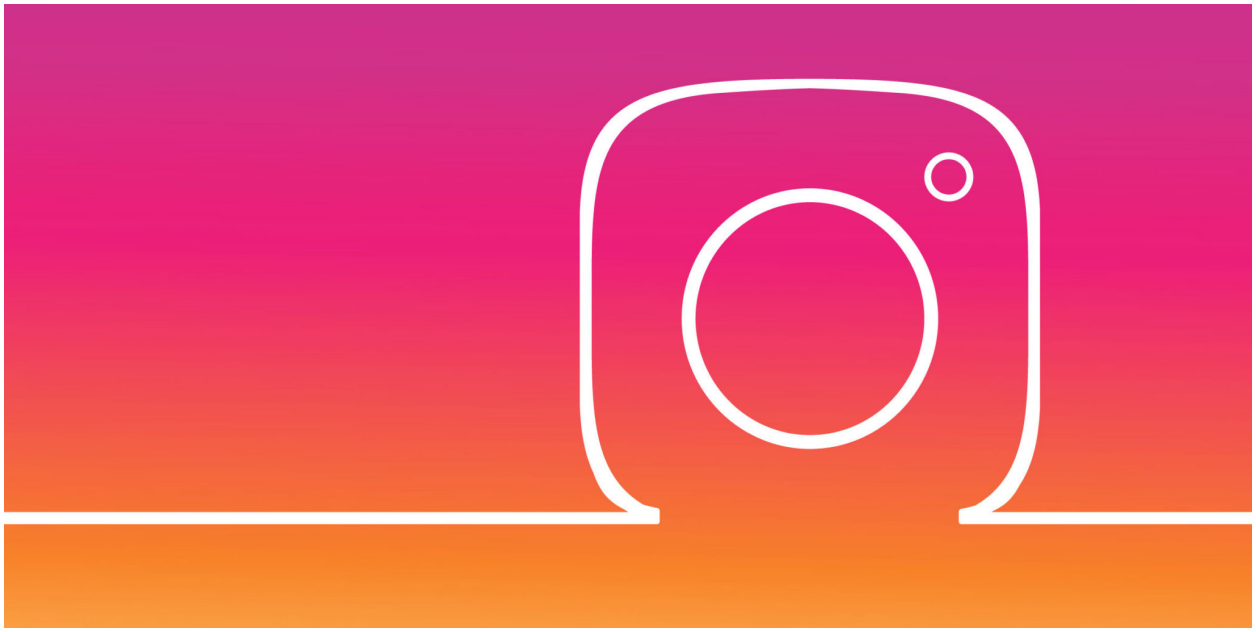


Food ads on Instagram

A new ads targeting algorithm based on user-generated content

07.06.2021

Valentina Rizzati
Data Scientist
Instagram

Opportunity

I am a Data Scientist at Instagram and I've been asked to come up with an idea to increase the company's Ads Revenue. The main objective is to build a tool allowing businesses to target users with very relevant ads, therefore creating both a pleasant in-app experience for users and simultaneously improving the targeting and increasing revenues for businesses.

As part of my prototype, I will focus on building a tool for **food-related businesses** (e.g. restaurants, subscription boxes, food producers). I will leverage user-generated content in the form of images posted on Instagram to classify the image (e.g. *ravioli* in the example above) with as much accuracy as possible; advertisers can then use this information (feature built in the Instagram business portal) to both boost their targeting capabilities as well as unlock new audiences that they were not aware of.

Impact Hypothesis

ADVERTISERS ON INSTAGRAM

If advertisers can improve their targeting algorithm, they will serve relevant ads to users who, in turn, will be more likely to convert and act on those ads (e.g. purchase the pasta box or reserve a seat at the advertised Italian restaurant). More accurate targeting will increase the advertiser's revenues and, in turn, Instagram ads revenues and advertisers' retention on the Instagram ads platform.

INSTAGRAM

By better understanding user-generated content (images), Instagram will be able to serve more relevant organic content to users on Reels. This will increase the number of Daily Active Users (DAU), Time Spent in App and, ultimately, app stickiness.

Data

I have collected image data from the Food 101 dataset, containing 101000 images labeled according to 101 food categories.

For each class, 250 manually reviewed test images are provided as well as 750 training images. On purpose, the training images were not cleaned, and thus still contain some amount of noise - which comes mostly in the form of intense colors

and sometimes wrong labels. All images were rescaled to have a maximum side length of 512 pixels.

The directory was already fairly well structured. The only relevant adjustment was for me to read in the labels included in the *test.json* and *train.json* in the metadata folder and create - through the os library in jupyter notebook - two additional directories - test and train - that themselves contained the 101 folders corresponding to the food classes labels.

Algorithm

The algorithm will be presented in line with how the work was structured across the 4 main jupyter notebooks.

DATA LOADING & EDA

In this section I loaded the image data and structured it in *test*, *train* and *metadata* folders. Within the *train* and *test* folders I have included all the 101 food classes image folders.

As a second step, I conducted EDA by visualizing images belonging to different food classes and plotting image width and height.

BASELINING

As part of this step, I first conducted preprocessing by resizing and grayscaling the images and then ran a multi-class random forest as my baseline model. Unsurprisingly, the baseline performance was poor.

Hence, I proceeded with building a Convolutional Neural Network (CNN) to improve model performance.

