CentraleSupélec

DeepL - Apprentissage profond (2021-2022)

# PROJECT REPORT : Food classification from images

**group members**
Wassim MECHERGUI

April 10, 2022

**Abstract**

Documenting dietary patterns and caloric intake is an important element to individuals worldwide. It's crucicial for many suffering of health concerns (obesity, allergies), or for some others for more personal usages (documenting caloric intake and foods for muscle training or weight loss) and also for professional athletes. Our work focuses primarily on recognizing foods from an image. We use a classic transfer learning approach on the Food101 dataset to evaluate various deep convolutional architectures and asses their usability. We explore in this article various deep features extractors and classifiers on the Food101 dataset and compare their top 1 precision scores. It's important to note our system classifies food images into a fixed number of types of foods. However, this work can be seen as a precursor to few shot learning techniques for general food recognition.
**Keywords** — Deep Learning, Convolutions, Fine-tuning, Food101

## 1 Introduction

Alot of people worldwide document their dietary habits and this for various reasons (Health, Fitness ..). With the worldwide usage of mobile phones, this process can be sped up drastically by using mobile phone embedded food recognition architectures. Popular current methods include mainly mobile apps that allow you to document manually the food. Suchs apps have alot of features but the user still has to manually insert at every step the food name and information. Our work comes in the context of speeding this process up by allowing the user to take a picture and obtain the proper category of the food.
Deep learning approaches have already been suggested to tackle this problem as deep computer vision techniques are powerful tools for extracting information from images and classifying them. In this article, we explore and compare various deep architectures for solving the task of food recognition on the Food101 [13] dataset, containing a total of 101 different classes. We introduce the dataset in section 2 and explain our approach in section 3. In section 4 we present our findings and finally we conclude about the work and present future directions in section 5.

## 2 Related work

The first related work is [1], where the authors used a deep convolution neural network trained on the food101 dataset. The architecture used was primarily based on recurring inception modules to build a convolutional feature extractor. On top of this feature extractor they added a fully connected layer to obtain the probabilities of each class. In their papers, they explored 3 different datasets : UEC100, UEC256 and Food101 and they achieved a top 1 accuracy of 76.3% and a top 5 accuracy of 94.6%. Various other works have appeared as others started to use different feature extractor architecture, notably [2] used a VGG16 architecture with a simple dense classifier. This work comes also as an inspiration to explore the various pre-existing deep convolutional feature extractors that exist, notably [3] and [4] all while performing data augmentation techniques such as the ones seen in [5].

In [13], where the Food101 dataset was initially proposed, the authors used a different method to classify the images into their corresponding categories. They proposed a random forest based approach to mine discriminative features on the image and later on use an SVM to perform the classification.

In [14], the authors used a different approach where the goal of the system is to asses the total calorie a dish has based purely from the image output. This using an object detector that detects the dishes and a reference object in the image. Once the object is detected, a neural network classifies the food into its category and they use an algorithm based on other detected objects in the scene to decide the dimensions of the dish. Based on these two informations they match accordingly and calculate the caloric intake.

## 3 Dataset overview

The Food101 dataset contains a total of 101000 images with a total of 101 class. For each class there are 1000 images split into 75% training and 25% testing. This dataset was initially proposed in [13] where the authors wanted to create a system that would extract discriminative features on various dishes.
The Training samples are left intentionally noisy and not cleaned. This noise comes primarily in the form of intense colors and sometimes wrong labels.
The images have varying widths but all a same height of 500 and are in 3 channels (RGB).

Figure 1: Various images from the Food101 dataset

## 4 Proposed approach

### 4.1 Model training

In this article, we propose various deep learning approaches to tackle the issue of classifying the images in the dataset. For this, we follow a very similar approach where we build a deep convolutional feature extractor pretrained on the imagenet dataset [6] and attach a fully dense classifier head on top of it. At first, we train only the classifier on the dataset for a total of 10 epochs with a learning rate of $10^{-3}$. This is done with an early stopping callback with a patience of 2 tracking the first 15% of the testing data as validation.
After this, we unfreeze the feature extractor aswell and train the entire model on the dataset for a maximum of 100 total steps. We use a learning rate of $10^{-4}$ with a reduction on plateau strategy. For this strategy, whenever the tracked validation loss stops improving for 2 epochs the learning rate is reduced by a factor of 70%. Moreover, we also used an early stopping callback with a patience of 5.
Mixed-Precision was also used to speed up training times but according to [7] this potentially results in model performance drops. We used the Adaptive Moment estimation optimizer (ADAM) during both steps as well as the sparse categorical cross entropy loss and with a batch size of 32.

Finally, images were resized to (224, 224) before introducing them to the feature extractors. The figure down below shows the overall model architecture with the training process.

