Research Article

A Region-Based Convolutional Neural Network Model for Quantitative Analysis of Carbohydrate Content in Foods

Zeki Oralhan[1*], Hüseyin Hakkomaz[2]

[1] Nuh Naci Yazgan University, Orcid ID: https://orcid.org/0000-0003-2841-6115,
E-mail: zoralhan@nny.edu.tr
[2] Erciyes University, , E-mail: hakkomazhuseyin@gmail.com
[*] Correspondence: zoralhan@nny.edu.tr

Received 20 March 2024
Received in revised form 14 September 2024
In final form 18 October 2024

Reference: Oralhan, Z., & Hakkomaz, H. (2024). A region-based convolutional neural network model for quantitative analysis of carbohydrate content in foods. The European Journal of Research and Development, 4(3), 57-74.

Abstract

Type 1 Diabetes Mellitus (T1DM) is a globally prevalent autoimmune disease, increasing in incidence annually. The condition necessitates continuous monitoring and meticulous record-keeping of dietary intake, with a strong emphasis on accurate carbohydrate counting. Recent advancements in computer vision have facilitated the estimation of nutritional content and values of meals, enabling calculations through both 2D and 3D image analyses. Significant progress in artificial neural networks has further enhanced the accuracy and efficiency of food recognition and volume estimation. In this study, we propose a system that estimates the carbohydrate content of meals by analyzing their diameters as input. The system demonstrated an average error rate of approximately 7%, with individual error rates ranging from 1% to 15%. While these variations are influenced by the presentation style of the food, the system's ability to achieve high accuracy from a single image highlights its effectiveness. Moreover, the system is adaptable for 3D volume estimation using multi-angle images, making it suitable for further development. With the integration of additional food categories and an expanded training dataset, the proposed system holds significant potential for practical application in dietary management and nutritional monitoring.

Keywords: *Machine Learning, Biomedical, T1DM, AI, CNN*

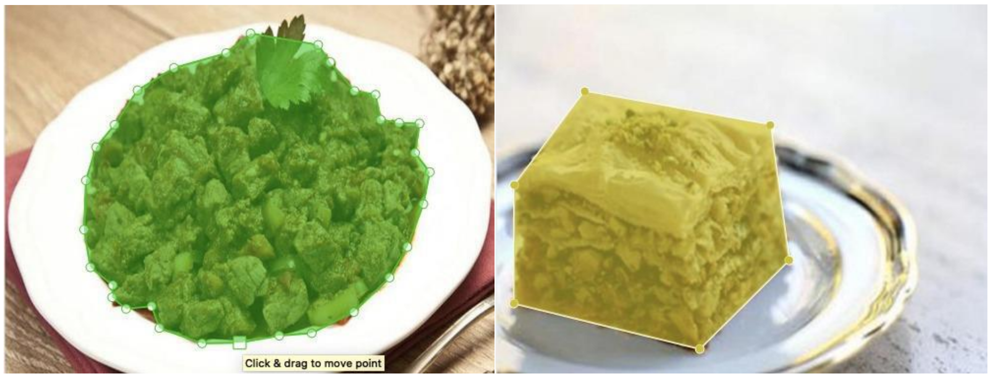
1. Introduction

One of the most significant health issues of our time, Diabetes Mellitus, is considered a global epidemic of the 21st century requiring urgent action, according to the International Diabetes Federation [1,2]. Type 1 Diabetes Mellitus (T1DM) is a chronic autoimmune disease characterized by insulin deficiency, followed by hyperglycemia. Medical nutrition therapy is crucial for preventing complications and ensuring metabolic control in T1DM. Additionally, through proper medical nutrition therapy, children with T1DM can grow and develop healthily while establishing adequate and balanced nutritional habits [3]. The global incidence of T1DM is increasing by approximately 3% annually [4]. The treatment of T1DM relies on maintaining good blood sugar control, which is an essential component of diabetes self-management. Carbohydrate consumption is the primary determinant of postprandial blood glucose levels. Current guidelines recommend that the amount of carbohydrates (CHO) in meals be considered when calculating insulin doses for meals [5]. Carbohydrate counting is a meal-planning method that focuses on estimating the content of macronutrients that primarily affect postprandial blood glucose responses [6]. Advances in food image processing and mobile technology have revolutionized dietary management for individuals with Type 1 Diabetes Mellitus (T1DM), particularly in estimating carbohydrate and sugar content. Studies such as Dehais et al. [7] and Alfonsi et al. [8] highlight the accuracy and efficiency of mobile-based carbohydrate counting systems compared to traditional methods, offering real-time solutions for glycemic control. Similarly, research by Anthimopoulos et al. [9] and Vasiloglou et al. [10] demonstrates the potential of computer vision and AI-powered tools like GoCARB in automating dietary assessments, reducing user burden, and enhancing self-management [8-10]. Contreras et al. [11] focused on enhancing the precision of dietary records in Type 1 Diabetes management by introducing active labeling correction of meal timings and carbohydrate types using mathematical models. Their study highlights the need for improved algorithms to accurately estimate the glycemic impact of different carbohydrate classes. Ladyzynski et al. investigated the accuracy of automatic macronutrient counting systems based on voice descriptions, specifically for people with Type 1 Diabetes. The study emphasized the potential of voice-based technologies to simplify dietary tracking while maintaining accuracy in estimating carbohydrates, proteins, fats, and calories [12]. Schmidt et al. conducted a systematic review evaluating the effectiveness of advanced carbohydrate counting techniques in improving glycemic control among Type 1 Diabetes patients. The review underscored the benefits of tailored dietary management strategies in achieving better metabolic outcomes [13]. Bell et al. examined the overall efficacy of carbohydrate counting in Type 1 Diabetes through a systematic review and meta-analysis. Their work provided robust evidence supporting carbohydrate counting as an effective method for optimizing blood glucose levels, particularly when integrated with individualized dietary guidance [14]. The advancement, use, and adoption of smartphone technology, along with the continuous growth of the entire mobile health ecosystem, have created opportunities for improved diabetes management. However, most current applications are often limited to requiring users to manually enter the food's name and, in some cases, specify the portion size.

