AKADEMIA GÓRNICZO-HUTNICZA IM. STANISŁAWA STASZICA W KRAKOWIE

Neural networks and deep learning

Food classification

Authors: Krzysztof Madej, Mateusz Mglej, Łukasz Obrzut

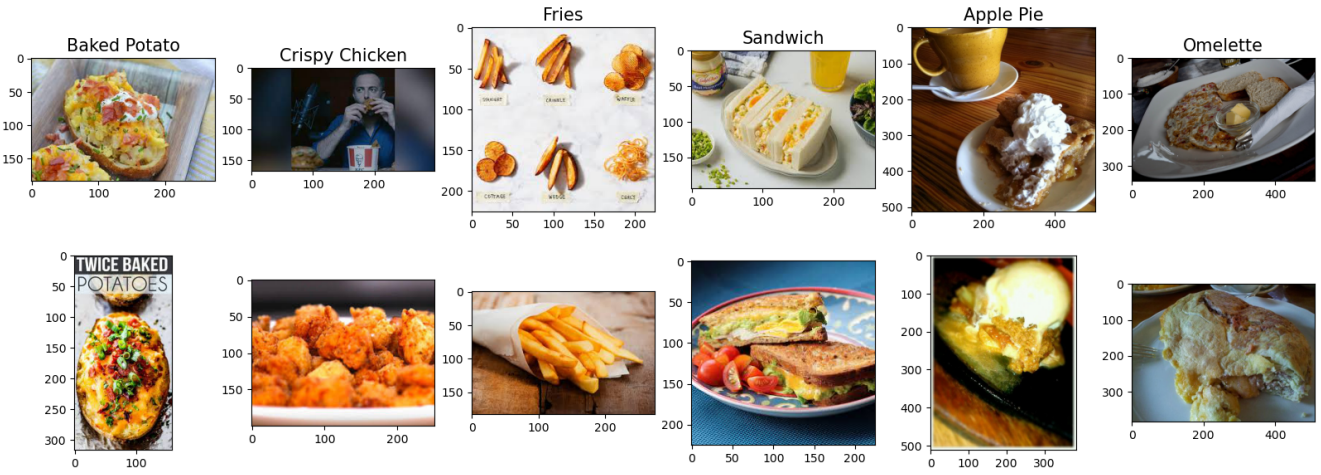
Kraków, 15.06.2024

1 Dataset and Problem Statement

The dataset Food Classifcation dataset contains 24,000 images obtained from various Google resources. It is divided into 35 varieties of Indian and Western appetizers. The goal of the project is to create a model with the highest possible prediction accuracy based on the images.

In our project, we choose only 6 out of the possible classes

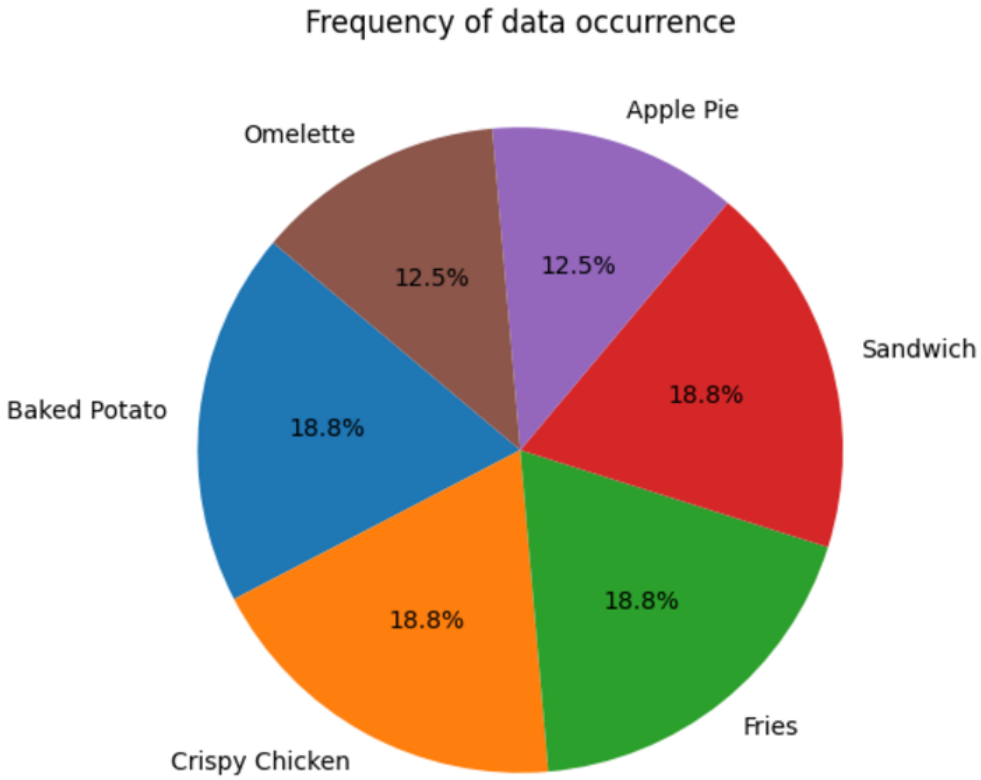
- Baked Potato
- Crispy Chicken
- Fries
- Sandwich
- Apple pie
- Omelette