TESTING CNN ON 3 FOOD CLASSES

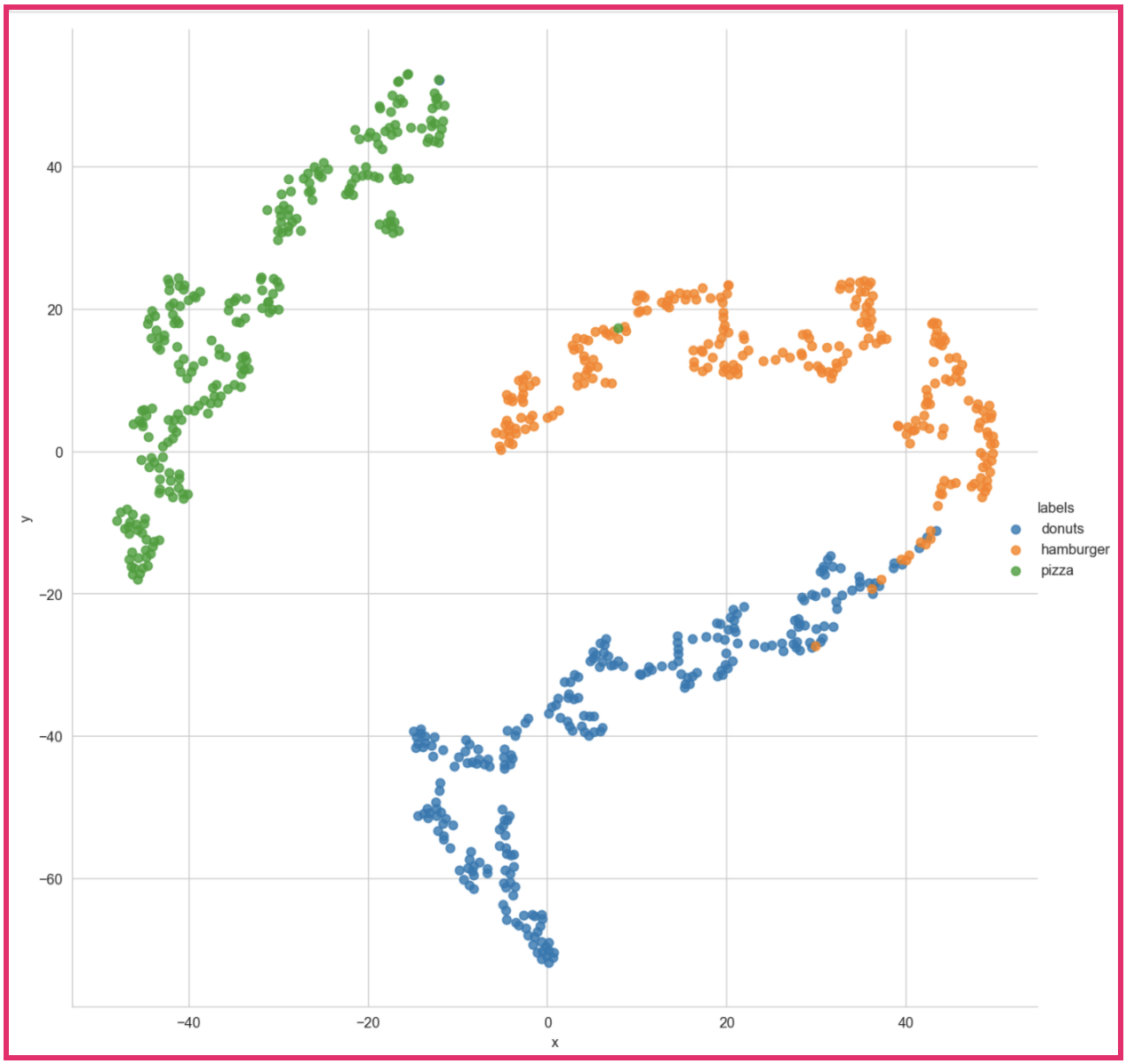
I first tested my image classifier prototype on Jupyter Notebook by running the model on three food classes only (*donuts*, *hamburger* and *pizza*). I implemented the transfer learning methodology by leveraging the ResNet50 pre-trained model, one of the most popular image classification CNNs.

In particular, I adopted the techniques of Mixup Image Augmentation, One Cycle Policy, and discriminative learning rates.

I then ran the model predictions both with Test Time Augmentation (TTA) and without it. I obtained a final **top-1 accuracy of 98.5%** and **top-5 accuracy of 99.9%**.

Due to the high performance of this image classifier, I did not see the need of testing new pre-trained models at this stage.

The TSNE plot below shows a relatively successful clustering of the three food classes. Donuts and hamburgers are sometimes confused for each other, probably due to the similar shape and bread-looking color.



SCALING CNN TO ALL 101 FOOD CLASSES

Finally, I am now in the process of scaling my image classifier to the full dataset of 101 food classes. I will run this last part of my project on Google Colab with GPU and high-RAM runtime settings.

Tools

For general data manipulation, baseline modeling and visualization, I have used pandas, numpy, and scikit-learn.

To conduct image classification through neural networks I have used fast.ai, a deep learning library that is wrapped around PyTorch. The reason I decided to use fast.ai is that I was faced with a fairly large dataset (101000 images across train and test set) including some image classes that are fairly similar to each other (e.g. steak and filet mignon), hence straining the algorithm's image classification capabilities. Therefore, I needed to ensure I was maximizing my chances in boosting accuracy from the start. After reading a few papers and blogs, I realized that fast.ai provides easily applicable methods and more best practices baked in, so that the image classification algorithm is normally faster to train and delivers higher accuracy.

In fact, after working with this library I was able to adopt techniques like one cycle learning, learning rate finder, and differential learning rates in a fairly straightforward manner.

Next Steps

To finalize the project, I will scale the CNN previously tested on 3 food classes to all the 101 food classes by using Google Colab.