2. Materials and Methods

The food images used in this study were not included in any pre-prepared dataset. Therefore, we had to create the data used for training ourselves. We downloaded the images used for training to our computer in sets of 200 by using a Python script provided with source which we developed to search for keywords on internet search engines. Subsequently, to prevent data contamination, we removed images unrelated to the keywords from our dataset.

After this step, we resized all images to 800x600 pixels using the Python script provided in with source to match the required pixel dimensions for training. However, these images were still not ready for training. To make them suitable, we used the LabelMe software, written in Python programming language, which we installed on a MacOS operating system. Using this method, we manually labeled each image one by one. Examples of labeled images are shown in Figure 1.



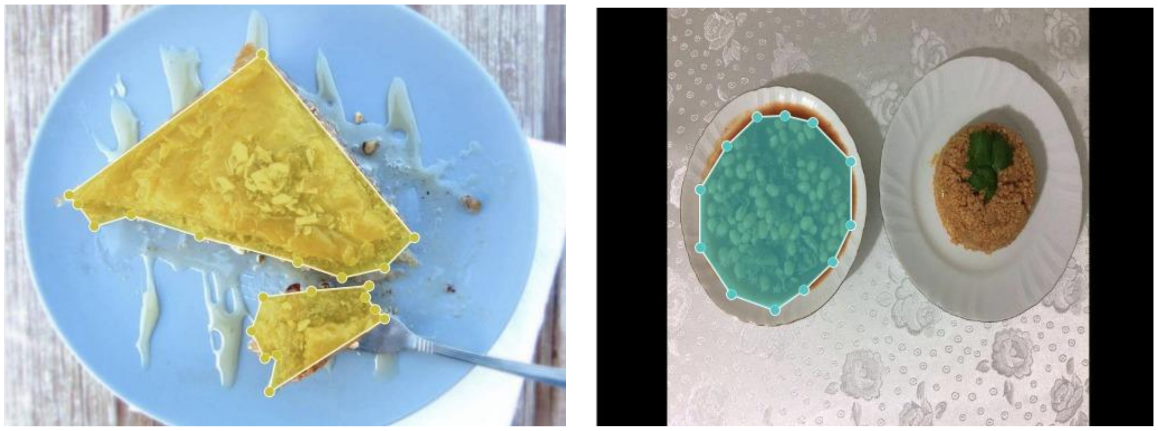


Figure 1: Labeled Foods

The number of labeled images classified into categories is provided in Table 1.

Table 1: Number of Labels for each Food

Class	Number Of Labels
Baklava	95
Meat Sauté	57
Rice Pilaf	81
Menemen	40
Chicken Sauté	65
Stuffed Grape Leaves	74
Kidney Bean Stew	14
Bulgur Pilaf	52
Dumplings (Mantı)	66
Lentil Soup	72
Trilece Dessert	51
Fresh Beans	87
Gilt-head Bream	60
Stuffed Eggplant	93
Dried Beans	31
Chickpeas	46
Reference Card	16

Figure 2 provides an overview of the basic steps in the system through a simplified flowchart. Images retrieved from the internet using specific keywords are first resized and then labeled with the assistance of labeling software. In our system, we created a dataset consisting of 17 classes, 906 images, and a total of 1000 labels. The subsequent sections of this chapter detail the operation of the ResNet50 model, which was employed as the training algorithm within the Detectron2 framework. Additionally, we explain the process of performing volume calculations using the trained model and conclude with the development of a web application as the final product.

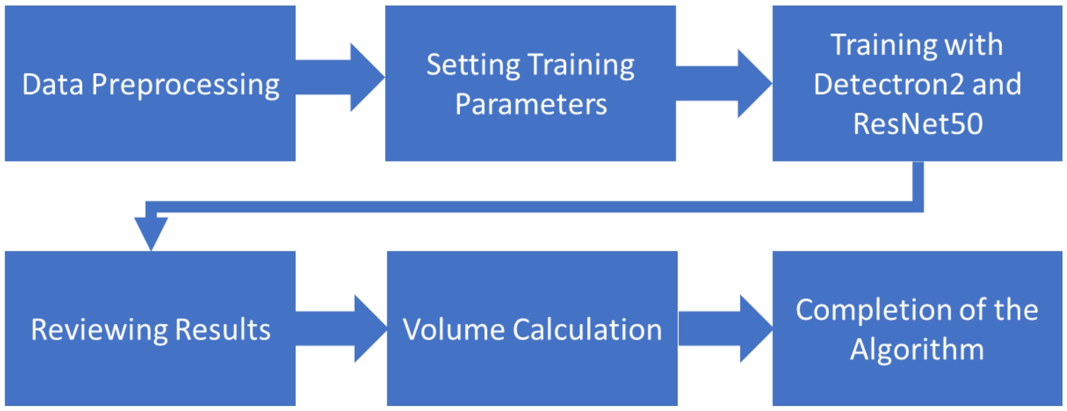

Data Preprocessing
Setting Training Parameters
Training with Detectron2 and ResNet50
Reviewing Results
Volume Calculation
Completion of the Algorithm

Figure 2: Diagram of System

2.1.Training

To create a system that produces outputs quickly and with high accuracy, several algorithms were tested. However, we conducted our training using the Detectron2 system, which allows for the development of a final product and the easy adaptability of model outputs with other algorithms. As discussed in Section 3.2.1.1, the Detectron2 system offers significant convenience when building object recognition systems. In 2018, Facebook AI developed Detectron, an object detection library [15]. Although it was a successful library, its usage was somewhat challenging. It was supported by Caffe2, but the presence of multiple codes integrating Caffe2 and PyTorch made the use of Detectron more complicated. In response to this situation, the Facebook AI Research group developed a new version. According to their own definition, Detectron2 is a next-generation software system that implements state-of-the-art object detection algorithms by Facebook AI Research. This version was developed using PyTorch, enabling it to compete more effectively with TensorFlow, a foundational library for artificial intelligence algorithms. With its straightforward installation instructions, this system provides an easy-to-use API for generating scoring results.