Rysunek 1: Example images from the dataset

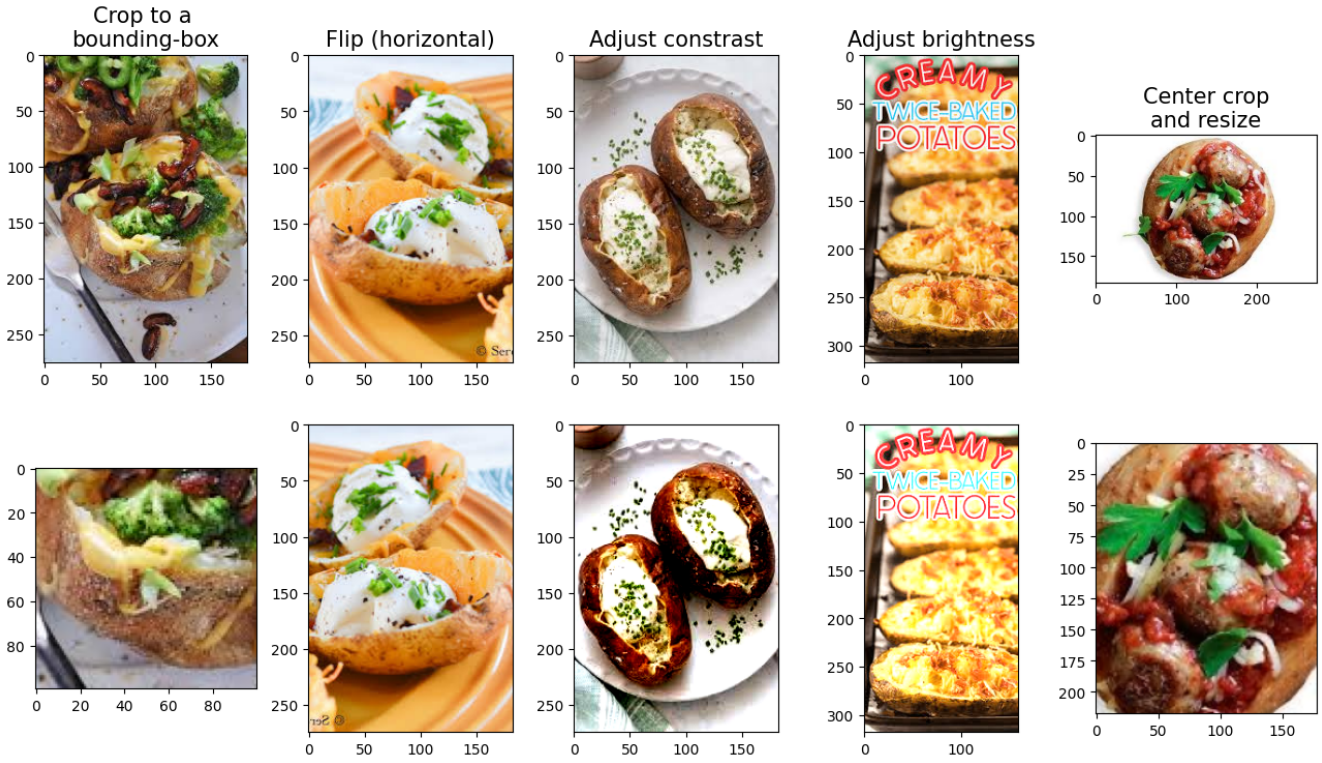
2 Data Exploration and Preparation

We load the data and create a DataFrame using the pandas library. Next, we filter the data to include only the classes of interest. Below, we present the class distribution in the training data.

