmentation, where information per pixel is required.

One of the most popular and successful alternatives to the classical vision transformer that tried to solve its weaknesses is the Shifted-Window (Swin) Transformer. Swin Transformers were introduced in the paper "Swin Transformer: Hierarchical Vision Transformer using Shifted Windows" by Liu et al. 2021. The architecture proposed in the paper is shown in Figure 3.5, while the introduced models and their specifications are presented in Table 3.2.

The input is first split into non-overlapping patches like ViT. Each patch is treated as a "token" and is projected to an arbitrary dimension d_{model} using a linear embedding layer. In the default Swin implementation, the patch size is set at 4×4, while in the base ViT implementation the patch size was set at 16×16. After the embedding, several Swin transformer blocks are applied on these patch tokens. A Swin transformer block is composed of a multi-head self attention module with regular (W-MSA) or shifted (SW-MSA) windowing configuration, a 2-layer MLP with GELU

Table 3.2: Details of Swin Transformer model variants introduced in Liu et al. 2021 and their performances on the 1000 classes of ImageNet.

Model	Hidden size	Layers	Params	ImageNet Top-1 acc.
Swin-T	96	12	29M	81.3%
Swin-S	96	24	50M	83.0%
Swin-B	128	24	88M	86.4% *
Swin-L	192	24	197M	**87.3**% *

* pre-trained on ImageNet-22K

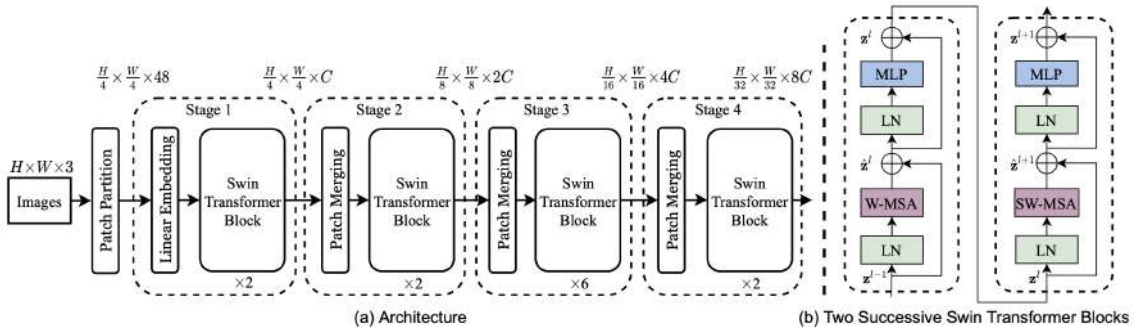

Figure 3.5: The architecture of a Swin Transformer (Swin-T) presented in Liu et al. 2021. (a) The architecture of the model (b) the structure of two successive Swin Transformer Blocks.

non-linearity, residual connections and normalization layers (LN). W-MSA is a key component of Swin Transformers, that differs from MSA due to the fact that attention is calculated "locally". Attention is not computed globally, but only between patches within a window of size $M \times M$, chosen arbitrarily.

The global computation leads to quadratic complexity with respect to the number of tokens, making it unsuitable for many vision problems. The W-MSA solve quadratic complexity problem using a "local" attention calculation. On the other hand, treating patches in groups of $M \times M$ means that there's a lack connections across windows, which limits its modeling power. To introduce cross-window connections while maintaining the efficient computation of non-overlapping windows, a shifted window partitioning approach is proposed, which alternates between two partitioning configurations in consecutive Swin Transformer Blocks.

With the shifted-window partitioning approach, consecutive Swin Transformer Blocks are computed as:

$$
\begin{aligned}
\hat{z}^l &= \text{W-MSA}(\text{LN}(z^{l-1})) + z^{l-1}, \\
z^l &= \text{MLP}(\text{LN}(\hat{z}^l)) + \hat{z}^l, \\
\hat{z}^{l+1} &= \text{SW-MSA}(\text{LN}(z^l)) + z^l, \\
z^{l+1} &= \text{MLP}(\text{LN}(\hat{z}^{l+1})) + \hat{z}^{l+1}
\end{aligned} \tag{3.3}
$$

where $\hat{z}^l$ and z^l are the output of the (S)W-MSA module and the MLP module for block l. To produce a hierarchical representation, the number of tokens is reduced by patch merging layers as the network gets deeper. The first patch merging layer concatenates the features of each group of 2×2 neighboring patches, and applies a linear layer on the $4C$-dimensional concatenated features. This reduces the number of tokens by a multiple of $2 \times 2 = 4$ ($2\times$ down-sampling of resolution), and the output dimension is set to $2C$. In this way, these type of models are capable of capturing both fine-grained and more general aspects of the image. Using the shifted window approach, Swin Transformers resolve the computational complexity and fine-grained processing problems. Unfortunately, these type of model share the identical drawback of ViT, they are extremely data-hungry.

Another very popular variant of Vision Transformers proposed by Touvron, Cord, Douze, et al. 2021 seeked to address this inability to generalize sufficiently in the absence of large amounts of data. The proposed architecture, namely Data-Efficient Image Transformers (DeiT), follows closely the original ViT architecture. The image is divided in patches of size 16×16 and a special [class] token is added for classification purposes. The model is then structured as a series of encoder blocks, that are identical to the blocks seen in Figure 3.4. The real addition in the architecture is the use of a special distillation token. The distillation token interacts with the class and patch tokens through the self-attention layers. It's is employed in a similar fashion as the class token, except that its objective is to reproduce the (hard) label predicted by a chosen teacher, instead of the true label. A teacher is a strong network that already achieves good results on the studied task and that can guide the "student" model in its training.

The student model is not only trained based on its capabilities to predict the true label, but it's also trained to replicate as closely as possible the prediction of a strong "teacher". It has been observed that using a ConvNet teacher gives better performance than using a transformer teacher. In Touvron, Cord, Douze, et al. 2021 ResNet is used as a teacher. The loss used to train DeiT

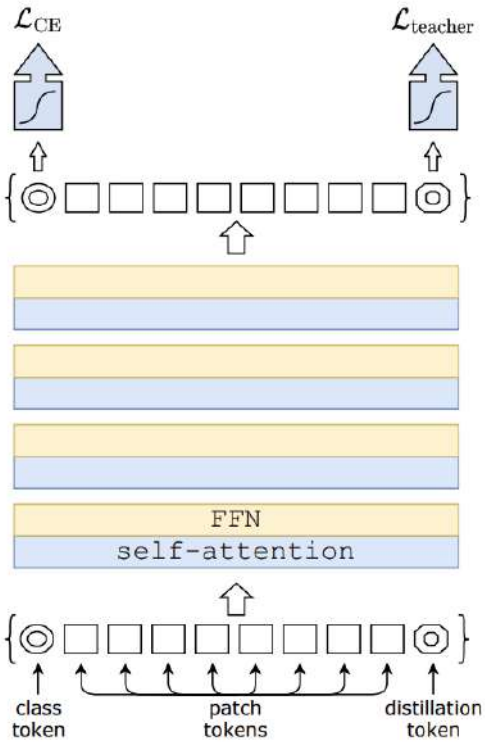

Figure 3.6: The architecture of DeiT model presented in Touvron, Cord, Douze, et al. 2021. The class token is used to predict the true label while the distillation token is used to emulate the prediction of a chosen "teacher" model.

Table 3.3: Details of DeiT model variants introduced in Touvron, Cord, Douze, et al. 2021 and their performances on ImageNet-1K.

Model	Hidden size	Layers	Params	ImageNet Top-1 acc.
DeiT-Ti	192	12	5M	74.5%
DeiT-S	384	12	22M	81.2%
DeiT-B	768	12	86M	**84.5**%

through hard-label distillation is:

$$L_{total} = \frac{1}{2}\mathcal{L}_{CE}(\psi(Z_s), y) + \frac{1}{2}\mathcal{L}_{CE}(\psi(Z_s), y_t)$$

where $\mathcal{L}_{CE}$ is the categorical cross-entropy function, y is the ground truth label, y_t is the label predicted by the teacher and $\psi(Z_s)$ is the label predicted by the student model. The use of the student-teacher approach has proven to be successful. DeiT is the first neural network without convolutional layer that can achieve competitive results against the the of the art on ImageNet with no external data. DeiT outperforms ViT by over 7% accuracy on ImageNet-1K without pre-training on JFT or ImageNet-22K.

In chapter 6, three of the classes of models presented will be trained on different food image datasets. Specifically, Swin-B (88 million parameters), ViT-B (86 million parameters) and DeiT-B (86 million parameters) will be fine-tuned on Food2K, starting from ImageNet-21K pre-trained weights. The fine-tuned model will be tested on Food-101 to verify their generalization capabilities. Finally, the ViT model will be tested on a segmentation task to verify if it can be applicable on tasks that differ from classification.

3.3 Explainability in Vision Transformers

In the field of artificial intelligence and computer vision maximizing a metric (like accuracy) has usually been a primary focus when evaluating models and algorithms. The will to achieve an higher accuracy has been the driving force behind many advancements in machine learning and deep learning and it's the main thing to look for when choosing a model. However, as these architectures become increasingly integrated into various aspects of our lives, there is a growing recognition that performance alone is not the only thing to look for. Explainability, or the ability to understand and interpret the decisions made by deep learning models, has emerged as an important consideration. Models like CNNs and transformer have a completely different structure but can achieve similar performances. It's extremely important to understand the inner workings of both approaches to grasp their strengths, weaknesses and differences. Traditionally, convolutional neural networks (CNNs) have been the dominant architecture in computer vision, and their interpretability has been well-studied. Given the central role of convolutional neural networks in computer vision and the amount of hours that researchers have invested in improving these architectures, it's remarkable that transformer almost identical to those used in NLP are capable of such good performances on

image tasks. This raises fundamental questions on whether these architectures work in the same way as CNNs. This question is partially answered in Raghu et al. 2022 where surprisingly clear differences were found in the hidden features and internal structures of ViTs and CNNs. The significance of transformer networks and their established distinctions from CNNs requires the development of tools to visualize their inner workings. Such a visualization can aid in debugging the models, help verify that the models are fair and unbiased, and enable downstream tasks (Selvaraju et al. 2017).

Explainability in convolutional neural networks (CNNs) has been an active research area for years, and various method have been created to explain these type of architectures. A class of methods focus on producing a heat-map that indicates local relevancy, given an input image and a CNN. The term "relevancy" refers to the degree of importance of certain regions or features within an image. It's a measure of how relevant a particular area or element is in relation to the overall content of the image. In food-related studies obtaining this type of insights is particularly important, in order to understand if the model focuses on the right piece of food or ingredient to make its prediction. Most of these explainability methods belong to one of two classes: gradient methods and attribution methods (Chefer, Gur, and Wolf 2021).

Gradient methods are based on the calculation of gradients with respect to the input of each layer, as computed through backpropagation. The gradient is then multiplied by the input activations to obtain a mask where high value should represent key area used for the prediction process. This method and all its variants are class agnostic: similar outputs are obtained, regardless of the class used to compute the gradient that is being propagated.