The purpose of Detectron2 is to provide a high-quality, high-performance codebase for object detection researchers. It is designed to be flexible, facilitating the rapid implementation and evaluation of new research. Detectron2 supports various advanced object detection models, including: Mask R-CNN, RetinaNet, Faster R-CNN, Region Proposal Network (RPN), Fast R-CNN, Region-based Fully Convolutional Networks (R-FCN). While performing these tasks, Detectron2 utilizes the following backbone architectures:, ResNeXt 50, 101, 152; ResNet 50, 101, 152; Feature Pyramid Networks

(FPN) integrated with ResNet and ResNeXt. Additionally, it incorporates VGG16 applications for object detection algorithms [15].

We integrate Detectron2 into our system. The example demonstrates the application of the system to a new dataset using an existing model. We first load our dataset into the system. Subsequently, we define the file path allocated for training. To assess whether the system accurately detects the introduced images, we randomly open the labeled images along with their annotations. Figure 3 presents an example from our own system. In the next step, the dataset is introduced to a pre-trained R50-FPN Mask R-CNN model by fine-tuning it, and the training process is initiated. During the fine-tuning phase, instead of starting training blindly, we utilize the weights of backbones previously trained on ImageNet images. These weights are derived from extensive training sessions conducted on high-capacity computers with prolonged computational times.

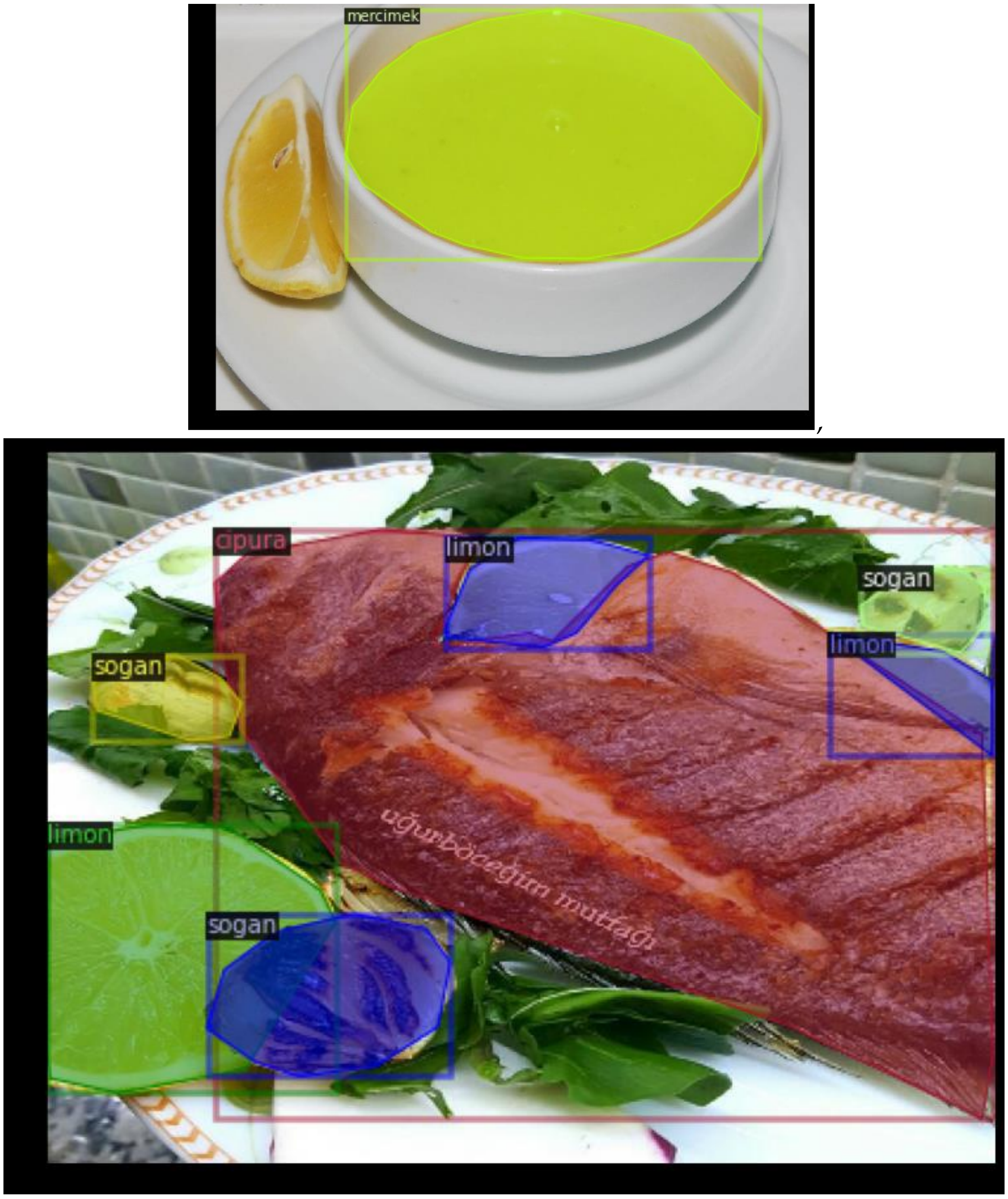


Figure 2: The Test Image of The System While The Training Set is Being Uploaded to The System.