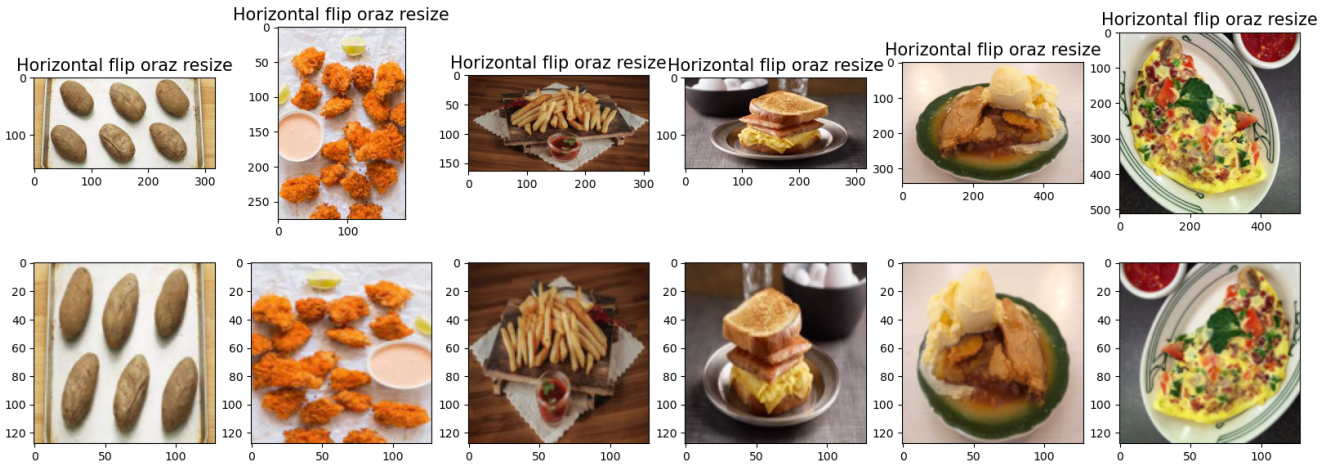


We convert the current text labels to numerical labels. Next, we split our data into training, validation, and test sets using the train_test_split function (from the scikit-learn library) twice.

Below, we present the image transformations we are considering.



In our model, we decided to use only horizontal flip and resize.



3 Model Building

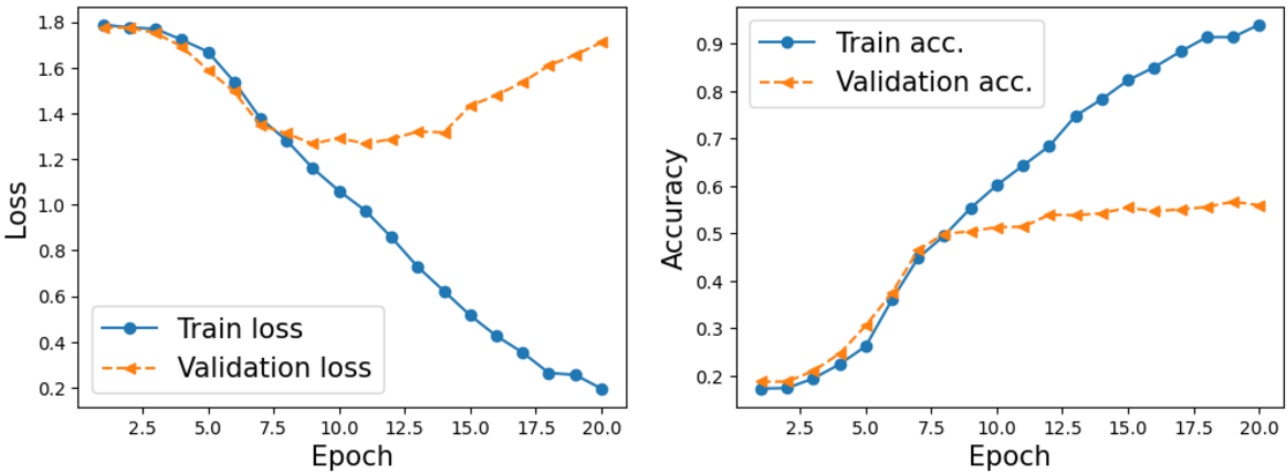
Below, we present the construction of our model.

