Comparison of Two Models on Food Recognition

Ernesto Pedrazzini
Artificial Intelligence for Science and Technology
Università degli Studi di Milano Bicocca
Email:e.pedrazzini4@campus.unimib.it

Viola Cavedoni
Artificial Intelligence for Science and Technology
Università degli Studi di Milano Bicocca
Email:v.cavedoni@campus.unimib.it

***Abstract*—The aim of this paper is to analyze and compare the performance of a CNN and a SVM coupled with SIFT+BoW feature extractor on a classification task, in particular food recognition on the dataset provided for the iFood challenge of 2019.**

1. Introduction

Classification tasks are widely present in several contexts: in this case we had to perform a food recognition task on images, which naturally implies the presence of many classes. The development of this task requires a feature extraction part and a classifier: for this reason, Convolutional Neural Networks are a powerful instrument. In our work, we implemented both a CNN and a Scale Invariant Feature Transform/Bag of Words feature extractor coupled with a Support Vector Machine classifier, and compared their performances.
Our analysis begins with the dataset description provided in the first section, while in the second section we show how pre-processing was carried out on the images. Then in the fourth section we describe our CNN, displaying the more significant results and analyzing the problems affecting its performance. The same analysis are repeated for SIFT/BoW + SVM in the fifth section, and finally a comparison is done in the sixth section.

2. Dataset description

The dataset provided is the one for the iFood challenge of 2019, containing 118475 training images from 251 classes and 11994 validation images from the same 251 classes, that we will use as test set.
We can notice the presence of noise both in the train and in the test set, that is the presence of images in which the food is badly visible or even not present in the image. Some examples follow.

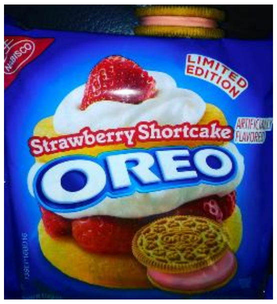

Figure 1. A test image of the class ”strawberry shortcake” in which the food is not clearly visible.

Figure 2. Here the image is a butterfly similar to the food in the class ”farfalle” (by Barilla).

3. Pre-processing

The validation set has been extracted as the 20% of the training set, then images have been organized in 251 different folders depending on the relative class in order to load the dataset in an optimal way to the Kaggle environment.
Since four of the classes presented evidently less images than the majority of others, we thought to augment those categories both in the training and validation set. In particular, the classes ”steak tartare”, ”hamburger” and ”veal cordon bleu” present less than 300 samples, while the class ”marble cake” presents even less than 100 samples.
So for the augmentation of the first three classes cited above, we employed a rotation of a random angle in $(0, 180)$ degrees, a random flipping and a decreasing brightness transformation of a random intensity.
For the last class cited above, we employed two rotations with random angles in $(0, 90)$ and $(90, 180)$ degrees

respectively, an horizontal, vertical and mixed flipping, an increasing and decreasing brightness transformations of a random intensity.

(a) Train image of class ”marble cake” without transformation.

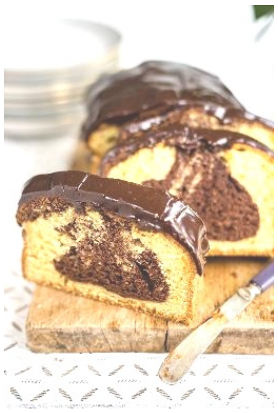
(b) Train image of class ”marble cake” with increased brightness.

Figure 3.

From the histogram shown below, we can see that after the augmentation there are no unbalanced classes.

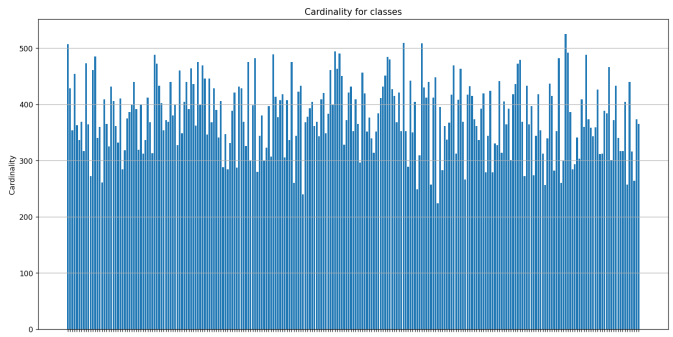

Figure 4. Histogram of the number of samples per class in the training set after the augmentation.