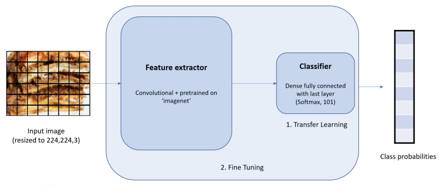


Figure 2: General architecture overview

### 4.2 Explored architectures and variations

In our work work, we explored for the feature extractor 3 different backbones. We ensured that all had a close number of parameters (with 15% each) and the total number of parameters was less than 10 million, as our approach should run ideally on the mobile devices (with quantization). Moreover, we explored two different types of classifiers, the first is a simple dense layer with a softmax activation equivalent on its own to a logistic classifier, the second is made of a dense layer with 256 units and ReLU activation fully connected to the final prediction dense layer.
We have also explored the effect of several classic data augmentation techniques, notably Random flipping, Random Rotation, Random contrast and random brightness adjustment. This is done by training the models first with the data augmentation then with it.
The tables down below summarize the information on the various architectures the explored, aswell as the various data augmentation techniques that were tested against not using them.

| Feature extractor | N params |
|---|---|
| EfficientNetV2B1 | 8.2M |
| EfficientNetB2 | 9.2M |
| DenseNet121 | 8.1M |

Different backbone architectures used

| Data augmentations | arguments |
|---|---|
| Random Flip | Horizontal and vertical |
| Random Rotation | $\alpha \in [-0.2\pi, 0.2\pi]$ |
| Random Contrast | Random factor of 0.1 |
| Random Brightness | Random factor of 0.1 |

Data augmentation techniques considered

| Classifier | parameters |
|---|---|
| Type 1 | Softmax Dense with 101 outputs |
| Type 2 | Relu Dense 256 w Softmax 101 |

Feature classifiers used

#### 4.2.1 EfficientNetB2

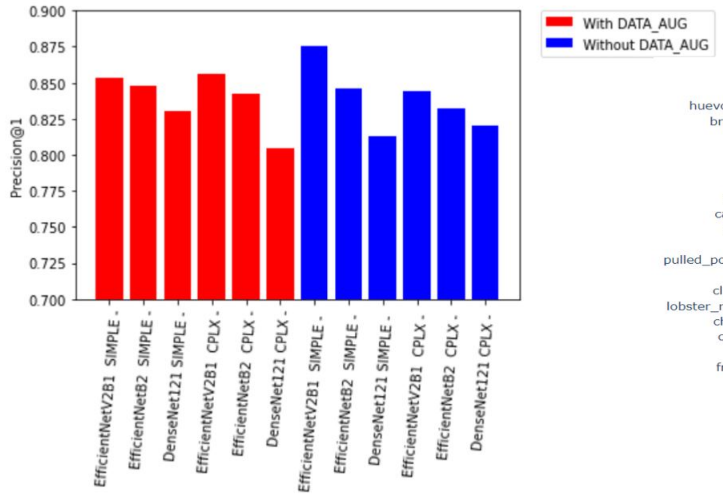
EfficientNetB2 [3] is a model part of the EfficientNet model architectures. The main premise of this class of models is that they were produced with the intention of optimizing on the number of channels and width of the convolutional blocks to obtain a better accuracy while penalizing the model for slower performance. EfficientNetB2 is one of the lightweight versions of this family of models.

### 4.2.2 EfficientNetV2B1

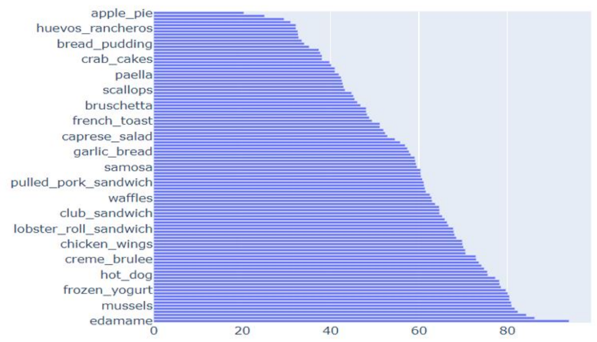
EfficientNetV2 [9] is an improvement of the original EfficientNet architecture where the authors used more modern techniques like MBConv [8] in the earlier layers while also giving a preference for smaller 3x3 kernel sizes. EfficientNetV2B1 is on the lightweight side of this family of models (computationally and memory-wise).

### 4.2.3 DenseNet121

DenseNet [4] is a family of models where each two convolutional blocks are tied with a fully connected dense block, where are all layers (with matching feature-map sizes) are connected to each other. each layer obtains additional inputs from all preceding layers and passes on its own feature-maps to all subsequent layers in order to preserve the feed forward nature of the model.



Precision @ 1 Results



Most common words by errors

Figure 3: Results on the various combinations

## 5

## .Results - Comparison between model results on transfer learning

From the results we have obtained on our model search, we found that the best Precision at 1 of the model was for the EfficientNetV2B1 with the Simple head and no data augmentation with a value of 0.872.

More over, we notice that the usage of data augmentation has a very slight negative effect on the results. It's because one of the augmentation is random vertical flipping wheras the images in the dataset are all placed at the center with a very straight (point upwards) orientation. It's best to not use this sort of augmentation on this dataset.

The Complex classifier head doesn't seem to have a big effect on the performance except for the DenseNet model where it performed the worst with the data augmentation techniques.

Moreover, the DenseNet architecture has lower performances than the EfficientNet architectures on average which is the same result we usually find common benchmarks between those two.

Additionally, the EfficientNetV2B1 model seems to perform better than its earlier counterpart on average, which isn't valid for most benchmark datasets. The right subfigure of the Figure 3 is the list of most frequent word by the average number of errors on all the obtained models. We notice words like edamame having the most errors. Moreover, we can notice the existance of items that all visually look the same and therefore are harder for the model to learn to differentiate. Such examples include : all the dishes that contain sandwich in the name, garlic bread and french toast, etc... Finally, the average precision on all the models of the models before fine-tuning (simple head training) was around 79% and without any significant difference when changing the head of the model. The best result at this step was with DenseNet with Complex Head and without data augmentation with a value of 81.1%

## 6 Conclusion

In this article, we have explored several possible architectures used with transfer learning and finetuning on the Food101 dataset. While the subject of using a deep convolutional neural network to classify images on the Food101 has already been done, we explored various other possibilities and alterations of this approach while documenting the results of each model. The best model we have found has a top 1 precision of 0.872. The obtained model remains however realistically still unusable in an industrial mass scale food recognition scenario as the total number of classes trained on in this dataset is only 101. One possible approach to tackle this issue is to use the fine-tuned feature extractors with few shot learning techniques such as siamese networks [9] to be able to learn potentially alot more new classes with few training samples.

## References

[1] Chang Liu, Yu Cao, Yan Luo, Guanling Chen, Vinod Vokkarane, Yunsheng Ma, DeepFood: Deep Learning-Based Food Image Recognition for Computer-Aided Dietary Assessment https://arxiv.org/abs/1606.05675

[2] Joseph Miguel, kaggle https://www.kaggle.com/code/ac2zoom/vgg16-pytorch-transfer-learning-from-imagenet

[3] Mingxing Tan, Quoc V. Le, EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks https://arxiv.org/abs/1905.11946

[4] Gao Huang, Zhuang Liu, Laurens van der Maaten, Kilian Q. Weinberger, Densely Connected Convolutional Networks https://arxiv.org/abs/1608.06993

[5] Connor Shorten , Taghi M. Khoshgoftaar, A survey on Image Data Augmentation for Deep Learning https://journalofbigdata.springeropen.com/articles/10.1186/s40537-019-0197-0

[6] Jia Deng; Wei Dong; Richard Socher; Li-Jia Li; Kai Li; Li Fei-Fei, ImageNet: A large-scale hierarchical image database https://ieeexplore.ieee.org/document/5206848

[7] Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, Hao Wu, Mixed Precision Training https://arxiv.org/abs/1710.03740

[8] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, Liang-Chieh Chen, MobileNetV2: Inverted Residuals and Linear Bottlenecks https://arxiv.org/abs/1801.04381

[9] Mingxing Tan, Quoc V. Le, EfficientNetV2: Smaller Models and Faster Training https://arxiv.org/abs/1801.04381

[10] Siamese Neural Networks for One-shot Image Recognition, Gregory Koch,Richard Zeme,Ruslan Salakhutdinov https://www.cs.cmu.edu/~rsalakhu/papers/oneshot1.pdf

[11] K. He and J. Sun, "Convolutional neural networks at constrained time cost," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 5353-5360.

[12] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, et al., "Microsoft coco: Common objects in context," in Computer Vision–ECCV 2014, ed: Springer, 2014, pp. 740-755.

[13] Bossard, Lukas and Guillaumin, Matthieuand Van Gool, Luc", Food-101 – Mining Discriminative Components with Random Forests, Computer Vision – ECCV 2014, Springer International Publishing, isbn "978-3-319-10599-4" https://data.vision.ee.ethz.ch/cvl/datasets_extra/food-101/

[14] Parisa Pouladzadeh;Pallavi Kuhad; Sri Vijay Bharat Peddi; Abdulsalam Yassine; Shervin Shirmohammadi Food calorie measurement using deep learning neural network https://ieeexplore.ieee.org/document/7520547