Nr	Layer names	Layer type	Parameters
1	conv1	Conv2d	in_channels=3, out_channels=32, kernel_size=5, padding=2
2	relu1	ReLU	-
3	pool1	MaxPool2d	kernel_size=2
4	conv2	Conv2d	in_channels=32, out_channels=64, kernel_size=5, padding=2
5	relu2	ReLU	-
6	pool2	MaxPool2d	kernel_size=2
7	flatten	Flatten	-
8	fc1	Linear	in_features=16384, out_features=1024
9	relu3	ReLU	-
10	dropout	Dropout	p=0.5
11	fc3	Linear	in_features=1024, out_features=6

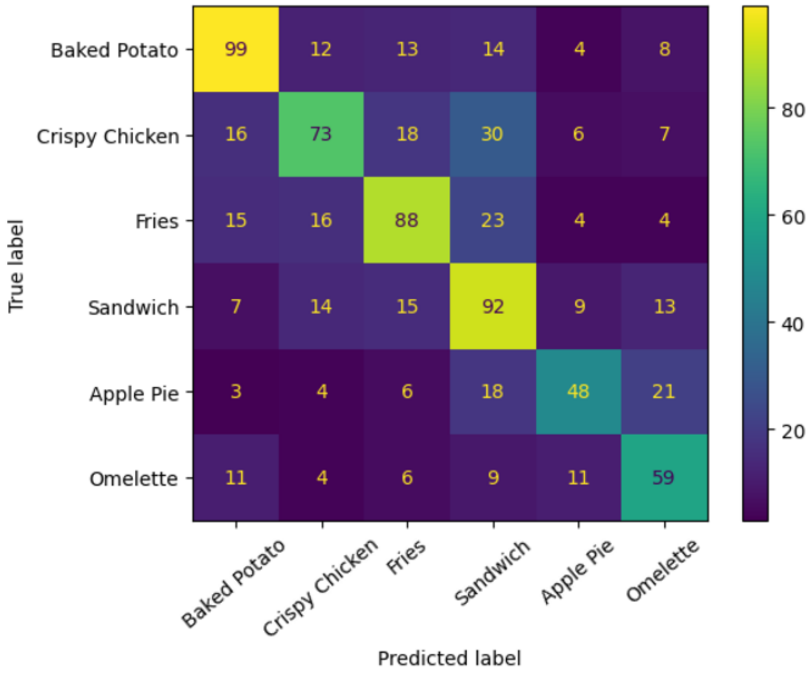
- Loss function: torch.nn.CrossEntropyLoss()
- Optimizer: torch.optim.Adam(model.parameters(), lr=0.001)
- The convolutional layer is responsible for extracting local features such as edges, textures, and patterns using filters (kernels).
- ReLU introduces non-linearity to the model through the formula: $f(x) = \max(0, x)$.
- MaxPool reduces the data size by selecting the maximum value from non-overlapping windows, helping to extract the most important features and reducing the dimensionality.
- Flatten transforms a multi-dimensional data matrix into a one-dimensional vector, preparing the data for fully connected layers.
- Linear computes a linear combination of input features with weights, often adding a bias term, allowing the mapping of input data to the output space.
- Dropout randomly deactivates a certain percentage of neurons in the layer during training, which helps prevent overfitting and improves the generalization of the model.

4 Model Evaluation

On the test set, we obtained an accuracy of **test accuracy: 0.5738**. Below, we present the plots of the loss function and accuracy of the model.



Below is the **confusion matrix** showing how well the model performs with respect to specific classes.

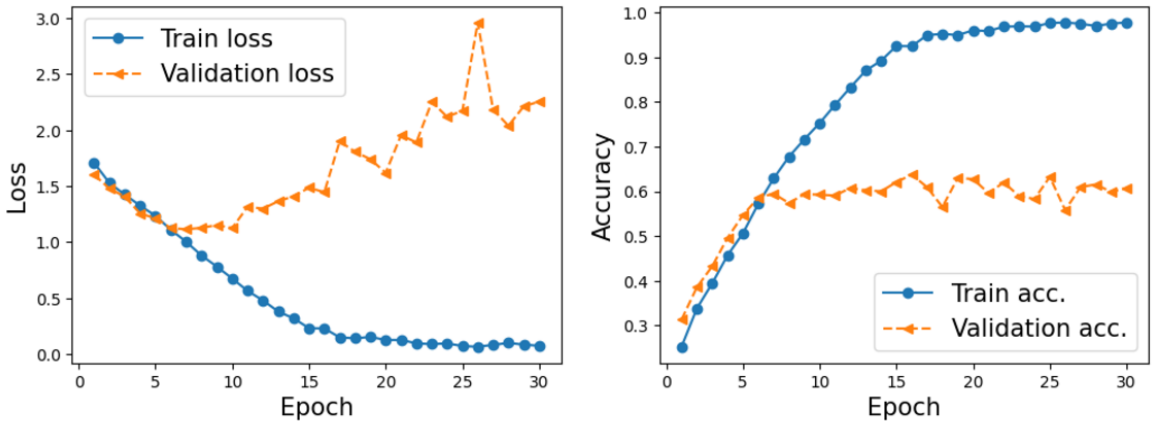


5 Model Refinement

5.1 Model 2

Base model + increased resize from 64→128 and added an additional convolutional layer