2.2.ResNet50

In this study, the open-source CNN-based ResNet-50 architecture was used as the system backbone. ResNet architecture achieved first place in the 2015 competitions and is highly adaptable for use with various datasets. The ResNet-50 architecture consists of 50 layers

organized into five convolutional blocks. These blocks are composed of 1x1, 3x3, and 1x1 convolutional layers. The 1x1 convolutions reduce the dimensions of input images, while the 3x3 convolutions perform filtering at larger scales. A pooling layer is employed to further reduce dimensions. The fully connected layer of the architecture utilizes the Softmax activation function, providing an output of 1,000 categories for image classification.
Figure 4 illustrates the layers of the ResNet architecture.

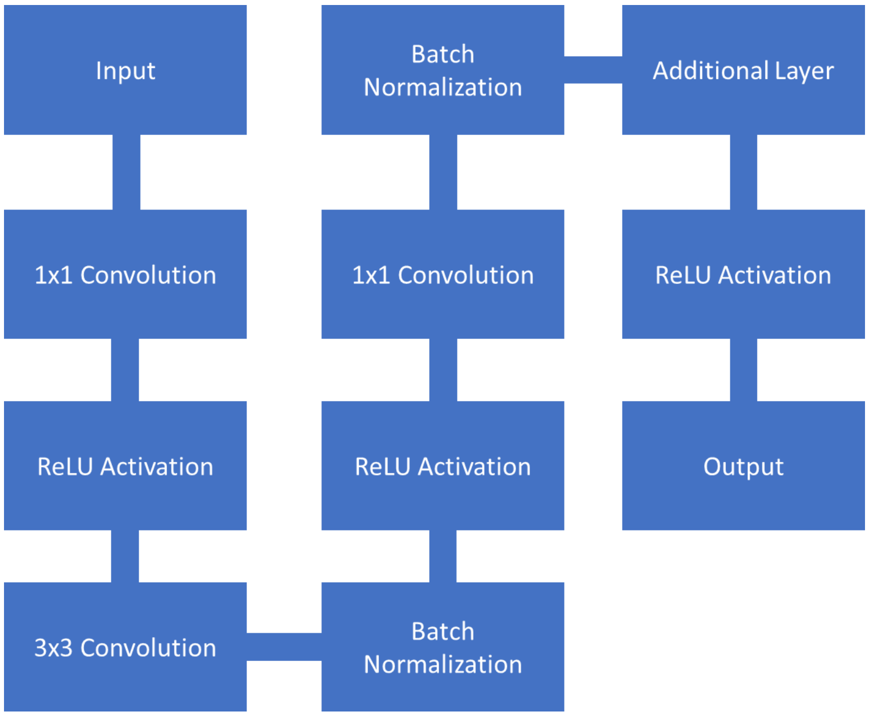


Figure 4: ResNet Architecture.

Before delving into shared layer behavior, the first step in ResNet involves a block consisting of convolution + batch normalization + max pooling operations. This block is referred to as Convolution 1.

Convolution itself is an integral operation that demonstrates how the shape of one function is modified by another function. The formula for convolution is given below:

$$(f * g)(t) = \int_{-\infty}^{\infty} f(\tau) g(t - \tau)\, d\tau \qquad (1)$$

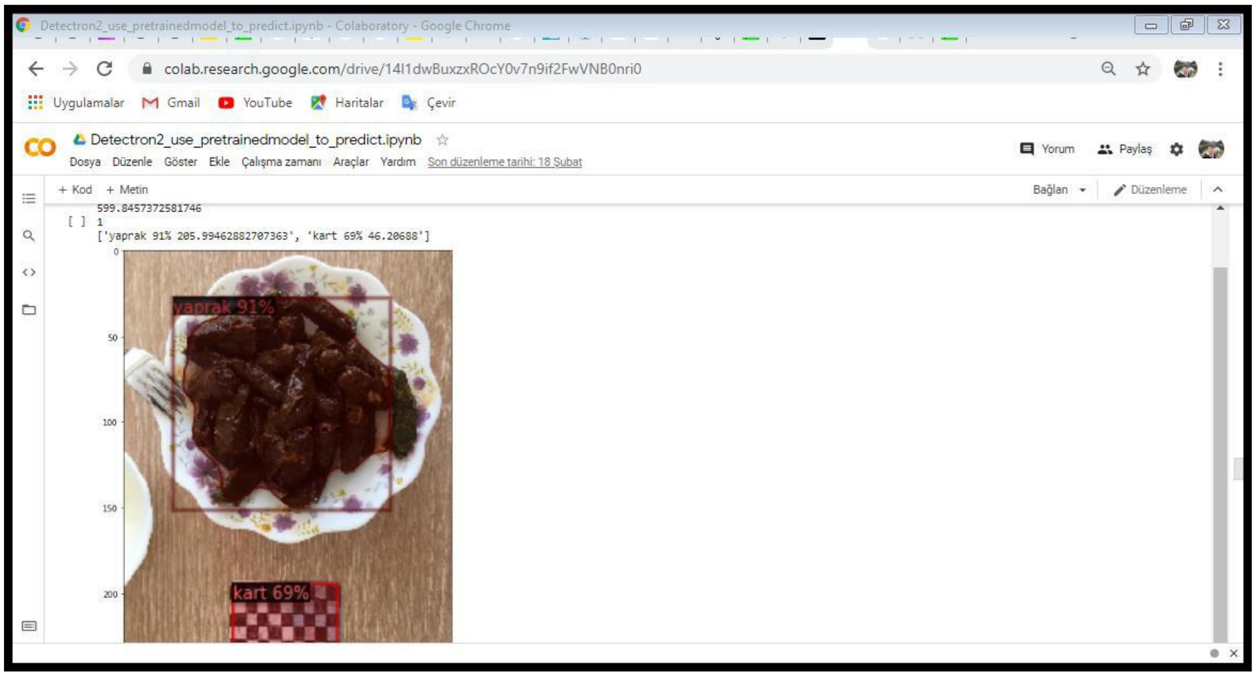
f(τ): Represents the input function (e.g., an image or signal).

g(t−τ) Represents the kernel or filter applied to the input.

t: The point at which the convolution is evaluated.