One of the most popular and used method proposed to solve the class agnostic problem is Grad-CAM (Selvaraju et al. 2017). The GradCAM method is a class-specific approach, which combines both the input features and the gradients of a network's layer. Being class-specific, and providing consistent results, this method is used by downstream applications, such as weakly-supervised semantic segmentation. Formally, the method calculates the class discriminative localization map $L^c_{Grad-CAM}$, which assigns a relevancy score to each pixel. To compute the map, it's first calculated the gradient of the score for class c, y^c, with respect to the feature map activations A_k of a convolutional layer as $\frac{\partial y^c}{\partial A^k}$. The gradients are then global-average-pooled over the height and width dimension using the formula:

$$\alpha^c_k = \frac{1}{K} \sum_i \sum_j \frac{\partial y^c}{\partial A^k_{ij}}$$

where α_k^c is defined as the neuron importance weights for class c for feature map k. This weight represents a partial linearization of the deep network downstream from A_k, and captures the "importance" of feature map k for a target class c. Finally, to obtain the final class-discriminative localization map a weighted combination of forward activation maps is computed, passing it to a ReLU activation function to filter out only the positive contribution to the class c:

$$L^c_{Grad-CAM} = ReLU(\sum_k \alpha_k^c A^k) \tag{3.4}$$

Grad-CAM relies on the gradient information flowing through convolutional layers. However, transformers do not have traditional convolutional layers and the method's computation is based only on the gradients of the a specific layer, usually the deepest one. The result, obtained by up-sampling these low-spatial resolution layers, is coarse. Grad-CAM are still usable with transformers, but the method was not thought for this usage and can perform quite poorly.

There aren't many contributions that explore the field of visualization for transformers and most of them employ the attention scores themselves. This practice ignores all the other components of the network that perform other types of computation, like the MLP block or residual connections. As underlined by Raghu et al. 2022, skip connections are crucial elements of transformers that have an high influence on its inner workings and in the way these architectures propagate features. Not taking into account these elements when calculating relevancy maps gives misleading and poor results. The main challenge in assigning attributions based on attentions is combining non-linearly from one layer to the next. The rollout method, proposed in Abnar and Zuidema 2020, combines attention from different layers, assuming that attentions can be combined linearly. It's observed that this method often leads to an emphasis on irrelevant tokens since even average attention scores can be attenuated.

A second class of methods, the Attribution propagation methods, are justified theoretically by the Deep Taylor Decomposition (DTD) framework. Such methods decompose, in a recursive manner, the decision made by the network, into the contributions of the previous layers, all the way to the elements of the network's input. The Layer-wise Relevance Propagation (LRP) method, propagates relevance from the predicated class, backward, to the input image based on the DTD principle. This assumes that the rectified linear unit (ReLU) non-linearity is used. Since transformers typically rely on other types of activations function, this method need some modification to be used with transformers. The method proposed by Chefer, Gur, and Wolf 2021 employs LRP-based relevance to compute scores for each attention head in each layer of a transformer model. It

then integrates these scores throughout the attention graph, by incorporating both relevancy and gradient information, in a way that iteratively removes the negative contributions. To calculate the needed relevancy map, gradient and relevance a propagated through the layers of the transformer, with respect to a class $t \in 1...|C|$. Using the chain-rule, gradients are propagated with respect to the classifier's output y, at class t, namely y_t:

$$\nabla x_j^{(n)} := \frac{\partial y_t}{\partial x_j^{(n)}} = \sum_i \frac{\partial y_t}{\partial x_i^{(n-1)}} \frac{\partial x_i^{n-1}}{\partial x_j^{(n)}} \tag{3.5}$$

where $x^{(n)}$ is the input of layer $L^{(n)}$, where $n \in [1...N]$ is the layer index in a network that consists of N layers. Relevance is propagated following the generic Deep Taylor Decomposition, with some modification to address the presence of GELU non-linearity in vision transformers and to account only for elements that have a positive weighted relevance:

$$R_j^{(n)} = \mathcal{G}_q(x, w, q, R^{(n-1)}) = \sum_{\{i \ lvert(i,j) \in q\}} \frac{x_j w_{ji}}{\sum_{j' | (j',i) \in q} x_{j'} w_{j'i}} R_i^{(n-1)} \tag{3.6}$$

Given the equations 3.5 and 3.6, it's possible to associate to each attention map $A^{(b)}$ its gradients $\nabla A^{(b)}$ and relevance $R^{(n_b)}$, with respect to the class t, which is not necessarily the correct label. The final relevancy map $C \in \mathbb{R}^{s \times s}$ is calculated as the weighted attention relevance:

$$\begin{aligned} \overline{A}^{(b)} &= I + \mathbb{E}_h(\nabla A^{(b)} \odot R^{(n_b)})^+ \\ C &= \overline{A}^{(1)} \cdot \overline{A}^{(2)} \cdot \dots \cdot \overline{A}^{(B)} \end{aligned} \tag{3.7}$$

where $\odot$ is the Hadamard product, and $\mathbb{E}_h$ is the mean across the "heads" dimension.

In Figure 3.7 are shown the results of the application of various methods to derive a relevancy map from an input image. Depending on the input image, the method can strongly vary on performances, but the transformer explainability method (labeled as "ViT exp." in the figure) seems to give more consistent and robust results. For this reason the Transformer explainability method proposed in Chefer, Gur, and Wolf 2021 will be used in section 6.3 to "explain" the prediction made by Vision Transformer and DeiT models.

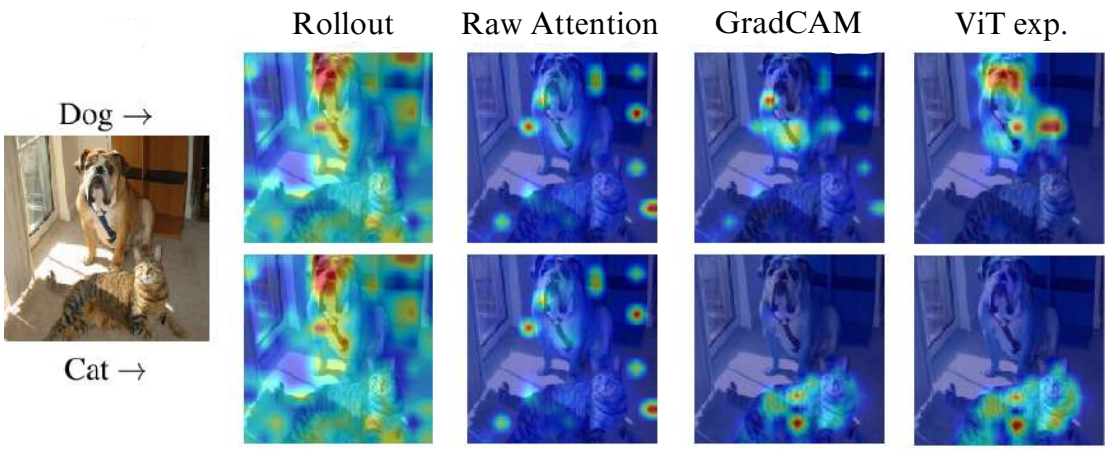

Figure 3.7: Examples of the relevancy maps produced using different explainability methods on Vision Transformers. Since all these methods are class-specific approaches, the produced relevancy map can change based on the predicted label.

4

Data

Contents

4.1 Food2K **33**
4.2 Food-101 . . . . . **36**
4.3 UECFoodPix **39**

4.1 Food2K

Vision transformers are data-hungry: to achieve the same performance as convolutional networks they need to be trained on large amounts of data. Although there are variants that also work well with medium-sized datasets, such as DeiT or CCT (Hassani et al. 2022), the increase in performance is evident when these models are pre-trained on large datasets like JFT-300M or ImageNet-21K. Vision transformers are relatively new and the lack of large datasets that depict dishes has led to little use of these models compared to the more common convolutional networks. However, with the recent introduction of increasingly large and variability-rich datasets depicting food scenes and the introduction of training practices that make it easier to train these types of networks, it is possible to employ vision transformers for food recognition and test whether these types of models can equal or even exceed the results obtained with convolutional networks. This study fits neatly into this context and investigates the use of the latest vision transformers to the largest public dataset for food recognition: Food2K.

Food2K is a large food recognition dataset that contains $1,036,564$ images with $2,000$ cat-

egories, belonging to different super-classes, such as vegetables, meat, barbecue and fried food. The dataset Food2K was created by collecting food images from the catering website Meituan. The raw-large scale dataset consisted of approximately seventy million food images uploaded by catering staff from take-out online restaurants and users of Meituan. The creation and cleaning process involved three phases: food category vocabulary construction, food image collection, and labeling. To construct the food category vocabulary, a bottom-to-up method was adopted. Noisy and redundant labels associated with the images were cleaned and different labels belonging to the same food were aggregated. Secondary label aggregation was performed by selecting representative images for each food label, extracting their visual features, and retrieving similar images from the dataset. The closest 50 food labels from the retrieved images were considered as candidate labels, which underwent manual verification to generate the final aggregated labels. This process resulted in the vocabulary of $2,000$ food categories that make up the dataset. For each category in the vocabulary, food images were aggregated based on their mapping and used to retrieve additional images from the unlabeled dataset. Representative images were chosen for each food category, and their visual features were computed using Inception-ResNet. The visual similarity between these representative images and the candidate images was used to assign each image to a category and remove images with low similarity. Additionally, the dataset was enriched including both merchant-provided images and user-generated images, to enhance diversity. To annotate the food images, an initial collection was followed by duplicate removal. Then, reference images where used to annotate each category and remove unqualified images, such as unclear or occluded ones. Some images were manually review and re-annotated. Through iterative annotation and multiple inspections, the high-quality Food2K dataset was created. Some example of categories in Food2K and a corresponding image is shown in Figure 4.1. All images are colored and their size varies from 220 pixels to 597 for both dimensions. The number of images per category is in the range $[153, 1999]$, showing quite a larger class imbalance compared to other existing food datasets.

Margherita Pizza Black pepper steak Crab seed sushi

Figure 4.1: Four images from 3 of the 2,000 classes in Food2K. Even images that belong to the same class show an high diversity in colors, presentation, ingredients and many other features.

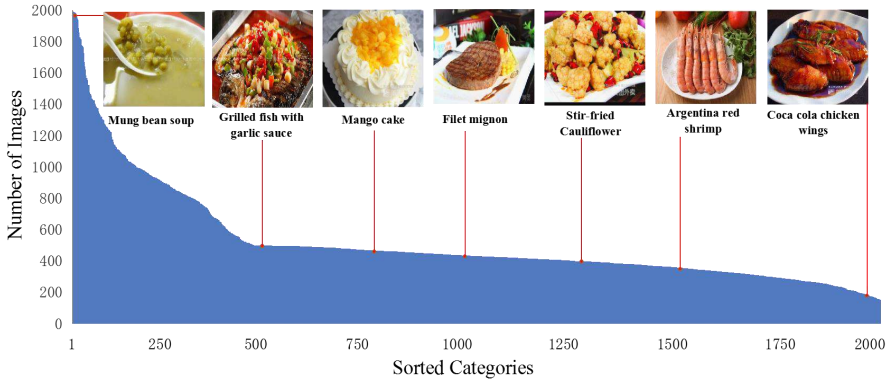

Figure 4.2: Distribution of the number of images in each class. The dataset is clearly long-tailed and half of the classes have less then 500 images.