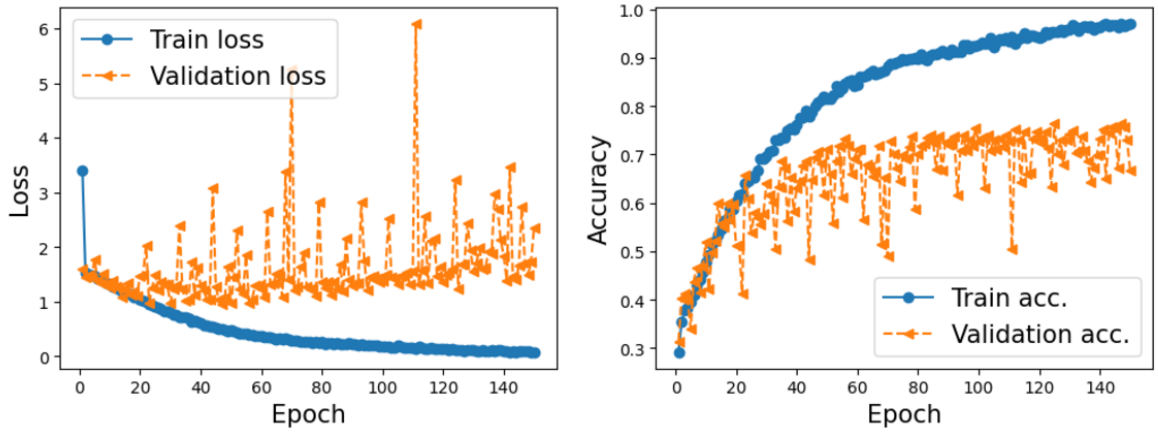
Test accuracy: 0.6237



5.2 Model 3

Model 2 + adding BatchNorm2d after each convolutional layer Batch normalization in convolutional networks works by standardizing (normalizing) the activations in each layer, which improves stability and learning speed by scaling and shifting these activations during the training of the network.

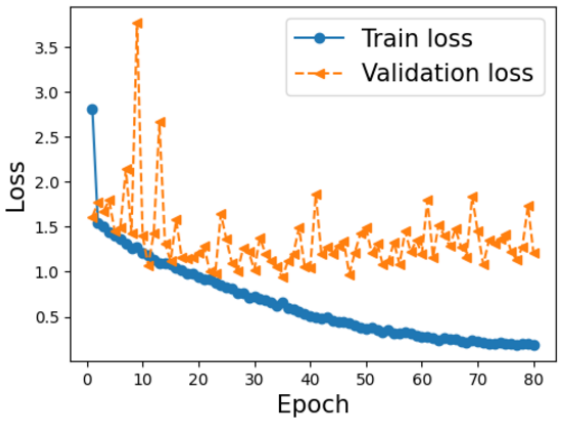
Test accuracy: 0.6812

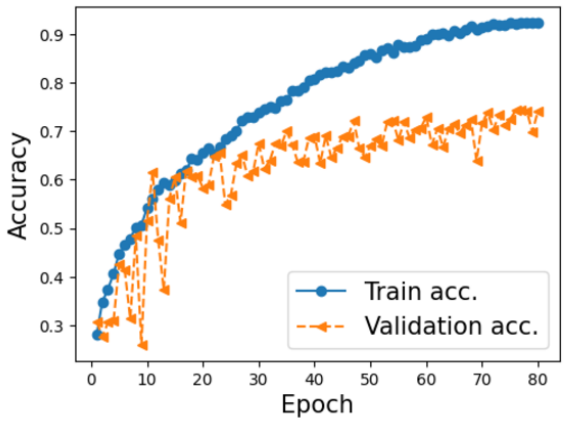


5.3 Model 4

Model 3 + twice the number of channels in convolutional layers, change the resize parameter to 64