This operation allows ResNet to extract features from input data by applying specific filters to detect patterns such as edges, textures, or other significant elements.

During both the training phase and all subsequent testing stages, we utilized Google Colab. This approach saved us significant time thanks to the GPU support. Once the training was completed, we developed our final program to run on a CPU. Figure 5 displays the screenshot of the screen where we conducted post-training activities and tested the model.


Detectron2_use_pretrainedmodel_to_predict.ipynb - Colaboratory - Google Chrome
colab.research.google.com/drive/14l1dwBuxzxROcY0v7n9if2FwVNB0nri0
Uygulamalar
Gmail
YouTube
Haritalar
Çevir
Detectron2_use_pretrainedmodel_to_predict.ipynb
Dosya Düzenle Göster Ekle Çalışma zamanı Araçlar Yardım Son düzenleme tarihi: 18 Şubat
Yorum
Paylaş
+ Kod + Metin
Bağlan
Düzenleme
yaprak 91%
kart 69%
0
50
100
150
200

```
599.8457372581746
1
['yaprak 91% 205.99462882707363', 'kart 69% 46.20688']
```

Figure 5: The Test Image Conducted After 15 Training Sessions

2.3. Volume Estimation

We saved the trained models to our local computer, completing our tasks on the browser. All subsequent processes were carried out using the Spyder editor on our local machine.

There are various methods available for volume estimation. Some of these methods include: Calculating volumes by creating a 3D visual representation, using images captured by RGB-D cameras that provide depth measurements. The method we used, which works seamlessly with visuals obtained from almost any camera. This approach involves first calculating the area using a reference object and then estimating the volume using a pre-defined table.

The most accurate method for volume estimation is using depth information provided by an RGB-D camera, but this requires specialized hardware. The next best approach involves estimating the volume from a 3D visual created using images captured from multiple angles, combined with certain algorithms. For this process, a reference object is also required.

There are applications available for food analysis using multiple images. However, in our system, we calculate the area of the food using a single top-down image and the measurements of a pre-defined reference object. After this step, we measured the depth, diameter, and volume information of various soup bowls and plates of different sizes, as detailed in Table 2 below.

Table 2: Measured Plate Dimensions

Top Diameter (cm)	Bottom Diameter (cm)	Depth (cm)
21	12	3.5
16	7	5
20	11	4.5
20	12	3.8
16	7.5	5.5
21	10	4
20	11.5	4.8
12	7.5	5.2
20	10	4.5
20	12	3.8
19	10	3.5
11	6.5	3.5
20	12	4

The formula for calculating the volume V of a frustum-shaped container is as follows:

$$r_1 = \text{Top Radius}, \quad r_2 = \text{Bottom Radius}, \quad h = 2 \cdot |r_1 - r_2|$$

$$V = \frac{\pi \cdot h}{6} \cdot \left(3 \cdot r_1^2 + 3 \cdot r_2^2 + h^2\right)$$

(2)

Using the equation in (2), we calculated the volumes of the plates; however, these calculations did not fully align with our actual measurements. The discrepancy was due to the varying tapering angles and thicknesses of the plates. To minimize the error rate,

we aimed to adjust the difference between our measurements and calculations by multiplying them with specific coefficients.
In the equation, the term "cap" represents the area predicted by our model, which is treated as the area of a circle. From the formula for the area of a circle, we derived the diameter information. We also added an **if-else loop** to our code to account for variations in the range of the "h" value. For lentil soup, a different formula was employed. In this formula, we considered the radius derived from the image as the radius of a hemisphere. Using another **if-else loop**, we calculated a smaller base radius, subtracted the corresponding hemisphere volume, and obtained a bowl-like shape for volume estimation.
Additionally, we measured the actual volumes of the food defined in our system and the corresponding weights of these volumes. To do this, we filled 200 ml water glasses with the food, weighed them using a pre-tared precision scale, and recorded the weight corresponding to 200 ml as shown in Figure 6. Some of the dishes were purchased ready-made, while others were prepared by us. For homemade dishes, we verified their caloric information using various online resources, and for packaged foods, we used the nutritional values provided on the packaging. This data was compiled into a table.
Table 3 includes weight-to-volume ratios (g/ml) for soups and liquid dishes, while for desserts like baklava and trileçe, the calculations are based on g/cm². Our software calculates the diameter for plated dishes and the top surface area for desserts using the reference object. It also calculates the volume of plated dishes.
For dishes like rice, sautéed meat, sautéed chicken, or stuffed grape leaves, which can be heaped onto the plate, it was impossible for our model to determine from the top-down view whether the food was level or heaped. Therefore, during software operation, users are prompted to specify whether the food is level or heaped. Based on their selection, the software adjusts the calculation, considering the heaped volume to be 1.3 times the level volume.