Since the introduction of this dataset in march 2021, it has been used few times because its size makes it more difficult to manage compared to other smaller datasets. The researchers who published the dataset also provided some useful benchmarks testing some of the most popular convolutional models. The results of the proposed experiments and the benchmark values of accuracy are shown in Table 4.1. The procedures that were used to obtain the results shown in Table 4.1 will be further discussed in section 5.3, comparing them with the procedures proposed in this work. Other experiments were carried out on a sub-sampling of Food2K, called Food1K. Food1K consists of half as many categories and images as Food2K, respectively 500,000 images divided into 1,000 categories. In 2023, a competition was held on Kaggle (www.kaggle.com) and many researchers tried their hand at applying different models and methodologies to achieve the best possible prediction performance. No paper has been published on the results obtained in this competition, but the top ranked projects used vision transformer variants such as DeiT and Swin

Methods	Top-1 acc.	Top-5 acc.
VGG16	78.96%	95.26%
Inception v4	82.02%	96.45%
ResNet50	80.79%	95.74%
ResNet101	81.28%	95.99%
ResNet152	81.95%	96.57%
Inception-ResNet-v2	82.07%	96.74%
DenseNet161	81.87%	96.53%
SE-ResNeXt101 32x4d	80.81%	95.61%
SENet154	**83.62**%	**97.22**%

Transformer.

Chapter 5 will present a collection of experiments made on this dataset, the transformation and pre-processing used, augmentation and models estimated on it.

4.2 Food-101

Although Food2K is rich in variability and number of images and classes, alone is not sufficient to assess the goodness of models and the quality of estimated weights. In addition to checking the performance of vision transformers on Food2K , Food-101 was used for two reasons:

- *generalization capabilities of ViT*: even if the accuracy of vision transformers is equal or higher to CNNs models, it's fundamental to evaluate the capabilities of these models when classifying images coming from other datasets. In the real context, the model will be applied on images that are quite different from those seen in the training phase. the test set is used to evaluate the accuracy of the model on unseen images, but this is often not sufficient. The images in the test set were collected in the same way as the training set and as a result they are similar to those already seen by the model and do not reflect a real context, where images can be truly different under several aspects. For this reason, applying the model on a totally new and differently structured dataset allows for a more accurate assessment of the generalization capabilities of vision transformers.
- *value of pre-training on Food2K*: once a model is chosen, it's always preferable not use random weights, but start from a set of weights and biases already optimized for a similar task. As seen in section 2.1, pre-training the model on an extremely large dataset such as JFT-300M or ImageNet and then performing a fine-tuning step on the dataset of interest leads to a significant increase in performance. When choosing a set of pre-trained weights to

start our personal experiments, two aspects are important: the weights have been estimated on a sufficiently big dataset and the images used to train the model are close to the images we expect to see in the context in which we want to apply that model. The first condition is usually met by using public weights like the ones estimated on ImageNet. Having weights that met both of the condition at the same time is far less likely, but it can strongly boost performances. Finding the perfect dataset for pre-training can have beneficial results on all food computing tasks. A secondary aim of this study is to test whether the weights obtained by pre-training on Food2K can be a viable alternative to ImageNet weights for food-related tasks.

In chapter 5 it will be presented a series of experiments to test whether vision transformer generalize well and the utility of pre-training on Food2K. To achieve these objectives it's obviously needed a dataset different from Food2K, so Food-101 was chosen.

Food-101 is a large-scale dataset introduced in "Food-101 – Mining Discriminative Components with Random Forests" by Bossard, Guillaumin, and Van Gool 2014. It consists of 101,000 images covering 101 food categories. Each image is labeled with a specific food category, providing fine-grained annotations. The dataset captures real-world variability by including variations in viewpoints, lighting conditions and scales, reflecting the challenges encountered in practical scenarios. In order to address the need for a real-world food dataset, the researchers collected the dataset by leveraging the website foodspotting.com. foodspotting.com allowed users to take images of the food they were eating, annotate the place and type of food, and upload this information online. The researchers utilized this platform to download images and associated metadata for their dataset. They focused on the top 101 most popular and consistently named dishes on the platform. For each food category, the researchers randomly sampled 750 training images. Additionally, they collected 250 test images for each class. The test images underwent a manual cleaning process to ensure their quality and accuracy. In contrast, the training images intentionally remained uncleaned, retaining some amount of noise. This noise primarily manifested as intense colors and occasional incorrect labels. The inclusion of such weakly labeled data aimed to challenge computer vision algorithms to cope with real-world variations and errors. To standardize the dataset, all the images were rescaled to have a maximum side length of 512 pixels. Images smaller than this threshold were excluded from the dataset creation process. The dataset encompassed a wide range of visually and semantically similar food classes, including examples such as apple pie, waffles, onion rings, mussels, edamame, paella and macarons. Food-101 has gained significant popularity in the research community. Its large scale, fine-grained labels, and real-world relevance make it

Figure 4.3: An image from 9 different classes in Food-101.

a standard benchmark dataset for evaluating food recognition models. Researchers widely adopt and reference the dataset, leading to the development and evaluation of numerous algorithms for food recognition.

In Figure 4.3 are shown some examples of images contained in Food-101 and the corresponding label. In Table 4.2 are reported some of the performance of popular CNNs on this dataset. In the cited table are reported the results of the convolutional nets trained starting from ImageNet weights, but also the results obtained using Food2K as dataset for pre-training. Many other examples and benchmarks for different training strategies can be found on the internet or in a lot of different studies.

Because of the way it is structured and its popularity as a benchmarking dataset, Food-101 is a perfect fit to fulfill the purposes described above. The dataset is strongly different from Food2K. The latter has a larger amount of categories that are “uncommon” or popular only in Eastern culture, while Food-101 has dishes that are much more common and known around the

Table 4.1: Accuracy on Food-101 obtained by some of the most popular convolutional models. These results are taken from Wu et al. 2021, the paper that introduced Food2K.

Methods	Top-1 Acc.	Top-5 Acc.
VGG16	79.02%	93.78%
+ Fine-tuned on Food2K	80.68%	94.45%
Inception v3	84.15%	96.11%
+ Fine-tuned on Food2K	87.61%	97.25%
Inception v4	90.00%	-
+ Fine-tuned on Food2K	**90.33**%	**98.18**%
ResNet50	84.50%	96.18%
+ Fine-tuned on Food2K	85.89%	96.66%
ResNet152	86.61%	96.95%
+ Fine-tuned on Food2K	87.58%	97.28%
DenseNet161	86.94%	97.03%
+ Fine-tuned on Food2K	88.22%	97.57%
SENet154	88.62%	97.57%
+ Fine-tuned on Food2K	89.68%	98.08%

world, especially in the western world. This difference will actually allow us to understand whether Food2k is a valid dataset to be used for downstream to other tasks and datasets, including Western ones, or its predominance of oriental foods is too much of a drawback.

4.3 UECFoodPix

As described in section 3.3, evaluating computer vision models from an explainability perspective usually translates in generating heatmaps using CAM (Class Activation Maps) or other transformer explainability techniques. However, after creating the relevancy maps that should highlight the pixels of the image that contributed most to the prediction, it's fundamental to evaluate them. Relying solely on visual inspection or "eye-balling" is insufficient and can be too subjective. To address this limitation, datasets thought for image segmentation can be used to derive useful metrics that can enrich the subjective evaluation. Much like other datasets created for supervised tasks, datasets used for image segmentation include a collection of images, each accompanied by a corresponding ground-truth annotation. These annotations provide pixel labels that precisely delineate the boundaries of the objects to be segmented. For this study, the chosen dataset to pursue an evaluation of the explainability side of transformer is UECFoodPix dataset, introduced by Ege and Yanai 2019 in the paper "A Large-Scale Benchmark for Food Image Segmentation". UECFoodPix is a comprehensive collection of images specifically curated for the purpose of food segmentation. It

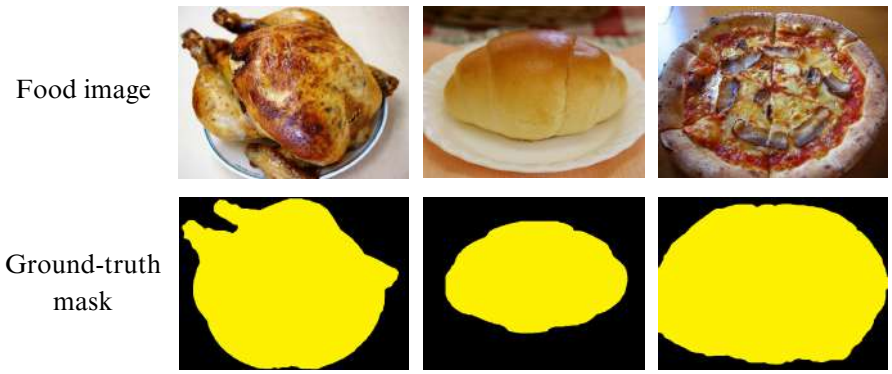

Figure 4.4: Three images and the corresponding segmentation from UECFoodPix. The mask highlights only the pixels that are food, considering as background everything else (like plates or cutlery).

contains 10,000 images of 102 types of dished. Each image in the dataset is accompanied by meticulously annotated ground-truth mask, which delineate the boundaries of individual objects within the scene. An example of images and the corresponding mask is shown in Figure 4.4. The dataset covers a broad range of food categories, including fruits, vegetables, prepared dishes, desserts, and more, making it representative of a real-world scenario. The UECFoodPix dataset plays a crucial role in the thesis by providing a reliable and standardized benchmark for assessing the accuracy of the generated relevancy maps and evaluating the explainability of vision transformer in the context of food segmentation.

Having a dataset that has both the image and its annotated mask enables for the calculation of several metrics to evaluate the segmentation performance. The following are the most commonly used metrics in image segmentation and the ones that will be applied in the experiments:

- *Intersection over Union (IoU)*: one of the most used metrics in this field, calculated as:

$$J(A, B) = \frac{| A \cap B |}{| A \cup B |}$$

  where A represents the predicted region and B the ground truth. It basically measures the overlap between the predicted segmentation mask and the ground-truth mask;

- *Pixel Accuracy (PA)*: Another important metric is pixel accuracy, which is calculated using the following formula:

$$PA = \frac{\#\ of\ \ correct\ \ pixels}{\#\ of\ \ total\ \ pixels}$$

It can be simply interpreted as the percentage of correctly classified pixels in relation to the total number of pixels.

It's important to note that evaluating explainability and image segmentation are two different things. The models that will be used are not trained for segmentation, and the heatmap produced are not meant to replicate as closely as possible the ground-truth. Also, the produced relevancy map can vary a lot based on the method used to produce it and based on the inputs given. Figure **??** clearly shows how changing the explainability method and given label can radically change the relevancy map produced, even using the same exact model. Despite this high variance in results, using metrics is still better than just evaluate the maps using eye-balling.

