Datasets for Nutrify - Bryan Cheng, NYU Zhengyuan Zhou Lab

1. UEC-Food256
    a. Summary
        i. 31k images in jpeg form
        ii. 256 dish classes
        iii. Annotations: 1 bbox per food region stored in a per-image .txt(label x y w h)
        iv. File layout: root/{class_name}/{img}.jpg + root/{class_name}/{img}.txt
        v. Compressed size is about 4 GB
    b. Details
        i. Japanese centric
        ii. Pairs classification labels and bounding-boxes
    c. Sample Record/Quick Access
        i.

```
katsudon/10011.jpg
katsudon/10011.txt →  0  84 138 512 400   # class-id 0 = katsudon
```

        ii.

```
!pip install -q kaggle   # 1-time setup: place your Kaggle token
!kaggle datasets download -d rkuo2000/uecfood256 -f
katsudon/10011.jpg -p data
from PIL import Image; import numpy as np, pathlib
img = Image.open(pathlib.Path('data')/'katsudon/10011.jpg')
print(np.array(img).sum())
```

2. UNIMIB 2016(Food Tray)
    a. Summary
        i. 1027 cafeteria-tray photos
        ii. 73 food categories, 3616 instance polygons(also provided as bounding boxes)
        iii. Masks in COCO-style JSON, RGB images, 640x480
        iv. 1.1GB
    b. Details
        i. Multi-item, italian centric
        ii. Ideal for multi-food detection, early ingredient segmentation
        iii. Realistic tray composition under indoor lighting
    c. Sample Record/Quick Access
        i.

```
{
```

```
  "file_name": "tray_0345.jpg",
  "height": 480,
  "width": 640,
  "annotations": [
    {"category_id": 12, "bbox": [186,72,210,140], "segmentation":
[[…]]},
    …
  ]
}
```

ii.

```
from datasets import load_dataset
ds = load_dataset("unisg/unimib2016", split="train[:1]")
img = ds[0]["image"]
print(img.numpy().sum())
```

3. FoodSeg-103/FoodSeg-154
    a. Summary
        i. 9490 images, 154 adds +5% amount of images
        ii. 103(or 154) ingredient labels, averaging 6 masks per photo
        iii. Pixel masks in PNG
        iv. 2.2 GB
    b. Details
        i. First large-scale ingredient level segmentation set
        ii. Can help model learn what/where the ingredients are
    c. Sample Record/Quick Access
        i.

```
images/pizza_0012.jpg
masks/pizza_0012.png   # 0=background, 4=cheese, 17=tomato-sauce…
```

        ii.

```
from datasets import load_dataset
seg = load_dataset("EduardoPacheco/FoodSeg103", split="validation",
streaming=True)
first = next(iter(seg))
print(first["image"].numpy().sum())
```

4. Nutrition5k
    a. Summary
        i. 5006 unique plates
        ii. Per-plate: 4 side-angle videos, overhead RGB-D, recipe list, ingredient masses, calories
        iii. 181GB on public GCS bucket
        iv. COCO JSON + CSV nutrition tables

b. Details
    i. Good for single-image volume estimation
    ii. Good for macro-nutrient regression benchmark
c. Sample Record/Quick Access

```
imagery/overhead/000123/rgb.jpg
meta/000123.json → {"total_kcal": 623, "protein_g": 27.5, …}
```

i.


```
imagery/overhead/000123/rgb.jpg
meta/000123.json → {"total_kcal": 623, "protein_g": 27.5, …}
```


5. School-Lunch-21
    a. Summary
        i. 3940 Japanese high school lunch photos
        ii. 21 dish classes + other stuff
        iii. Per-item bounding boxes, plain text
        iv. 900MB
    b. Details
        i. Consistent viewpoint
        ii. Good for training tray-style detectors where multiple dishes are present
    c. Sample Record/Quick Access
        i.

```
img_2401.jpg
img_2401.txt:
  rice_bowl 59 88 240 210
  miso_soup 300 95 215 200
```

ii.

```
import urllib.request, tarfile, numpy as np
urllib.request.urlretrieve(SCHOOL_LUNCH_TAR, "lunch.tar.gz")  # replace with actual link
tarfile.open("lunch.tar.gz").extractmember("images/img_0001.jpg", "tmp.jpg")
from PIL import Image; print(np.array(Image.open("tmp.jpg")).sum())
```

6. MyFoodRepo-273
    a. Summary
        i. 24119 app-captured meals
        ii. 273 classes, 39325 polygon masks
        iii. Public benchmark splits(training, validation, testing) in COCO JSON
        iv. 6GB
    b. Details
        i. Crowd-sourced, smartphone photos

ii. Diverse lighting, cuisines
iii. Good for testing models in “in the wild” settings

c. Sample Record/Quick Access

i.

```
{"image":"mfr_000087.jpg",
 "annotations":[{"category_id":61,"segmentation":[[…]]}]}
```

ii.

```
from datasets import load_dataset
d = load_dataset("salathe/mfr273", split="test", streaming=True)
im = next(iter(d))["image"]
import numpy as np; print(im.numpy().sum())
```