Table 3: Food Measurement Table

Food	Gr (g)	Ml (ml)	Carbohydrates (100g)	Carbohydrates (100ml)	Gr/ml	100g Calories	Surface Area (cm²)	Gr/Surface Area
Lentils	227	200	8.28	9.40	1.135	55.9	0	0
Kidney Beans	226	200	24.46	27.64	1.13	257	0	0
Gilthead Bream	200	200	0.00	0.00	1.00	96	0	0
Baklava	41	21	37.30	72.82	1.95	285	25	1.64
Beans	205	200	7.60	7.79	1.025	85	0	0

Food	Gr (g)	Ml (ml)	Carbohydrates (100g)	Carbohydrates (100ml)	Gr/ml	100g Calories	Surface Area (cm²)	Gr/Surface Area
Meat Stew	206	200	3.15	3.24	1.03	107	0	0
Rice Pilaf	153	200	18.60	14.23	0.765	118	0	0
Stuffed Eggplant	390	300	5.33	6.93	1.3	52.84	0	0
Dumplings	10	20	29.71	14.85	0.5	170	0	0
Menemen	106	114	3.33	3.15	0.93	71	0	0
Chickpeas	376	385	20.59	20.11	0.98	164	0	0
Rice	153	200	28.59	21.87	0.765	130	0	0
Chicken Stew	159	200	4.51	3.59	0.795	169	0	0
Tres Leches	200	243	21.05	17.33	0.823	163	64	3.13
Stuffed Leaves	411	360	21.36	24.39	1.14	165.53	0	0
Dry Beans	376	385	25.09	24.50	0.98	139	0	0

Figure 6: The Weight of 200 ml of Foods

As previously mentioned, liquid dishes provide the most accurate results, while the calculation method for desserts is different. For desserts, there is a direct relationship between the surface area and weight as shown in Figure 7. Since the average thickness of a piece of baklava is known, we can directly associate its surface area with its weight.

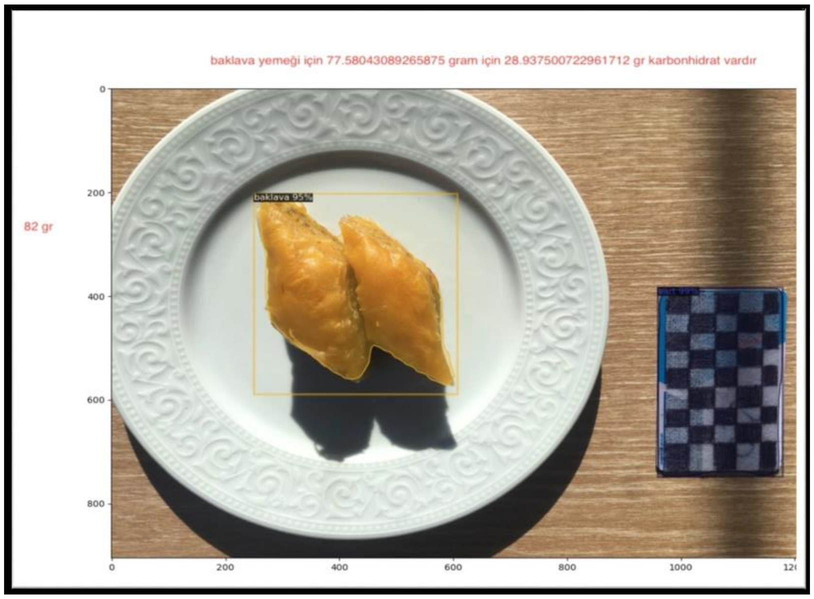
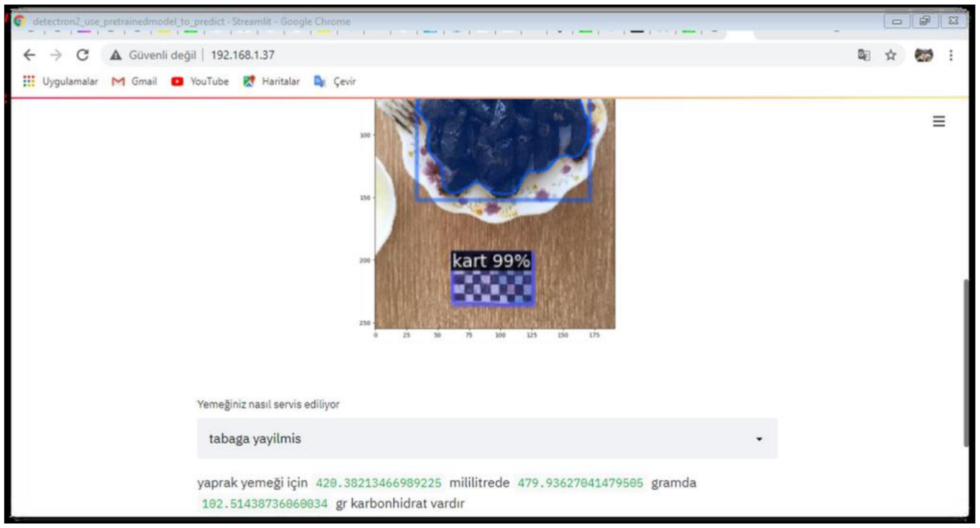

baklava yemeği için 77.58043089265875 gram için 28.937500722961712 gr karbonhidrat vardır
82 gr
baklava 95%
0
200
400
600
800
1000

Figure 7: The predicted weight of the baklava, which was measured at 82 grams.