4. First model: Convolutional Neural Newtork

Convolutional Neural Networks are the most used architectures in computer vision: dealing with images, they exploit local connectivity to extract the information of a region by convolution with various kernels. Convolution is a local and translational invariant operation, which uses the same weights for all the visted regions.
Then a stacked feature representation is performed providing a depth to the layer structure. In this way local patterns can be processed in combination.
The various layers are organized to provide a hierarchy of feature representations.

4.1. Description of our model

Our network is composed by four convolutional layers, four pooling layers (one between each couple of convolutional layers), and two fully connected layers.
The kernel size progressively goes from 5 to 2 since the information is compressed step by step and finer features are computed. On the other side the layer depth increases in order to collect more patterns to be combined. This also helps providing more depth to the CNN, but using few parameters.
Zero padding size is always 1 since we didn’t want too much parameters, while the stride is always set to 1 not to lose information.
Each pooling layer employs a max pooling with a kernel of size 3 and stride 2, and is useful to compress the size of the output.
The first fully connected layer receives a flatten output from the convolutional layers, and it’s composed by 550 neurons, while the output layer is composed of course by 251 neurons, according to the number of classes.
The activation function used is a leaky ReLu, which is similar to ReLu but returns a negative value (proportional to the input) for a negative input, allowing also negative inputs to contribute.
The structure obtained is represented in Figure 5: as you can see 953201 parameters are to be tuned.

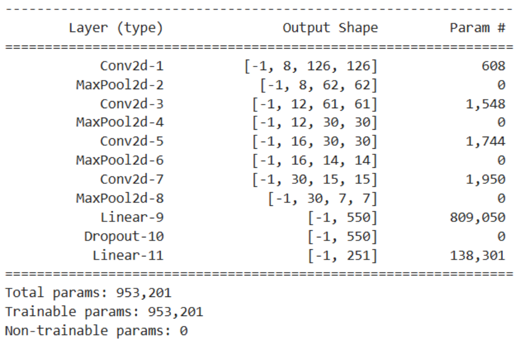
```
----------------------------------------------------------------
        Layer (type)               Output Shape         Param #
================================================================
            Conv2d-1          [-1, 8, 126, 126]             608
         MaxPool2d-2            [-1, 8, 62, 62]               0
            Conv2d-3           [-1, 12, 61, 61]           1,548
         MaxPool2d-4           [-1, 12, 30, 30]               0
            Conv2d-5           [-1, 16, 30, 30]           1,744
         MaxPool2d-6           [-1, 16, 14, 14]               0
            Conv2d-7           [-1, 30, 15, 15]           1,950
         MaxPool2d-8             [-1, 30, 7, 7]               0
            Linear-9                  [-1, 550]         809,050
          Dropout-10                  [-1, 550]               0
           Linear-11                  [-1, 251]         138,301
================================================================
Total params: 953,201
Trainable params: 953,201
Non-trainable params: 0
```

Figure 5. Overall scheme of the network parameters.

Moreover a summary of the various receptive fields is present in Figure 6, where we can see that an increasing depth corresponds to a higher value of the spatial region of input that influences a neuron.

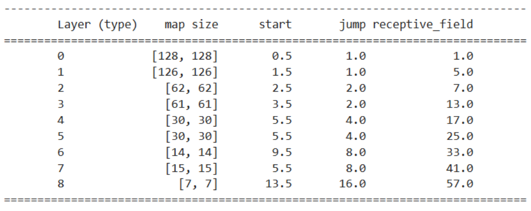
```
---------------------------------------------------------------------------
        Layer (type)    map size      start       jump receptive_field
===========================================================================
        0             [128, 128]        0.5        1.0             1.0
        1             [126, 126]        1.5        1.0             5.0
        2               [62, 62]        2.5        2.0             7.0
        3               [61, 61]        3.5        2.0            13.0
        4               [30, 30]        5.5        4.0            17.0
        5               [30, 30]        5.5        4.0            25.0
        6               [14, 14]        9.5        8.0            33.0
        7               [15, 15]        5.5        8.0            41.0
        8                 [7, 7]       13.5       16.0            57.0
===========================================================================
```

Figure 6. Receptive field of the network.

4.2. Training and validation performance evaluation

The training and validation were performed with a stochastic gradient descend optimizer (in which gradients are computed using only a random subset of the samples) with 0.01 initial learning rate (which controls the speed of the learning) and 0.09 momentum (which takes into account previous steps of the algorithm), while the cosine annealing scheduler is used to search for global optima in a wider space and for more precise optima in a smaller one after some iterations. For the loss we chose cross entropy loss, coherently with multi-class task.
Since we noticed that the validation loss got worst after few epochs, while the training loss improved, we understood that the model overfitted, so a solution we found was to add dropout of 50% in the first linear layer. This improved the validation loss and the best model was updated after the 14-th epoch of 15.

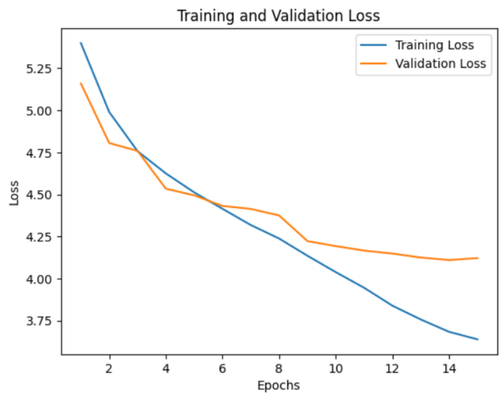

Figure 7. Training and validation loss after 15 epochs.

4.3. Test performance interpretation

The overall accuracy on the test set is 16%, with 229 classes associated to a non null percentage of correctly classified samples.
In particular, we wanted to check for misclassifications between similar classes, as "sashimi" and "sushi": the corresponding percentages of correctly classified samples are 17% and 11%, so we expect some errors.
From the following confusion matrix we can see that most of the sushi samples are classified as sashimi: in fact, sashimi is a particular kind of sushi, so we expect features to be similar both for color and shape.

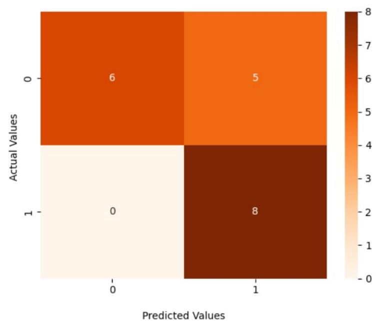

Figure 8. Confusion matrix for the classes "sushi" (on label 0) and "sashimi" (on label 1).

Let's now see an example of classes which are clearly different from an human point of view, for example "stuffed tomatoes" and "buffalo wings": the corresponding percentages of correctly classified samples are 46% and 54%, quite high with respect to the previous example.
In this case the color of the food may be similar, but looking at the confusion matrix we can see that only one sample of buffalo wings is classified as stuffed tomatoes, proving that our network is able to distinguish between samples from well separated classes.

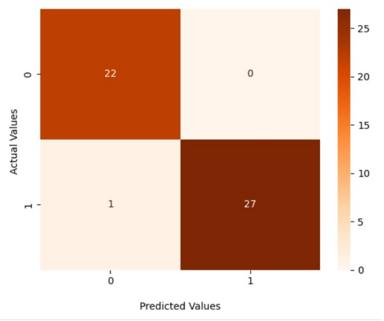

Figure 9. Confusion matrix for the classes "stuffed tomatoes" (on label 0) and "buffalo wings" (on label 1).

In conclusion, we can say that probably with more parameters (more types of kernels and more layers) our network would provide a better result. Also the presence of noise in the dataset may influence the performance.
Despite this, we think that CNN is a powerful and fast instrument to detect features, retaining also the color information (fundamental for food recognition).

5. Second model: SIFT and BoW coupled with a Support Vector Machine

5.1. SIFT algorithm

The general idea is to extract features which are important enough to be able to recognize a particular object in the picture.
Since we usually find objects in different positions, perspectives and scales, the feature extractor should be invariant to those transformations. In particular, SIFT is not influenced by scale, rotation, illumination and viewpoint.
Scale invariance is achieved with different blur levels obtained with gaussian filters, which are also useful to detect keypoints, that are the most representative points in a picture. In practice, only corners are taken into account using a sliding filter on the difference of gaussians.
To achieve more robustness, after keypoint detection a keypoint description is performed: considering gradient orientations in a neighbourhood of the keypoint, we assign the dominant one in order to reach also rotation invariance.

5.2. BoW algorithm

SIFT descriptors from training images are used to construct a ”visual vocabulary” by Bag of Words algorithm. In particular, k-means is employed to cluster the descriptors and find significant ”words”.
After this, for each image a representative histogram is computed by counting how many times a visual word occurs.
Now we are ready to use the histogram information to train a SVM model that separates images in different classes.

5.3. Description of our model

To use SIFT feature extractor, we converted images to gray scale, resized them to 128x128 and normalized using the mean and the standard deviation.
Since computational time in this case is very long, only a subset of 30% of images from each training class has been used.
For SIFT extractor we decided to keep the default parameters values because a change in those values didn't provide a noticeable improvement.
Next, the BoW vocabulary is constructed to contain 380 visual words.
Regarding the support vector machine, for multi-class tasks two approaches are possible: one-to-one and one-to-rest. In this case we used the first one, in which the problem is decomposed in binary classification tasks, each with an associated SVM. The final output is determined by majority voting.
We performed a grid search on 10% of the training data: the parameters involved are the type of kernel (either gaussian or polynomial) and the misclassification tolerance parameter C (ranging from 20 to 45). From this search, the optimal parameters found are C = 30 and gaussian kernel.
In addition some other parameters selection has been performed: we tried to change the number of words in the dictionary, or the image size. The first approach didn't show any significant result, while the second was able to almost duplicate the overall accuracy: passing from 128 to 224 pixels, we obtained $1,6\%$ and $2,7\%$ of accuracy respectively.

5.4. Performance evaluation

The overall test accuracy is of $2,7\%$, which is pretty much lower than the one found by our CNN.
Given the fact that SIFT works with gray scale images, it's clear that all the color information, which is of key importance for this task, is lost.
Also the fact that we trained the model with only a subset of the data influences the performance.
In addition, SIFT detects corner like features which are not suitable to fully represent various types of food (like smooth shape food, as yogurt and cream).
Probably increasing the size of the images would help finding more relevant features, thus improving the performance, but for computational reasons this is not possible.

6. Comparison

Convolutional Neural Networks are a powerful tool for classification tasks, in particular if used with image datasets. CNNs are faster and better performing than SIFT since the feature extraction and classification are not performed separately, allowing more complex features processing. Indeed with deep neural networks we can combine together more information and get better results. In addition, this kind of structures better adapt to the usage of GPU and parallelization of operations.
On the other side, SIFT and BoW don't provide a hierarchy of representations (as CNNs do) and several operations are needed to form the definitive feature vectors. Thus the procedure is expensive in terms of computational time, as we saw also during our analysis.

7. Conclusion and future perspectives

From the comparison analyzed in this paper, Convolutional Neural Networks emerge as best model for food recognition on iFood19 challenge dataset.
The number of samples and classes, the presence of noise and different classes with very similar samples (in some cases hardly distinguishable by a human eye), the unbalanced cardinalities of the classes and the limitation on the number of CNN's parameters are the main challenges of this task.

We believe that with a data cleaning procedure to discard all non food images (for example exploiting anomaly detection methods) and more than one million parameters allowed for the CNN, the performance could increase.
Thus in the future we will go in this direction to improve performances on recognition tasks, which find applications not only in the culinary context, but also in healthcare and manufacturing. Thanks to the structure of CNNs, the more general features extracted in the first layers can be re-used for a variety of tasks, for example employing transfer learning.

8. Authenticity statement

The authors declare that the content of this report was entirely written by themselves, taking inspiration from the sources listed in the references and from Professor Bianco's slides. No Large Language Model was used to generate contents.

References

[1] Lu Yuzhen, *Food Image Recognition by Using Convolutional Neural Networks (CNNs)*, 2016, https://www.researchgate.net

[2] https://github.com/docongminh/Classification-SIFT-SVM/blob/master/README.md

[3] https://medium.com/@sreeku.ralla/activation-functions-relu-vs-leaky-relu-b8272dc0b1be

[4] https://www.baeldung.com/cs/svm-multiclass-classification

[5] L. Fei-Fei and P. Perona, *A Bayesian hierarchical model for learning natural scene categories*, 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, San Diego (USA), 2005, pp. 524-531 vol. 2, doi: 10.1109/CVPR.2005.16, https://ieeexplore.ieee.org

[6] D.G. Lowe, *Distinctive Image Features from Scale-Invariant Keypoints*, International Journal of Computer Vision, 2004, pp. 91–110 vol. 60, doi: 10.1023/B:VISI.0000029664.99615.94, https://link.springer.com

[7] https://medium.com/@deepanshut041/introduction-to-sift-scale-invariant-feature-transform-65d7f3a72d40

[8] https://github.com/karansikka1/iFood-2019