7. UEC-FoodPix Complete
    a. Summary
        i. 10k images, 9k train, 1k test
        ii. 102 dish classes, hand-refined segmentation masks
        iii. Mask PNGs
        iv. 760MB
    b. Details
        i. Pixel-accurate dish masks
        ii. Good for training refined segmentation models
    c. Sample Record/Quick Access
        i.

```
train/000234.jpg
train/000234_mask.png  # 0 background, 27 curry-rice …
```

        ii.

```
tar = "UECFoodPixComplete.tar.gz"          # download from project
page
import tarfile, numpy as np
tarfile.open(tar).extractmember("train/000234.jpg", "x.jpg")
from PIL import Image; print(np.array(Image.open("x.jpg")).sum())
```

8. Roboflow “Dishes” v9
    a. Summary
        i. 4259 photos, multiple dish classes
        ii. YOLO-format, bx1 y1 w h labels (TXT)
        iii. JSON & COCO exports provided automatically
        iv. 460MB
    b. Details
        i. Community-curated
        ii. Built through roboflow API

c. Sample Record/Quick Access
    i.

```
train/images/frame_002121.jpg
train/labels/frame_002121.txt → 0 0.51 0.62 0.34 0.27   # 0 = dish
```

    ii.

```
import roboflow, numpy as np
rf = roboflow.Roboflow(api_key="YOUR_KEY")
project = rf.workspace().project("dishes-aj0a6")
dataset = project.version(9).download("yolov8")
from PIL import Image;
print(np.array(Image.open(dataset.location/"train/images/frame_0001.jpg")).sum())
```

9. UEC-Food100
    a. Summary
        i. 14000 JPEGs
        ii. 100 Japanese dishes
        iii. 1 bounding-box TXT per image
        iv. 1.7 GB
    b. Details
        i. Classific Japanese set for single-item dish detection
        ii. Good starting point
    c. Sample Record/Quick Access
        i.

```
sushi/000123.jpg
sushi/000123.txt   # → "0  85  96  420 390"
```

        ii.

```
from datasets import load_dataset
ds = load_dataset("mjperk/uec_food100", split="train[:1]")
img = ds[0]["image"]; import numpy as np
print(np.array(img).sum())
```

10. Food-101
    a. Summary
        i. 101000 images
        ii. 101 western and asian classes
        iii. Classification only, no boxes
        iv. 5GB
    b. Details
        i. Good for training classification models
    c. Sample Record/Quick Access

i.

```
images/pad_thai/421123.jpg          # train
meta/test.txt → “grilled_salmon/70603.jpg”
```

ii.

```
from datasets import load_dataset
ds = load_dataset("ethz/food101", split="train[:1]")
print(ds[0]["image"].numpy().sum())
```

11. PFID(Pittsburgh Fast-Food Image Dataset)
    a. Summary
        i. 4545 images
        ii. 11 US Chains
        iii. 101 fast-foods
        iv. 303 360 degree videos and 27 eating event videos
        v. 1 GB
    b. Details
        i. Captured in restaurants and a lab
        ii. Good for depth, gives you multi-view shots
    c. Sample Record/Quick Access
        i.

```
PFID/BurgerKing_whopper/img_3.jpg
```

        ii.

```
import urllib.request, zipfile, numpy as np, pathlib, PIL.Image as Image
url = "http://pfid.intel-research.net/Whopper.zip"   # sample pack
urllib.request.urlretrieve(url, "w.zip")
zipfile.ZipFile("w.zip").extract("Whopper/img_1.jpg", ".")
print(np.array(Image.open("Whopper/img_1.jpg")).sum())
```

12. UNICT-FD1200
    a. Summary
        i. 4754 images
        ii. 1200 distinct plates
        iii. 4 angles each
        iv. 1.2 GB
    b. Details
        i. Real-life lighting, background, viewpoint
    c. Sample Record/Quick Access
        i.

```
UNICT-FD1200/class_023/image_023_2.jpg
meta.csv → id,class,tags
```

ii.

```
!wget -q https://iplab.dmi.unict.it/UNICT-FD1200/sample.zip
!unzip -q sample.zip "class_023/image_023_2.jpg"
from PIL import Image, ImageStat
print(sum(ImageStat.Stat(Image.open("class_023/image_023_2.jpg")).sum))
```