2.4.Creating a Web Application

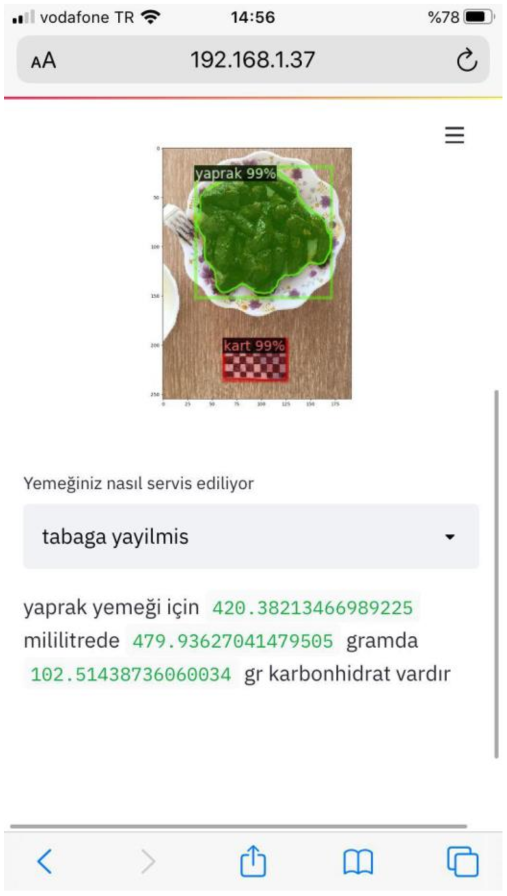
Streamlit is a library that allows us to use all the system codes we have prepared without modification, requiring only minor user interface adjustments. It provides the infrastructure for running our system through the web.
In the web application, the system can be run either by entering the image URL through the web or by uploading an image from the local computer. The system operates on the CPU.


detectron2_use_pretrainedmodel_to_predict · Streamlit - Google Chrome
Güvenli değil | 192.168.1.37
Uygulamalar
Gmail
YouTube
Haritalar
Çevir
kart 99%

Yemeğiniz nasıl servis ediliyor

tabaga yayilmis

yaprak yemeği için `420.38213466989225` mililitrede `479.93627041479505` gramda `102.51438736060034` gr karbonhidrat vardır



Yemeğiniz nasıl servis ediliyor

tabaga yayilmis

yaprak yemeği için `420.38213466989225` mililitrede `479.93627041479505` gramda `102.51438736060034` gr karbonhidrat vardır

Figure 8: Screenshots of Web Application

3. Results

The necessary codes have been written for the interface we built to allow users to upload images. Users can upload their images to the system using the previously mentioned different methods. Once an image is uploaded, the system restarts, processes the uploaded image as input, and attempts to recognize the food in the image along with the reference object. The system also tries to mask the area of the food. Using the identified area, along with the method described in the methodology section, the system performs the necessary calculations and provides the user with the estimated carbohydrate content of the food.

Before uploading the images, the foods were weighed using a precision kitchen scale, and their weights were recorded. The results obtained from our system, along with the actual measurements and error rates, are provided in Table 4. Since our system calculates the volume from a single image, the results indicate high accuracy.

In more advanced systems, the volume is calculated by generating a 3D model of the food using images taken from multiple angles, which leads to more accurate results. However, requiring a user to upload images of a dish from multiple angles may not be a practical or sustainable approach. In this respect, our system achieves high accuracy rates for commonly used plated dishes.

Table 4: *Actual Values, Predicted Values, and Error Percentages*

Dish Name	Actual Volume	Predicted Volume	Volume Error (%)	Actual Weight	Predicted Weight	Weight Error (%)
Bulgur Pilaf	342	363.89	6.40	-	-	-
Sautéed Meat	200	206.66	3.33	-	-	-
Baklava				41	46.90	14.39
Menemen				114	102.47	10.11
Chickpeas				311	310.95	0.016
Trilece				400	428.59	7.15
Lentil Soup	220	239.24	8.75	-	-	-
Stuffed Grape Leaves				274	316.88	15.65
Green Beans				122	128.04	4.95
Chicken Sauté				276	236.77	14.21

We observe the average errors resulting from measurements across five different classes. From these results, it is evident that foods with solid structures that do not conform to the shape of the plate have higher error rates compared to liquid dishes. However, in all cases, the average error levels remain below 10%. The difference between chickpeas and lentil soup stems from the fact that one is served in a bowl and the other on a plate. Bowls tend to have more diverse alternatives and depths compared to plates. Our system performs a generalized calculation to account for these variations. The system estimates the diameter in the image and calculates the depth (in cm) of the soup corresponding to that diameter using its formula. This formula has been adjusted to provide an average value based on previous measurements. As shown in Figure 9, the error range is narrower for liquid dishes, while it is relatively larger for dishes with more solid structures. In our system, the dish most prone to error is stuffed grape leaves. Since the presentation of stuffed grape leaves can take many different forms, achieving the most accurate result with this method is not possible.

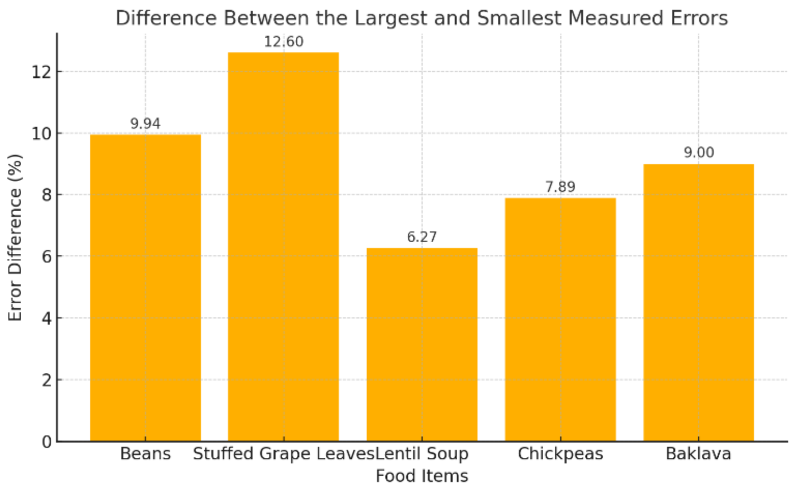


Figure 9: Difference Between the Largest and Smallest Measured Errors

4. Discussion and Conclusion

It is critically important for individuals with T1DM (Type 1 Diabetes Mellitus) to perform carbohydrate counting quickly, accurately, and consistently. Typically, this task is accomplished either by entering portion information online to calculate carbohydrate content or by weighing and measuring ingredients while preparing meals themselves. However, this process can be exhausting and time-consuming for individuals. With the use of computer vision techniques, the goal is to minimize the amount of manual calculation T1DM individuals need to do regarding their meals and enable them to spend higher-quality time. A significant number of studies have been conducted in this area.
Unlike methods used in previous studies, our system has been developed using Detectron2. The speed of Detectron2 has been proven in various studies, making it a