5

Proposed Approach

Contents

5.1 Experiments and objectives **43**
5.2 Data preprocessing and regularization **44**
5.3 Models and training scheme **48**

5.1 Experiments and objectives

The experiments proposed in the following section are all based on training and fine-tuning the vision transformers presented in chapter 3: ViT, Swin Transformer and DeiT. The following work is based on the current studies on vision transformers, that give some guidelines and hints on how to correctly fine-tune these models on a specific task like food recognition. The following tasks are performed to answer the three questions posed in section 1.1:

1. *classification on Food2K dataset*: starting from ImageNet-21K pre-trained weights, ViT, Swin and DeiT are fine-tuned for a pre-determined number of epochs on Food2K in order to obtain the best possible accuracy on the 2,000 available classes;

2. *assessing the value of Food2K weights*: ImageNet-21K weights are publicly available for all the popular convolutional and transformer models and represent a good starting point for most computer vision tasks. This study aims to verify if weights estimated on Food2K are a better stating point then the commonly used ImageNet weights for food-related tasks. To

assess the viability of these weights, the previously cited transformer models are fine-tuned on the popular Food-101 dataset, starting from ImageNet-21K and Food2K weights.

3. *explainability for vision transformers*: other than evaluating models in terms of top-1 and top-5 accuracy, this study explores how vision transformer "see" and to which region of the dish they assign the highest importance. The Layer-wise Relevance Propagation (LRP)-based explainability method for transformers explained in section 3.3 is used to produce a relevancy map that highlights the pixels that contributed most to the prediction. The aim is to verify if the relevancy maps obtained using vision transformers that were trained on a food-centric dataset are preferable to the ones produced by a model trained on a generic dataset. Additionally, the relevancy maps are evaluated to verify if the trained ViT can also be used for weakly supervised food segmentation. The images and masks in UECFoodPix dataset presented in section 4.3 are used to compute IoU and Pixel Accuracy, metrics that allow the comparison between the predicted segmentation and the ground-truth. All the presented variants of ViT will be used to produce relevancy maps and evaluate which set of pre-trained weights works best.

The results obtained with vision transformers are compared with the benchmarks available on convolutional neural networks, both on classification and segmentation tasks. Section 5.2 and 5.3 will explain the transformations applied to the data and the training scheme used for transformer and convolutional models trained on Food2K.

5.2 Data preprocessing and regularization

Food2K dataset is first split in two sets: train set and test set. The train set is the portion of the dataset used to train a machine learning algorithm. It contains labeled examples, where both input data and their corresponding target outputs are provided. The test set, on the other hand, is a separate subset of the dataset that the model has not encountered during training. It serves as an unbiased evaluation set to assess how well the trained model can generalize to new data. These sets have been created using stratified sampling, in order to replicate the proportion of images per class in both training and test set. Of the $1,036,564$ images in Food2K, $620,124$ are used for training and the rest for testing purposes.

Convolutional networks and vision transformers expect a consistent input size to process the images effectively. So each picture in the dataset has been resized to 224×224 pixels. Using a larger

image size usually gives better results. However, increasing the image size leads to an increase in training time and required memory. Additionally, raw images are composed of pixels with varying scales (ranges of values). For example, one image may have a pixel value range from 0 to 255, and another may have a range of 0 to 150. Although not required, it's preferred to standardize the pixel values, centering their mean value around 0 and their variance to 1, to boost learning performance and make the network converge faster. Data normalization is done by subtracting the mean from each pixel and then dividing the result by the standard deviation. The distribution of such images should resembles a Gaussian curve centered at zero. The mean and standard deviation have been computed using only the images in the training set. Standardizing images in both size and values leads to faster convergence for most types of networks. However, utilizing other transformations and methods can further improve the model's performance. As reported in Steiner et al. 2021 and various other studies, vision transformers' accuracy is strongly influenced by regularization.

Regularization is a set of techniques used to prevent overfitting and help the model generalize better on unseen data. When training complex models with millions of parameters, regularization plays a crucial role in ensuring that the network doesn't merely memorize training data but instead constructs a model capable of effectively recognizing patterns and objects. The following regularization techniques were used:

- *dropout*: when using dropout, at every training iteration, every neuron has a probability p of being temporarily ignored (dropped out) for a single iteration. This technique reduces complex co-adaptations of neurons, since a neuron cannot rely on the presence of particular other neurons. Therefore, the perceptron is forced to learn more robust features that are useful in conjunction with many different random subsets of the other nodes. The probability p is a hyperparameter that is called dropout rate and it's set at the default value of 0.1 for all vision transformer models used;

- *stochastic depth*: stochastic depth is a regularization technique where layers of a deep neural network are randomly dropped with a probability of p during training, but are kept during testing. Unlike dropout, which randomly sets a proportion of individual neurons' activations to zero, stochastic depth randomly skips entire layers;

- *label smoothing*: label smoothing is used to improve the generalization capabilities and accuracy of trained models. An ideal network assigns a probability of 1 to the correct class and 0 to all other classes. However, this can lead to overconfident predictions and hinder the network's ability to generalize. Label smoothing introduces a small amount of uncertainty

into the ground truth labels during training. Instead of using a hard label of 1 for the correct class and 0 for other classes, label smoothing replaces these hard labels with a smoothed distribution. The typical smoothed distribution assigns a value slightly less than 1 to the correct class and distributes the remainder of the probability mass among the other classes. In the following study the value of the correct class is set at 0.9, while the remaining 0.1 is distributed among the remaining $1,999$ classes;

- *weight decay*: weight decay is used to add a penalization mechanism to the loss function by adding a regularization term to it, that discourages the use of extreme values for weights and biases. The cross-entropy loss is modified by including a regularization term in the following way:

$$L_{new} = L_{CE} + regularization\ \ term$$

where the regularization term is equal to:

$$regularization\ \ term = \frac{\lambda}{2m} \times \Sigma \|w\|^2$$

λ is the regularization parameter, m is the number of instances and w is the weight. An higher value of λ penalizes more strongly the use of high value for the parameters and forces weights and biases to decay towards zero;

- *data augmentation*: augmenting data means generating new instances by applying random transformations every time an image is used for training. Image augmentation techniques include flipping, rotation, scaling, zooming, change in lighting conditions, and many other transformations that increase the variety of images available for training. The following augmentation techniques were used when fine-tuning on Food2K:

  - *random horizontal flip*: it involves randomly mirroring an image along its vertical axis. Each image is flipped with a probability of 0.5. It's commonly employed in deep learning models to improve their robustness and performance;

  - *color jitter*: introduces random variations in the color of an image during training. It alters brightness, contrast, saturation and hue, providing more diverse data to improve a model's ability to handle different lighting conditions and color variations in real-world scenarios;

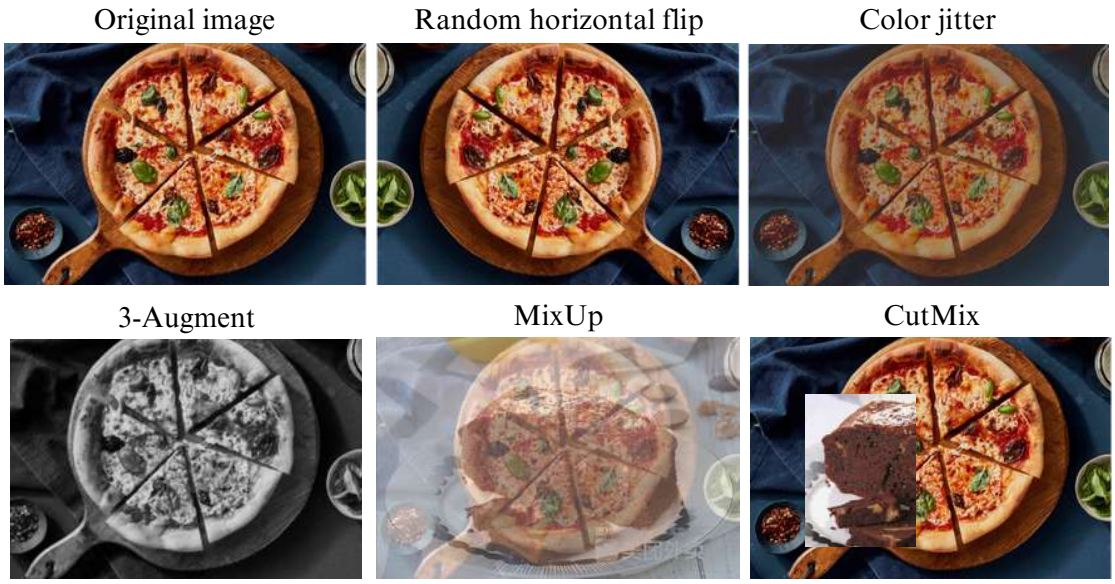

Figure 5.1: Examples show how the original image changes when augmentations are applied. MixUp and CutMix are the most drastic type of technique that combines two or more images.

– *3-Augment* (Touvron, Cord, and Jégou 2022): one augmentation between grayscale, solarization and gaussian blur is randomly applied to every image during training. The use of this augmentation has proven to be helpful when training ViT;

– *MixUp* (H. Zhang et al. 2017): it blends pairs of input images and their labels using weighted averages. Not only the image is changed, but also the associated label. This encourages the model to learn smooth decision boundaries, improving generalization and robustness;

– *CutMix* (Yun et al. 2019): it randomly crops patches from two images and pastes them together, blending the labels proportionally to the area of the patches. This reduces over-fitting and enhances model performance with limited data. As for Mixup, both the image and label are changed.

For the generalization task on Food-101 a simpler augmentation pipeline was used. The augmentations used are random Horizontal filp, color jitter and 3-augment. The images of Food-101 are also resized at 224×224 and normalized.

Model	Emb. size	Patch Size	Layers	Params	IN Top-1 Acc.
ViT-B	768	16x16	12	86M	85.7%
DeiT-B	768	16x16	12	86M	84.2%
Swin-B	128	4x4	24	88M	**86.4**%

Table 5.1: Specifics of the trained models. All models are used in their "base" configuration, which is the most commonly used form in the literature.

5.3 Models and training scheme

The vision transformers that are fine-tuned for food recognition are: Vision Transformer, Swin Transformer and Data-efficient image Transformer. Each of these models is used in the following experiments in their "base" form. The details of each model configuration are provided in Table 5.1. As explained in section 3.2, there are different configuration of the same models. For example, Swin transformers can be used in their "tiny", "small", "base" and "large" form. The bigger configurations are characterized by an higher number of parameters. For example, Swin-Large has 197 million trainable parameters compared to the 88 million parameters of Swin-Base. Usually, deeper models can obtain better performances, but training such an huge model is challenging even with multiple GPUs. For this reason, this study focuses only on the intermediate "base" configurations, which is the most popular configuration and a good compromise between complexity and performances.

Choosing the model to train is crucial, but another pivotal aspect during the training process is measuring the error. An error is measured through the selection of a function, called error function or loss function. A loss function measures of how "wrong" the neural network prediction is with respect to the expected output (the label). Optimization problems focus on defining an error function and trying to optimize the tunable parameters to get the minimum error. The error function expresses in a mathematical language which type of behaviour we want to penalize, and usually it's making less mistakes as possible. In this study, three different loss functions are used. Swin Transformer and Vision Transformer models are trained using two types of loss functions:

- *cross-entropy loss*: the models are trained using a cross-entropy loss that compares the label y_c with the corresponding predicted probability p_c:

$$L_{CE} = -\sum_{c=1}^{M} y_c \log(p_c). \tag{5.1}$$

The cross-entropy loss typically exhibits faster convergence compared to other loss functions and it's the standard choice when tackling classification problems with deep neural networks.

- *BERT-assisted training* (Y. Zhou et al. 2022): the models are trained using both the information coming from the image representation, and the label associated to the image. Given an image X_k and its label Y_k which represents the name of the dish, the embedding model *BERT* (Devlin et al. 2019) $e(\cdot)$ is used to produce a semantic embedding $t_k \in \mathbb{R}^d$ using the following formula:

$$t_k = \frac{1}{n}\sum_{i}^{n} e(y_{ki}), \tag{5.2}$$

where $y_{k1}, y_{k2}, \ldots, y_{kn}$ is a sequence of n tokens that composes the label Y_k. The multi-layer perceptron layer j produces an image feature x_{jk} of the input image X_k. This hidden feature is a vectorial representation in R^d, just like the representation t_k given to the label. The distance between these two representations can be used to learn key semantic information contained in the food label that may guide the prediction assigned to the image. The distance is calculated as:

$$\mathcal{L}_{emb} = \|x_{jk} - t_k\|^2 \tag{5.3}$$

and can be added to the cross-entropy loss to influence the training process during the backpropagation, obtaining the following total loss:

$$\mathcal{L}_{total} = \alpha\mathcal{L}_{CE} + \beta\mathcal{L}_{emb}, \tag{5.4}$$

where L_{CE} is the cross-entropy loss and α and β are the hyper-parameters that balance the influence of the two components. The aim is to obtain a similar latent representation for different images that share a similar name. In the following experiments α and β are set to 0.6 and 0.4 respectively. For ViT, BERT-base is used to produce embeddings of size 768, while for Swin Transformer BERT-large is used to obtain embeddings of size 1024. In Figure 5.2 is shown that summarizes what has been just explained.

While Swin transformer and ViT can be trained with a loss function of choice, DeiT is based on a loss function that follows the "student-teacher" paradigm. A "teacher" model, which is strong on the selected task, is used to guide the learning phase of the "student" model. To implement this interaction between the student and a teacher model the following loss, explained in section 3.2 is used:

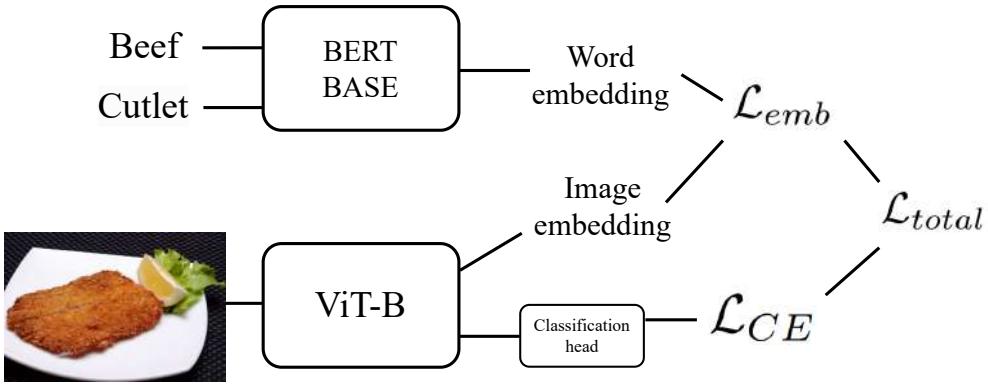

Figure 5.2: Loss computation using BERT. The label associated to the image is tokenized and embedded. The image is classified and an embedding of it it's extracted from the last layer before the classification head. The prediction is used to compute $\mathcal{L}_{CE}$, while the embeddings are used to compute $\mathcal{L}_{emb}$.

$$L_{total} = \frac{1}{2}\mathcal{L}_{CE}(\psi(Z_s), y) + \frac{1}{2}\mathcal{L}_{CE}(\psi(Z_s), y_t)$$

where the student model used is DeiT-B, while the teacher used is ResNet152. The ResNet152 model was pre-trained on Food2K for 200 epochs and Min, Wang, et al. 2023 reported that the model achieved an accuracy of 82% on Food2K test set. These weights were made public. However, when tested on Food2K this model with the pre-trained weights only achieves an accuracy of 62.22%. As will be seen in the following chapter, this negatively affected the performance of DeiT.

Essentially, in the following experiments five vision transformer models will be trained:

- ViT-Base using cross-entropy loss (86 million parameters, 5 hours and 50 minutes per epoch)
- ViT-Base using BERT assisted training (86 million parameters, 5 hours and 55 minutes per epoch)
- Swin Transformer using cross-entropy loss (88 million parameters, 6 hours per epoch)
- Swin Transformer using BERT assisted training (88 million parameters, 6 hours and 5 minutes per epoch)

- DeiT using distillation loss and ResNet152 as "teacher" network (86 million parameters, 5 hours and 50 minutes per epoch)

In the realm of neural network, the model plays a central role, but a successful classifier is built upon a delicate interplay of various key elements. These elements, ranging from optimizers and learning rates to training time, shape the trajectory of a model's learning process and ultimately determine its performance. The choice of which elements must be used and how to use them is not easy. Often times, researches introduce a new configuration that beats the state of the art, but it's not easy to pinpoint which exact change brought a boost in performances. Also, trail and error is usually the most successful strategy when searching for a working configuration.

For the following experiments, a configuration that brought good results was chosen by studying the state of the art, analyzing which choices were more common in successful studies and which elements are more effective when training vision transformers. Some of the hyperparameters were chosen through trail and error, while for others it was necessary to fix them at a certain value given the limits of memory and training time. Each of the models previously listed is trained using the following configuration:

- *optimizer*: an optimizer is a fundamental component in the field of machine learning and deep learning. It is an algorithm used to update the parameters of a model during the training process to minimize the loss function and improve the model's performance. In other words, an optimizer is responsible for guiding the learning process of a machine learning model, helping it to find the optimal set of parameters that best fit the training data and generalize well to unseen data. The optimizer used in this study is AdamW. AdamW is an optimization algorithm that extends the popular Adam optimizer by introducing a weight decay term to address potential weight decay issues in Adam. The algorithm was proposed in the paper titled "Decoupled Weight Decay Regularization" by Loshchilov and Hutter 2019. Weights and biases are updated using the following formula:

$$x_t \leftarrow x_{t-1} - \alpha \frac{\beta_1 m_{t-1} + (1-\beta_1)(\nabla f_t + w x_{t-1})}{\sqrt{v_t} + \epsilon} \tag{5.5}$$

  Adam and AdamW optimization algorithm are regulated by two hyperparameters, β_1 and β_2, which are set at their default value of 0.9 and 0.999. AdamW is one of the most used and successful optimizer at time of writing. Other promising optimizers that have shown performances capable of surpassing Adam and AdamW are LAMB, introduced by You et al.

2020, and Lion, by X. Chen et al. 2023. These two optimizers can bring faster and better convergence to a minimum for CNN and ViT, but are not recommended when the batch size chosen is too low (less than 64).

- *learning rate and scheduler*: the learning rate is a critical hyperparameter in the training process of neural networks. It determines the step size by which the model's parameters are adjusted during each iteration of gradient descent optimization. The learning rate governs the speed at which the model converges towards an optimal solution. A high learning rate can cause a faster convergence, but can lead to instability and divergence when it's too high. With a low learning rate the optimization will eventually converge to a minimum (global or, more likely, local) but it can slow down too much the training process if it's too low. Finding the learning rate value that is just right is an iterative process. Also, having a dynamic learning rate that decays after a certain amount of training time has shown to be effective. Since ImageNet-21K weights are already a good starting point, the initial learning rate is set to the relatively low value of 2×10^{-4}, and divided by half every 7 epochs. The aim is to slowly adjust the pre-trained weights and reducing the learning rate when the optimization process is close to a minimum, in order to avoid divergence and "bouncing" behaviours;

- *batch size and epochs*: batch size is another hyperparameter needed to be tuned in the optimizer algorithm. The batch size hyperparameter has a big effect on resource requirements of the training process and speed. A larger mini-batch size allows a computational boost that uses matrix multiplication in the training calculations. But that comes at the expense of needing more memory for the training process and generally more computational resources. In the following experiments, the batch size is set to a relatively low value of 16, since the memory requirements for models like Swin-B are quite high. All models were trained for 30 epochs, for a total of $1,162,860$ steps.

In section 6.1 will be shown how this configuration resulted in superior performances in a limited number of epochs. The trained vision transformers are compared with the only other available benchmark provided by the creators of Food2K. The available CNN benchmarks are trained for 200 epochs, using stochastic gradient descent as optimizer and a batch size of 2. The starting learning rate of their experiments is set at 1×10^{-2}, and divided by 10 after 30 epochs. Random horizontal flip and color jittering are used for data augmentation. The training and testing resolution is set at 224×224 pixels. To provide a meeting point between the different approaches, ResNet50 is trained using our configuration and the configuration proposed in the Food2K paper for 30 epochs.

6

Results

Contents

6.1 Performance on Food2K **53**
6.2 Generalization on Food-101 . . . . . **55**
6.3 Segmentation and explainability **58**

6.1 Performance on Food2K

The first step towards answering all the questions posed in section 1.1 is training the vision transformer models on Food2K, in order to estimate multiple predictors capable of recognizing food. Once these predictors are trained, they are used to:

- classify images belonging to Food2K test set in order to estimate their top-1 and top-5 accuracy and compare them. Top-1 accuracy is computed by dividing the total number of accurate predictions by the overall number of images in the test set. Similarly, the calculation of top-5 accuracy follows a parallel approach, with the distinction that a prediction is deemed correct if the right class resides within the initial five classifications that the model identifies as the most probable;

- assess their generalization capabilities by using them to predict images from other datasets (Food-101), and verify if a pre-training on Food2K can produce a better classifier;

- explain their prediction by visualizing the pixels of the images that the classifier considers important for the prediction. The produced relevancy maps are compared with a ground-truth mask using the metrics explained in section 3.3.

The estimated models can also be used for a variety of other purposes, like image retrieval or detection using bounding boxes.

Table 6.1 shows the performance obtained with the previously described vision transformers using the configuration explained in section 5.3. Top-1 and top-5 accuracy on the $416,440$ images used for testing are reported for each of the trained vision transformer. ViT-B and Swin-B are trained using cross-entropy loss and BERT-assisted training (section 5.3), while DeiT-B is trained using the student-teacher cross-entropy loss.

Table 6.2 shows the performance of popular convolutional models on Food2K, trained for 200 epochs using the configuration explained in 5.3 and reported in Min, Wang, et al. 2023. Finally, to provide a way for comparing the results, Table 6.3 shows a comparison of the performance of ResNet50 trained using the same configuration of the models from Table 6.1 and the configuration of the models from Table 6.2 for 30 and 200 epochs. These three tables used in conjunction allows us to compare different types of models trained with different configuration, and have a snapshot of the performances of vision transformers compared to CNNs.

Regarding vision transformers, the recognition performance is slightly superior in terms of top-1 and top-5 accuracy using ViT-B and Swin-B trained with BERT-assisted training compared to the other vision transformers shown. DeiT obtained an accuracy notably lower than the other two models. This is mainly caused by the lack of an available "good teacher" to guide the estimation of the model parameters. The ResNet152 pre-trained on Food2K only achieves an accuracy of 62.58% on Food2K test data, which makes this model not good enough to correctly guide DeiT model's training. The use of BERT-training did not have a positive effect on the performance of performance ViT, while it slightly increased top-1 accuracy for Swin-B.