13. UNICT-FD889
    a. Summary
        i. 3583 images
        ii. 889 plates
        iii. Variability in rotation, scale, illumination
        iv. 900 MB
    b. Details
        i. Compact forerunner to FD-1200
        ii. Real-scene single-plate shots
    c. Sample Record/Quick Access
        i.

```
UNICT-FD889/plate_00432.jpg
```

        ii.

```
import requests, numpy as np, PIL.Image as Image, io
r =
requests.get("https://iplab.dmi.unict.it/UNICT-FD889/images/plate_00001.jpg")
img = Image.open(io.BytesIO(r.content))
print(np.array(img).sum())
```

14. Vireo-Food 172
    a. Summary
        i. 110241 images
        ii. 172 dishes
        iii. 353 ingredient labels per-dish
        iv. Free for research after email agreement(https://fvl.fudan.edu.cn/dataset/vireofood172/list.htm#download)
    b. Details
        i. Dish to ingredient mappings
    c. Sample Record/Quick Access
        i.

```
images/000001.jpg
labels/ingredients.csv → id,img_path,ingredient_ids
```

ii.

```
# After you receive the links
!curl -O https://example.com/vireo172_shard1.tar
!tar -xf vireo172_shard1.tar images/000001.jpg
from PIL import Image; import numpy as np
print(np.array(Image.open("images/000001.jpg")).sum())
```

15. ChineseFoodNet
    a. Summary
        i. 185628 images
        ii. 208 chinese dishes
        iii. 19.4 GB
    b. Details
        i. Largest single-cuisine corpus
        ii. Intra-class variation
    c. Sample Record/Quick Access
        i.

```
train/042/MapoTofu_000212.jpg
val_list.txt → train/042/MapoTofu_000212.jpg  42
```

        ii.

```
from datasets import load_dataset
ds = load_dataset("chinese_food", split="train[:1]", streaming=True)
img = next(iter(ds))["image"]
import numpy as np; print(img.numpy().sum())
```

16. Recipe1M
    a. Summary
        i. 13 million JPEG images
        ii. 1M+ recipes with title, ingredients, instructions, nutrient slots
        iii. Majority western cuisines
        iv. 85 GB(images) + 2GB(JSON/CSV)
    b. Details
        i. Massive paired text-image data
        ii. Very big very big
    c. Sample Record/Quick Access
        i.

```
{
 "id": 262, "title": "Classic Macaroni and Cheese",
 "images": ["images/262/262564.jpg", …],
 "ingredients": ["2 cups elbow macaroni", "1½ cups cheddar", …],
 "instructions": [...]
}
```

ii.

```
!wget -q https://data.recipes/datasets/recipe1m_images_part1.tar
!tar -xf recipe1m_images_part1.tar images/262/262564.jpg
from PIL import Image; import numpy as np
print(np.array(Image.open('images/262/262564.jpg')).sum())
```

17. ISIA Food-500
    a. Summary
        i. 399726 images
        ii. 500 classes
        iii. Single-dish classification label per image
        iv. 40 GB
    b. Details
        i. Go-to large-scale benchmark, 10x larger than Food-101
        ii. Western centric
    c. Sample Record/Quick Access
        i.

```
ISIA_Food500/Grilled_Steak/00010234.jpg
```

        ii.

```
!aria2c -x6 http://123.57.42.89/Food500/Grilled_Steak_00010234.jpg
from PIL import Image; import numpy as np
print(np.array(Image.open('Grilled_Steak_00010234.jpg')).sum())
```

18. Food2K
    a. Summary
        i. 1036564 images
        ii. 2000 classes, NA/EU heavy
        iii. Class label(folder = dish)
        iv. 133 GB
    b. Details
        i. Largest single-image, western-inclusive food dataset
    c. Sample Record/Quick Access
        i.

```
Food2K/Apple_Pie/IMG_00000045.jpg
```

        ii.

```
!wget -q
https://opendatalab.com/Food2K/Apple_Pie/IMG_00000045.jpg
from PIL import Image; import numpy as np, pathlib
print(np.array(Image.open('Apple_Pie/IMG_00000045.jpg')).sum())
```

19. FoodAI-756

a. Summary
    i. 400000 crowd-sourced images
    ii. 756 food categories
    iii. CSV with image_id, class
    iv. >60% western items
    v. 32 GB
b. Details
    i. Good for apps aimed at cafe/restaurant foods
c. Sample Record/Quick Access
    i.

```
FoodAI_Images/waffle_000543.jpg
annotations.csv → waffle_000543.jpg, 183   # 183 = waffle
```

    ii.

```
!wget -q
https://github.com/foodai/dataset/releases/download/v1.0/waffle_00054
3.jpg
from PIL import Image, ImageStat
print(sum(ImageStat.Stat(Image.open('waffle_000543.jpg')).sum))
```