suitable choice for this field of research. On the other hand, since the dataset we used was created from images we collected online, we do not have a large amount of training data. Despite this, our findings show that the system recognizes and masks foods with high accuracy. Additionally, since the images in our dataset were not specifically taken for our system, they were often captured at inclined angles. Nevertheless, our model performs successfully with top-down images, which is a testament to its effectiveness.

As highlighted in the findings of our system, while it delivers highly accurate results for liquid dishes, it may not perform as well for solid dishes. To address this, different methods could be tested for analyzing solid dishes. For instance, in the case of stuffed grape leaves, the system could identify a single piece in the image and count the total number of pieces to achieve more accurate results. The system's performance with liquid dishes, however, is at a level suitable for easy use by end-users.

Occasionally, the system may encounter recognition issues for the reference object due to the limited number of training images for such objects. However, these issues can often be resolved by ensuring proper lighting conditions.

Our system is suitable for 3D food analysis processes using different angles in the future and can be developed in that direction. Since the system we built can be easily used with any phone that has internet access, it allows users worldwide to perform analysis through a server without needing to rely on their device's GPU or CPU. Additionally, if desired, the system can be easily converted into a mobile application compatible with Android or iOS operating systems.

Currently, the application has 17 defined classes, out of which 12 classes function properly, based on the training data and iterations conducted during the training process. If necessary improvements are made to the system, the number of classes can be increased, and the quality of the training data can be enhanced. This would eliminate any barriers to the system's usability for end-users.

Our system works seamlessly with the currently defined classes, and it can be used effortlessly by anyone. It also has the potential to be further developed for more advanced 3D food analysis using images from different angles, making it versatile and future-proof.

References

[1] Nilson, N. J. (1998). Introduction to machine learning. Stanford University, USA, 89 s.

[2] IBM Cloud Education. (n.d.). What is machine learning? Retrieved April 2021, from https://www.ibm.com/cloud/learn/machine-learning#toc-deep-learn-nOh7s5Rf

[3] McCulloch, W. S., & Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. The Bulletin of Mathematical Biophysics, 5(1), 115–133.

[4] Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. Psychological Review, 65(6), 386.

[5] Kelley, H. J. (1960). Gradient theory of optimal flight paths. ARS Journal, 30(10), 947–954.

[6] Dreyfus, S. (1962). The numerical solution of variational problems. Journal of Mathematical Analysis and Applications, 5(1), 30–45.

[7] Dehais, J., Anthimopoulos, M., Rhyner, D., & Loher, H. (2016). Carbohydrate estimation by a mobile phone-based system versus self-estimations of individuals with type 1 diabetes mellitus: A comparative study. Journal of Medical Internet Research, 18(5), e101. https://doi.org/10.2196/jmir.5567

[8] Alfonsi, J. E., Choi, E. E. Y., & Arshad, T. (2020). Carbohydrate counting app using image recognition for youth with type 1 diabetes: Pilot randomized control trial. JMIR mHealth and uHealth, 8(10), e22074. https://doi.org/10.2196/22074

[9] Anthimopoulos, M., Dehais, J., & Diem, P. (2015). Computer vision-based carbohydrate estimation for type 1 patients with diabetes using smartphones. Journal of Diabetes Science and Technology, 9(4), 759-764. https://doi.org/10.1177/1932296815580159

[10] Vasiloglou, M. F., Mougiakakou, S., & Aubry, E. (2018). A comparative study on carbohydrate estimation: GoCARB vs. dietitians. Nutrients, 10(6), 741. https://doi.org/10.3390/nu10060741

[11] Contreras, I., Muñoz-Organero, M., Beneyto, A., & Vehi, J. (2023). Active labeling correction of mealtimes and the appearance of types of carbohydrates in Type 1 Diabetes information records. Mathematics, 11(19), 4050. https://doi.org/10.3390/math11194050

[12] Ladyzynski, P., Krzymien, J., & Foltynski, P. (2018). Accuracy of automatic carbohydrate, protein, fat and calorie counting based on voice descriptions of meals in people with type 1 diabetes. Nutrients, 10(4), 518. https://doi.org/10.3390/nu10040518

[13] Schmidt, S., Schelde, B., & Nørgaard, K. (2014). Effects of advanced carbohydrate counting in patients with type 1 diabetes: A systematic review. Diabetic Medicine, 31(8), 886-898. https://doi.org/10.1111/dme.12446

[14] Bell, K. J., Barclay, A. W., Petocz, P., Colagiuri, S., & Brand-Miller, J. C. (2014). Efficacy of carbohydrate counting in type 1 diabetes: A systematic review and meta-analysis. The Lancet Diabetes & Endocrinology, 2(2), 133-140. https://doi.org/10.1016/S2213-8587(13)70144-X

[15] R.G.I.R.G.G.P.D.Detectron. GitHub.https://github.com/facebookresearch/Detectron), Release Date: April 2021.