Table 6.2 highlights the superior performances obtained by convolutional neural networks in terms of top-1 and top-5 accuracy. However, the results presented in Table 6.1 and 6.2 are not comparable, since the training procedures are totally different and the convolutional models were trained for more epochs.

Table 6.3 shows the accuracy achieved by ResNet50 using the two different configurations presented. The results clearly show that increasing the number of epochs can substantially improve

Models	Params	Top-1 Acc.	Top-5 acc.
ViT-B	86M	78.41%	**96.33**%
ViT-B + BERT	86M	74.82%	95.12%
Swin-B	88M	77.58%	96.17%
Swin-B + BERT	88M	**78.52**%	96.09%
DeiT-B	86M	73.45%	94.42%

Table 6.1: Performance on Food2K of ViT trained for 30 epochs.

the obtained performance. ResNet50 trained using the configuration proposed in Min, Wang, et al. 2023 achieves an increase in top-1 accuracy of 28% when trained for 170 more epochs. Also, ResNet50 has an accuracy of 62.22% when trained for 30 epochs using the configuration proposed in this study. This shows that the proposed configuration can bring an improvement in convergence time and possibly in final accuracy. Despite vision transformers showing slightly lower accuracy, an adjustment in the architecture, training time and configuration can possibly bring the accuracy of these models on par with or beyond convolutional models. Also, in 30 epochs, ResNet50 achieved a much lower accuracy then all the other vision transformers trained under the same conditions, which shows an advantage of vision transformers in term of convergence time over ResNet models. The "worst" vision transformer in terms of accuracy, DeiT-B, reached an accuracy of 73.45% in 30 epochs, which is much higher then the 62.22% achieved by ResNet50 in the same setting.

Table 6.4 illustrates the food categories on which the ViT-B model performed worst. Among the nineteen categories with the lowest accuracy scores, all of them have fewer than 500 images. This number is below the average number of images per class in the Food2K dataset (the average is 512 images per class). Food2K is a long-tailed dataset and this can affect the accuracy per class, with higher accuracy performance for categories that are represented by many images, and lower for less "populated" classes. This difference in accuracy based on category representation highlights the complex relationship between the dataset's distribution and the model's ability to make accurate predictions.

6.2 Generalization on Food-101

Food-101 is used to assess the generalization capabilities of vision transformers models, and to measure if fine-tuning on Food2K improves the classification ability of the models on Food-101. The models fine-tuned on Food2K are Swin-B and ViT-B using the two losses explained before. DeiT-B has been dropped since the model doesn't work properly without a valid teacher.

Models	**Params**	**Top-1 Acc.**	**Top-5 Acc.**
VGG16	136M	78.96%	95.26%
Inception v4	43M	82.02%	96.45%
ResNet50	26M	80.79%	95.74%
ResNet101	45M	81.28%	95.99%
ResNet152	60M	81.95%	96.57%
DenseNet161	29M	81.87%	96.53%
SENet154	256M	**83.62**%	**97.22**%

Table 6.2: Performance on Food2K of CNNs trained for 200 epochs.

Models	**Epochs**	**Top-1 acc.**	**Top-5 acc.**
ResNet50 (our config)	30	62.22%	87.81%
ResNet50 (Food2K config)	30	52.72%	81.24%
	200	80.79%	95.74%

Table 6.3: Comparison of ResNet50 performance on Food2K with three different training schedules.

Label	**Errors**	**Accuracy**	**# of Images**
Chocolate sundae	225	0.49	274
Stir-Fired pork	202	0.49	340
Artic bay sashimi	179	0.48	474
Fried rice	163	0.48	291
Chocolate cake	152	0.47	324

Table 6.4: The 5 categories on which the model scored the lower accuracy.

Models	Top-1 Acc.	Top-5 Acc.
ViT-B (IN-21k)	88.46%	98.05%
+ FT on Food2k	90.38%	**98.51**%
+ FT on Food2k (BERT)	**90.63**%	98.41%
Swin-B (IN-21k)	**92.67**%	98.95%
+ FT on Food2k	92.62%	98.96%
+ FT on Food2k (BERT)	92.59%	**98.98**%

Table 6.5: Performance on Food-101 using pre-trained ViT.

Models	Top-1 Acc.	Top-5 Acc.
VGG16 (IN-21k)	79.02%	93.78%
+ FT on Food2k	**80.68**%	**94.45**%
ResNet50 (IN-21k)	84.50%	96.18%
+ FT on Food2k	**85.89**%	**96.66**%
ResNet152 (IN-21k)	86.61%	96.95%
+ FT on Food2k	**87.58**%	**97.28**%
Inception V3 (IN-21k)	84.15%	96.11%
+ FT on Food2k	**87.61**%	**97.25**%
SENet154 (IN-21k)	88.62%	97.57%
+ FT on Food2k	**89.68**%	**98.08**%

Table 6.6: Performance on Food-101 using pre-trained CNNs.

The pipeline used to compare models on Food-101 is the following. ViT-B and Swin-B models were first trained on ImageNet-21K. Starting from these set of pre-trained weights, these models are fine-tuned on Food2K for 30 epochs, using the training scheme explained in section 5.3 and obtaining the results shown in Table 4.1. For each of the models there are two sets of pretrained weights: ImageNet-21K weights and Food2K weights. Starting from these two different sets of weights, Swin transformers and Vision Transformers are fine-tuned for 10 epochs on Food-101. The configuration used for fine-tuning on Food-101 is almost identical to the one used for fine-tuning on Food2K. AdamW is used as optimizer and the batch size is set at 16. The starting learning rate is fixed and set at 2×10^{-4}. Training resolution is set to 224×224 pixels, and the augmentation techniques randomly applied are 3-Augment, color jitter and random horizontal flip.

Table 6.5 shows the performance obtained by fine-tuning vision transformers on Food-101, while Table 6.6 shows the generalization capabilities of popular convolutional baselines. Two key conclusions can be drawn from the results:

- fine-tuning on Food2k does lead to a significant increase in accuracy on Food-101 for ViT-B model, while ImageNet-21K weights are a sufficiently good starting point for food recognition tasks when using Swin Transformer. Also, under the same training conditions, Swin transformers seems capable of generalizing much better where trained on different tasks, achieving an accuracy higher then ViT-B models;

- although CNNs had performed better on Food2K, when fine-tuned on Food-101 they tend to perform markedly worse than vision transformers. As reported in other fields, the generalization capabilities of vision transformers may prove to be superior to CNNs (Raghu et al.

2022);

- Table 6.5 also shows that using BERT-adjusted loss does not improve the generalization capabilities of proposed vision transformers.

6.3 Segmentation and explainability

As explained in section 3.3, various explainability tools can be used to produce relevancy maps that highlight pixels that most influenced the prediction of the model. An ideal model should make a correct classification and produce a relevancy map that attributes a high importance to the main ingredients shown in the image. The produced relevancy maps that will be shown in this section are calculated using models trained for classification purposes and not for segmentation. For this reason, the segmentation results are not comparable with models specifically trained for segmentation, but can give interesting insights on how the models presented really work and if they focus on the dish that are classifying.

The relevancy maps are produced adapting an LRP-based method for ViT (Chefer, Gur, and Wolf 2021) explained in section 3.3 and applying it on the trained ViT-B variants. The architecture used is the same (ViT-B) but the relevancy maps will be calculated using four different sets of weights: weights calculated by training on ImageNet-21K, weights fine-tuned on Food2K using cross-entropy and BERT-assisted loss and weights fine-tuned on Food-101. The aim is to verify if weights estimated on food-related datasets (Food2K and Food-101) produce relevancy maps closer to the ground-truth then ImageNet-21K weights. Basically, the expectation is that models trained on food recognition are better at recognizing food, and only focus on the main ingredients, ignoring everything that is background.

The relevancy maps are a matrix of size $H \times W$, which is the same as the one of the image inputted in the recognition model. For each position of this matrix, the method that produces the map assigns a value from 0 to 1, which represents the importance that the model gave to that pixel when classifying the image. The importance is calculated as a combination of the gradient and the activation, as explained in Chefer, Gur, and Wolf 2021 and in section 3.3. To segment the image, a fixed or dynamic threshold can be chosen. For example, if the importance of a pixel is higher then 0.5 then it's considered a piece of food and it's included in the mask, otherwise it's considered as background. This is called fixed thresholding. Otherwise, a dynamic thresholding method like Otsu's can be used. When using a dynamic threshold the value is not fixed for all the images, but

it's chosen by minimizing or maximizing a certain metric. Otsu's thresholding, also known as Otsu's method or Otsu's algorithm, is a widely used image segmentation technique that automatically determines an optimal threshold value for binarizing an image. It was developed by Nobuyuki Otsu in 1979 and is particularly effective for separating objects from the background in grayscale images.The basic idea behind Otsu's thresholding is to find a threshold value that minimizes the intra-class variance or maximizes the inter-class variance of pixel intensities in the two resulting classes (foreground and background) after binarization. In simpler terms, the algorithm tries to find a threshold that separates the two classes in such a way that the spread within each class is minimized, while the separation between the classes is maximized.

The UECFoodPix dataset is used to assess the quality of relevancy maps produced by the explainability methods. UECFoodPix is composed of $10,000$ images, all resized to 224×224 pixels. Each image in the dataset is accompanied by meticulously annotated ground-truth masks, which precisely delineate the boundaries of individual food objects within the scene. The dataset covers a broad range of food categories, including categories in common with Food2K and Food-101. In Figure 6.1 are shown some images and mask present in UECFoodPix and an example of the relevancy maps produced. The segmentations in UECFoodPix are compared to the maps produced using relevancy maps and thresholding using mean intersection over union (mIoU) and pixel accuracy. These two metrics are calculated for each of the 10000 images in UECFoodPix and then averaged to get an estimation of the average quality of the masks produced by each model.

Figure 6.2 shows four images from UECFoodPix, their ground truth, and the predicted masks obtained by applying Otsu's thresholding method on the relevancy map produced by ViT-B pre-trained on Food-101, Food2K and ImageNet. The same images are shown in Figure 6.3, but this time the segmentation mask is calculated by applying a fixed threshold of 0.1 onto the relevancy map. For some images, like the ones shown in the first two rows, the mask obtained by pre-training on Food2K seem to focus more on the actual dish rather than giving importance to the plate or the background. In complex images composed by multiple plates of food all models struggle to produce a sufficiently good mask, while in simpler images where there is only one dish the Food2K weights allow us to produce an accurate region that highlights the main dish. Pre-training only on Food-101 seems to not working as well as pre-training on Food2K. The produced mask are far from the ground truth even for simpler images like (a) or (b).

Using a low fixed threshold like 0.1 tends to produce more "inclusive" segmentations. The lower the value of the threshold the easier is for a pixel to be considered "foreground". Also, using a lower threshold tends to produce maps that score higher values in both mIoU and mPA.

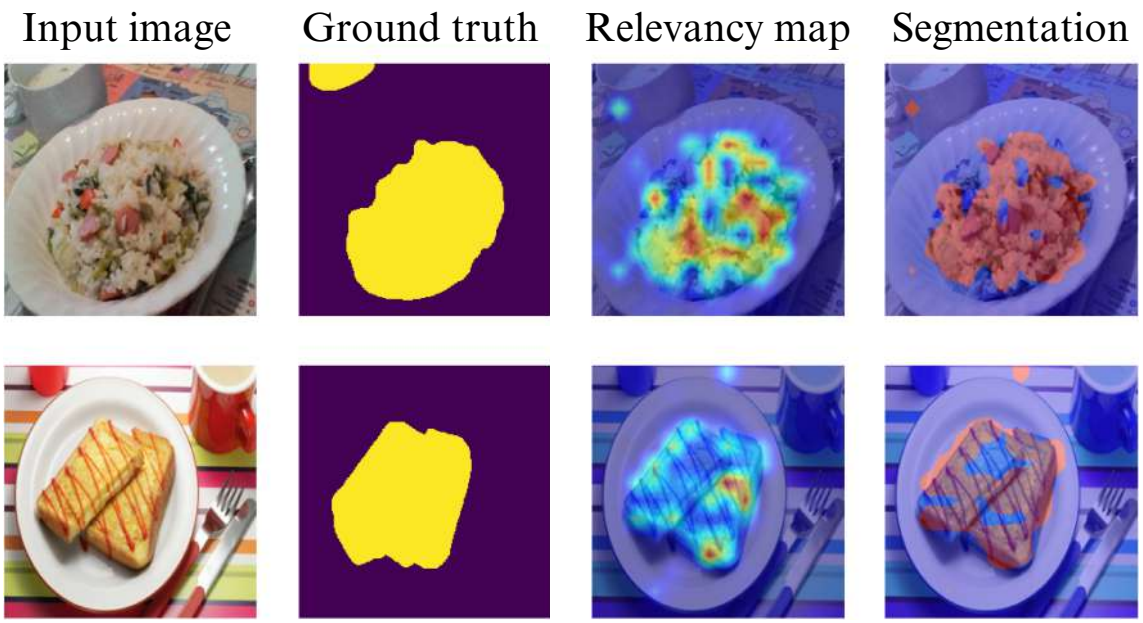

Figure 6.1: Examples of input image, ground-truth mask, relevancy map and segmentation obtained with Otsu's thresholding

Table 6.7 reports segmentation results of the ViT-B model used with different sets of weights. The mIoU values underline the gain in performance obtained by fine-tuning on Food2K. It's noticeable that pre-training on food-image datasets, and especially on Food2K, produces relevancy maps that are closed to the ground-truth masks. This is reflected both in the examples shown in Figures 6.2 and 6.3 in Table 6.7. The mIoU increases by 156% and the mPA by 112% when using Food2K weights. The use of BERT-training had a positive effect on the produced relevancy maps, improving both mIoU and mPA. In section 6.1, the classification performances of ViT-B trained using BERT were much worse compared to the accuracy obtained using cross- entropy loss. It's possible that while this training method doesn't have a positive effect on classification performances, it can improve the capability of the model of producing meaningful feature that highlight the right portion of a dish. Also, using a fixed and more "inclusive" threshold gives better mIoU and mPA values.

Figure 6.4 shows how the mean IoU and Pixel Accuracy change choosing different thresholds. Clearly, choosing a lower fixed threshold gives much better maps, while using a threshold that is too high produces maps that are way too selective.

In general, it can be concluded that models pre-trained on Food2K tend to be able to focus in a better way on the actual food in the image. Pre-training on Food-101 still brought an improvement to mIoU and mPA with respect to ImageNet-21K weights, but the results, both metrics-wise and

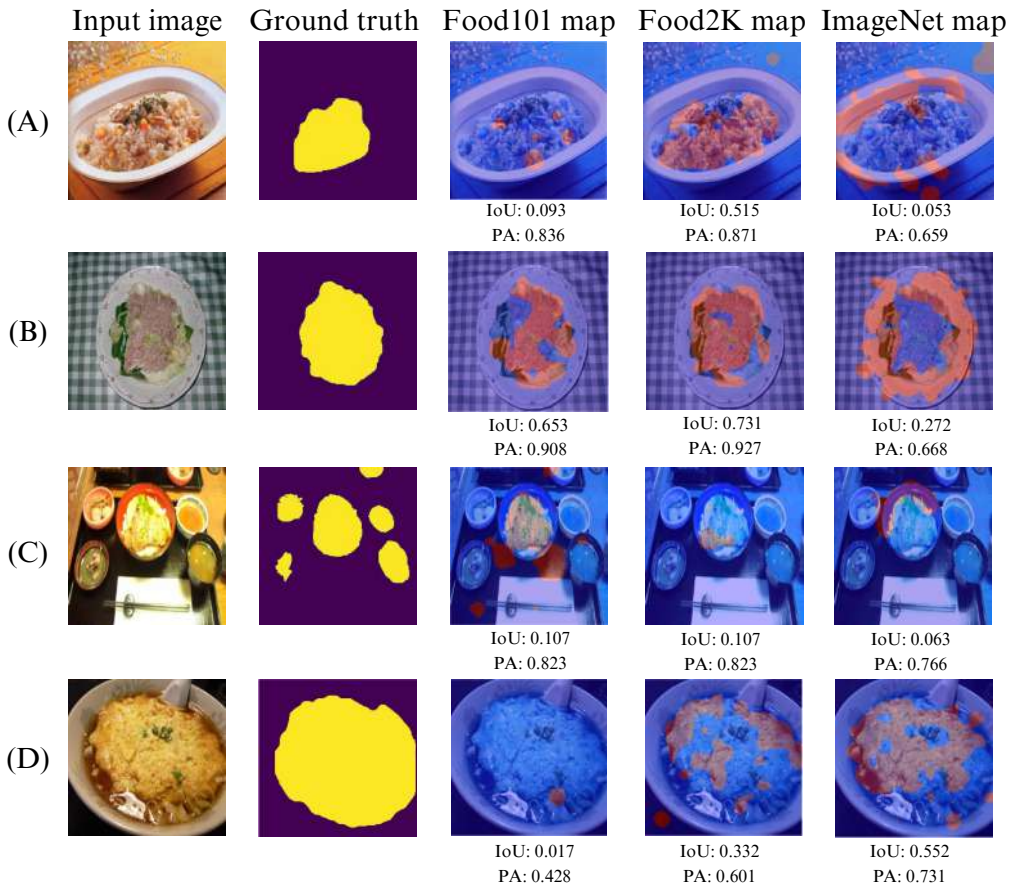

Figure 6.2: Four images of UECFoodPix, the corresponding ground-truth and predicted mask calculated using Otsu's thresholding. The goodness of the predicted mask is measured using IoU and PA.

Models	Otsu		Fixed at 0.1	
	mIoU	**mPA**	**mIoU**	**mPA**
ViT ImageNet-21K	0.257	0.639	0.412	0.670
ViT Food2K	0.402	0.722	0.604	0.802
ViT Food2K + BERT	**0.438**	**0.735**	**0.623**	**0.816**
ViT Food-101	0.32	0.683	0.508	0.765

Table 6.7: mIoU and mPA for the four ViT trained on different dataset and strategies. Vision Transformers trained on Food2K significantly outperform the models trained on ImageNet and Food-101 for all metrics and segmentation threshold.

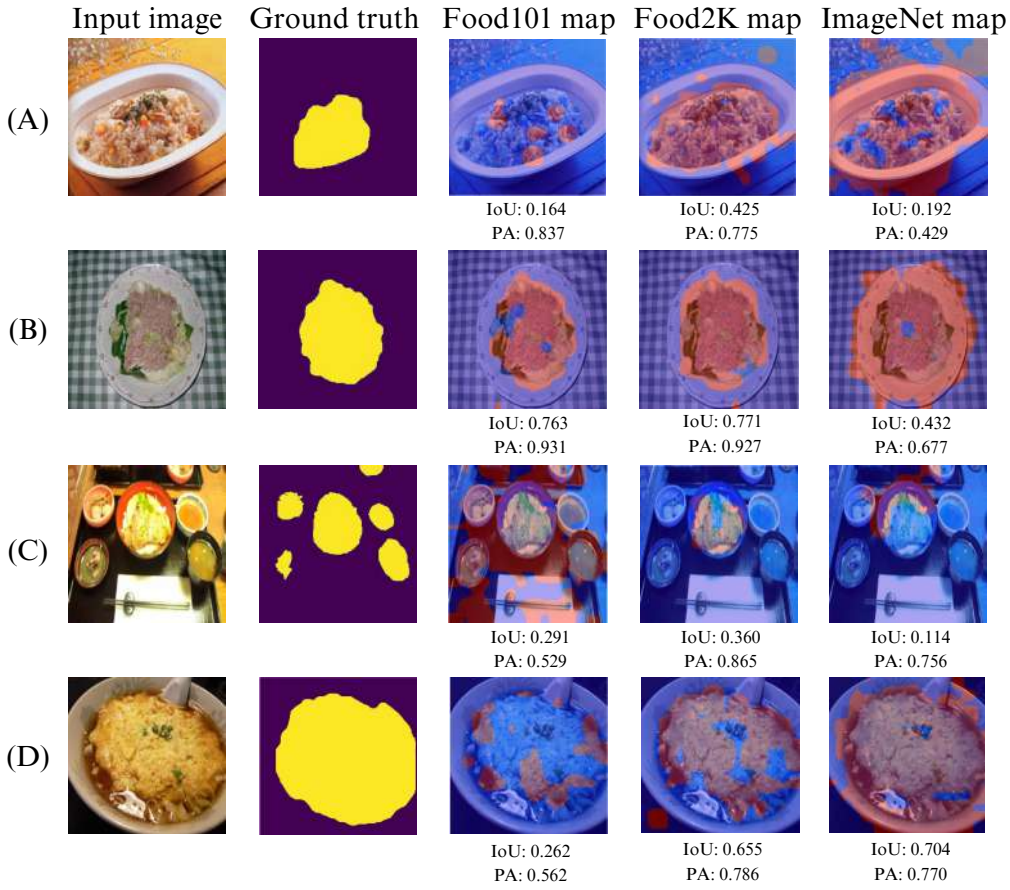

Figure 6.3: Four images of UECFoodPix, the corresponding ground-truth and predicted mask calculated fixing the threshold at 0.1.

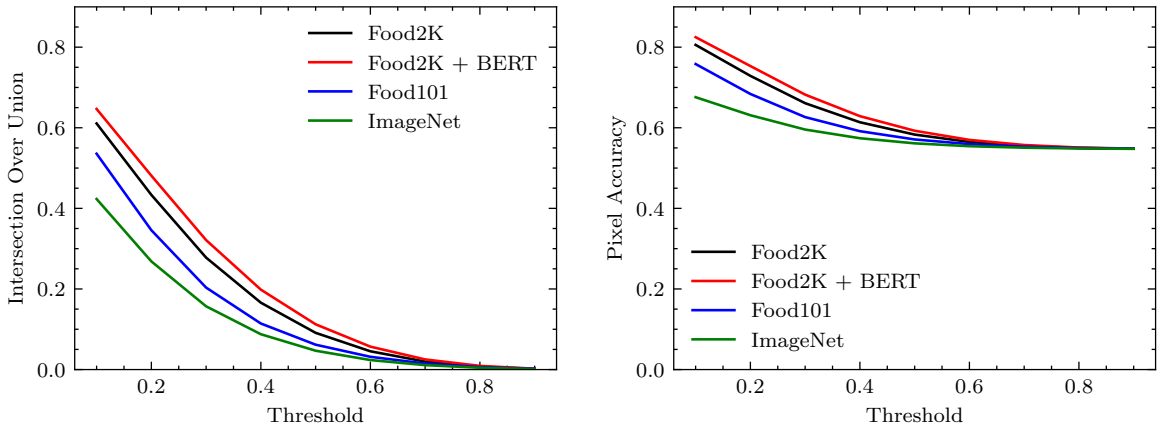

Figure 6.4: Change in Intersection over Union value (left) and Pixel accuracy (right) when choosing a different threshold used to categorize a pixel as relevant or not. An higher threshold means that it is harder to classify a pixel as relevant.

visually are much worse then the ones obtained with Food2K pre-trained transformers. Regarding explainability on vision transformer, it has been show both in this study and many other paper that it's possible and the relevancy map obtained can also be visually appealing. The decision made by vision transformers can be explained, highlight which region the network considers more important. The maps and the corresponding segmentation calculated by applying a threshold can function as a segmentation model, and be used for image segmentation purposes. That these types of model can easily be adapted for food segmentation, and pre-training on a food-image dataset, especially Food2K, is key to obtain even better performance.

Conclusions

In this thesis it has been demonstrated the effectiveness of vision transformers for food recognition tasks. ViT-B outperforms other models on the Food2K dataset, particularly when fine-tuned on the same dataset. Additionally, vision transformers showcase superior generalization capabilities on the Food-101 dataset compared to popular convolutional models. Furthermore, the study highlights the potential of vision transformers for food segmentation tasks through food recognition (i.e. without training for semantic segmentation). The explainability tools used to produce relevancy maps indicate their ability to identify regions of interest accurately, making them promising for segmentation purposes. Additionally, in this study it has been found that pre-training on Food2K can be beneficial for the results obtained in other food-related downstream tasks, like recognition on other food datasets or segmentation. Overall, vision transformers show promise in food computing applications, providing competitive performance and improved interpretability. This research paves the way for future investigations in utilizing vision transformers for various food-related tasks and advancing the field of computer vision in the context of food analysis.

Bibliography

Abnar, Samira and Willem Zuidema (2020). *Quantifying Attention Flow in Transformers*. arXiv: 2005.00928 [cs.LG].

Boser, Bernhard E., Isabelle M. Guyon, and Vladimir N. Vapnik (1992). "A Training Algorithm for Optimal Margin Classifiers." In: *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*. COLT '92. Pittsburgh, Pennsylvania, USA: Association for Computing Machinery, pp. 144–152. ISBN: 089791497X. DOI: 10.1145/130385.130401. URL: https://doi.org/10.1145/130385.130401.

Bossard, Lukas, Matthieu Guillaumin, and Luc Van Gool (2014). "Food-101 – Mining Discriminative Components with Random Forests." In: *Computer Vision – ECCV 2014*. Ed. by David Fleet et al. Cham: Springer International Publishing, pp. 446–461. ISBN: 978-3-319-10599-4.

Breiman, Leo (2001). "Random Forests." English. In: *Machine Learning* 45.1, pp. 5–32. ISSN: 0885-6125. DOI: 10.1023/A:1010933404324. URL: http://dx.doi.org/10.1023/A%3A1010933404324.

Chefer, Hila, Shir Gur, and Lior Wolf (2021). *Transformer Interpretability Beyond Attention Visualization*. arXiv: 2012.09838 [cs.CV].

Chen, Mei et al. (2009). "PFID: Pittsburgh fast-food image dataset." In: *2009 16th IEEE International Conference on Image Processing (ICIP)*, pp. 289–292. DOI: 10.1109/ICIP.2009.5413511.

Chen, Xiangning et al. (2023). *Symbolic Discovery of Optimization Algorithms*. arXiv: 2302.06675 [cs.LG].

Csurka, Gabriela (2004). "Visual categorization with bags of keypoints." In: *European Conference on Computer Vision*.

Dehghani, Mostafa et al. (2023). *Scaling Vision Transformers to 22 Billion Parameters*. arXiv: 2302.05442 [cs.CV].

Deng, Jia et al. (2009). "ImageNet: A large-scale hierarchical image database." In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255. DOI: 10.1109/CVPR.2009.5206848.

Devlin, Jacob et al. (2019). *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. arXiv: 1810.04805 [cs.CL].

Dosovitskiy, Alexey et al. (2020). *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*. URL: http://arxiv.org/abs/2010.11929.

Ege, Takumi and Keiji Yanai (2019). "A New Large-scale Food Image Segmentation Dataset and Its Application to Food Calorie Estimation Based on Grains of Rice." In: *Proc. of ACMMM Workshop on Multimedia Assisted Dietary Management(MADiMa)*.

Foret, Pierre et al. (2021). *Sharpness-Aware Minimization for Efficiently Improving Generalization*. arXiv: 2010.01412 [cs.LG].

Granlund, Goesta H. (1978). "In search of a general picture processing operator." In: *Computer Graphics and Image Processing* 8.2, pp. 155–173. ISSN: 0146-664X. DOI: https://doi.org/10.1016/0146-664X(78)90047-3. URL: https://www.sciencedirect.com/science/article/pii/0146664X78900473.

Hassani, Ali et al. (2022). *Escaping the Big Data Paradigm with Compact Transformers*. arXiv: 2104.05704 [cs.CV].

He, Kaiming et al. (2015). *Deep Residual Learning for Image Recognition*. arXiv: 1512.03385 [cs.CV].

Liu, Ze et al. (2021). *Swin Transformer: Hierarchical Vision Transformer using Shifted Windows*. arXiv: 2103.14030 [cs.CV].

Loshchilov, Ilya and Frank Hutter (2019). *Decoupled Weight Decay Regularization*. arXiv: 1711.05101 [cs.LG].

Martinel, Niki, Claudio Piciarelli, and Christian Micheloni (2016). "A supervised extreme learning committee for food recognition." In: *Computer Vision and Image Understanding* 148. Special issue on Assistive Computer Vision and Robotics - "Assistive Solutions for Mobility, Communication and HMI", pp. 67–86. ISSN: 1077-3142. DOI: https://doi.org/10.1016/j.cviu.2016.01.012. URL: https://www.sciencedirect.com/science/article/pii/S1077314216000436.

Min, Weiqing, Shuqiang Jiang, et al. (2018). "A Survey on Food Computing." In: *ACM Computing Surveys (CSUR)* 52, pp. 1–36.

Min, Weiqing, Zhiling Wang, et al. (2023). *Large Scale Visual Food Recognition*. arXiv: 2103.16107 [cs.CV].

Paul, Sayak and Chen (2021). *Vision Transformers are Robust Learners*. arXiv: 2105.07581 [cs.CV].

Qiu, Jianing et al. (2022). *Mining Discriminative Food Regions for Accurate Food Recognition*. arXiv: 2207.03692 [cs.CV].

Raghu, Maithra et al. (2022). *Do Vision Transformers See Like Convolutional Neural Networks?* arXiv: 2108.08810 [cs.CV].

Selvaraju, Ramprasaath R et al. (2017). "Grad-cam: Visual explanations from deep networks via gradient-based localization." In: *Proceedings of the IEEE international conference on computer vision*, pp. 618–626.

Smeulders, A.W.M. et al. (2000). "Content-based image retrieval at the end of the early years." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22.12, pp. 1349–1380. DOI: 10.1109/34.895972.

Steiner, Andreas et al. (2021). "How to train your vit? data, augmentation, and regularization in vision transformers." In: *arXiv preprint arXiv:2106.10270*.

Subhi, Mohammed Ahmed, Sawal Hamid Ali, and Mohammed Abulameer Mohammed (2019). "Vision-Based Approaches for Automatic Food Recognition and Dietary Assessment: A Survey." In: *IEEE Access* 7, pp. 35370–35381. DOI: 10.1109/ACCESS.2019.2904519.

Szegedy, Christian et al. (2014). *Going Deeper with Convolutions*. arXiv: 1409.4842 [cs.CV].

Tahir, Ghalib Ahmed and Chu Kiong Loo (2021). "A Comprehensive Survey of Image-Based Food Recognition and Volume Estimation Methods for Dietary Assessment." In: *Healthcare* 9.12. ISSN: 2227-9032. DOI: 10.3390/healthcare9121676. URL: https://www.mdpi.com/2227-9032/9/12/1676.

Thames, Quin et al. (2021). *Nutrition5k: Towards Automatic Nutritional Understanding of Generic Food*. arXiv: 2103.03375 [cs.CV].

Touvron, Hugo, Matthieu Cord, Matthijs Douze, et al. (2021). *Training data-efficient image transformers and distillation through attention*. arXiv: 2012.12877 [cs.CV].

Touvron, Hugo, Matthieu Cord, and Hervé Jégou (2022). "Deit iii: Revenge of the vit." In: *European Conference on Computer Vision*, pp. 516–533.

Touvron, Hugo, Matthieu Cord, Alexandre Sablayrolles, et al. (2021). *Going deeper with Image Transformers*. arXiv: 2103.17239 [cs.CV].

Vaswani, Ashish et al. (2017). "Attention is All you Need." In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon et al. Vol. 30. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Wei, Yunchao et al. (2018). *Object Region Mining with Adversarial Erasing: A Simple Classification to Semantic Segmentation Approach*. arXiv: 1703.08448 [cs.CV].

Wu, Xiongwei et al. (2021). *A Large-Scale Benchmark for Food Image Segmentation*. arXiv: 2105.05409 [cs.CV].

You, Yang et al. (2020). *Large Batch Optimization for Deep Learning: Training BERT in 76 minutes*. arXiv: 1904.00962 [cs.LG].

Yun, Sangdoo et al. (2019). "Cutmix: Regularization strategy to train strong classifiers with localizable features." In: *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 6023–6032.

Zhai, Xiaohua et al. (2022). *Scaling Vision Transformers*. arXiv: 2106.04560 [cs.CV].

Zhang, Hongyi et al. (2017). "mixup: Beyond empirical risk minimization." In: *arXiv preprint arXiv:1710.09412*.

Zhang, Yudong et al. (2023). "Deep learning in food category recognition." In: *Information Fusion* 98, p. 101859. ISSN: 1566-2535. DOI: https://doi.org/10.1016/j.inffus.2023.101859. URL: https://www.sciencedirect.com/science/article/pii/S1566253523001756.

Zheng, Sixiao et al. (2021). *Rethinking Semantic Segmentation from a Sequence-to-Sequence Perspective with Transformers*. arXiv: 2012.15840 [cs.CV].

Zhou, Bolei et al. (2015). *Learning Deep Features for Discriminative Localization*. arXiv: 1512.04150 [cs.CV].

Zhou, Yongxin et al. (2022). "Semantic Center Guided Windows Attention Fusion Framework For Food Recognition." In: *Pattern Recognition and Computer Vision: 5th Chinese Conference, PRCV 2022, Shenzhen, China, November 4–7, 2022, Proceedings, Part II*. Shenzhen, China: Springer-Verlag, pp. 626–638. ISBN: 978-3-031-18909-8. DOI: 10.1007/978-3-031-18910-4_50. URL: https://doi.org/10.1007/978-3-031-18910-4_50.