Test accuracy: 0.7362



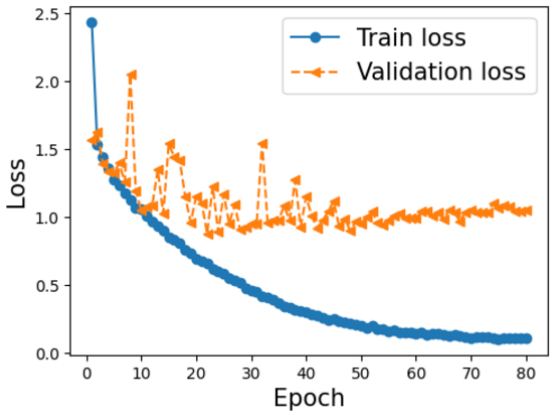


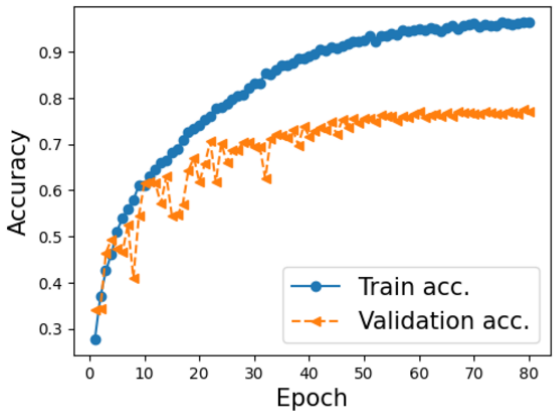
5.4 Model 5

Model 4 + variable learning rate (lr)

In each epoch, we reduce the learning rate by 5%: $lr = 0.95 \cdot lr$.

Test accuracy: 0.7563





6 Coclusions

Model 2 performs similarly to the base model.

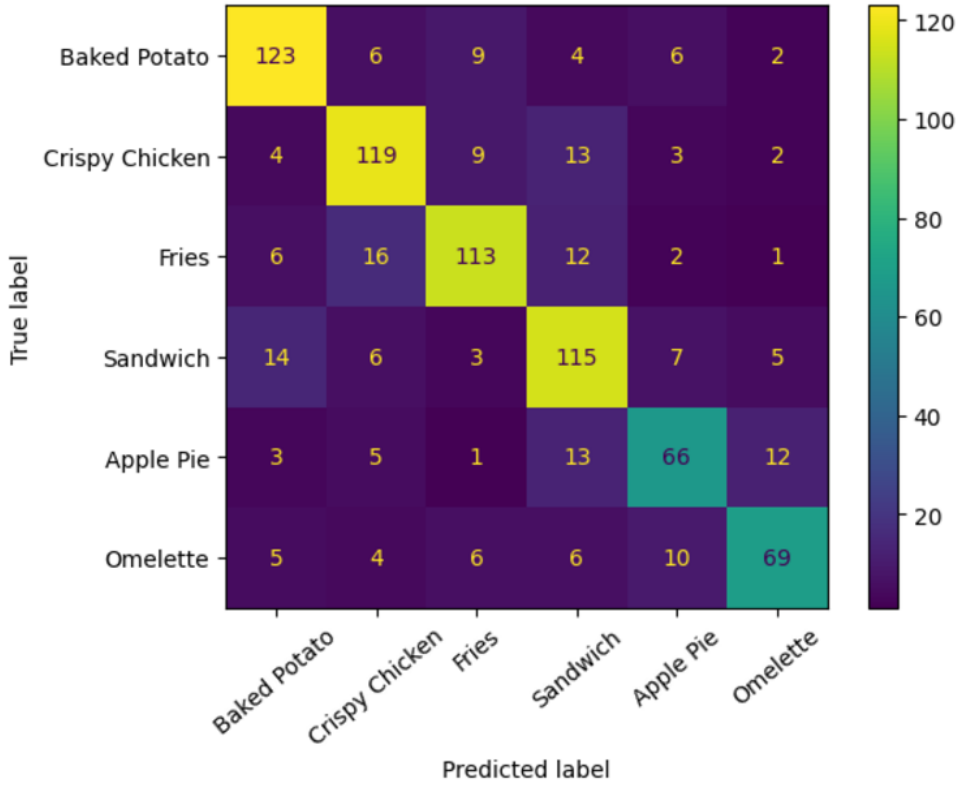
Model 3 gives us much better results, but the loss function and accuracy are unstable.

Model 4 has similar results to Model 3, but with slightly better stability.

Model 5 has the best results and model stability.

We choose Model 5 as our final model. Test accuracy: 0.7563

Confusion Matrix



Incorrectly classified examples: