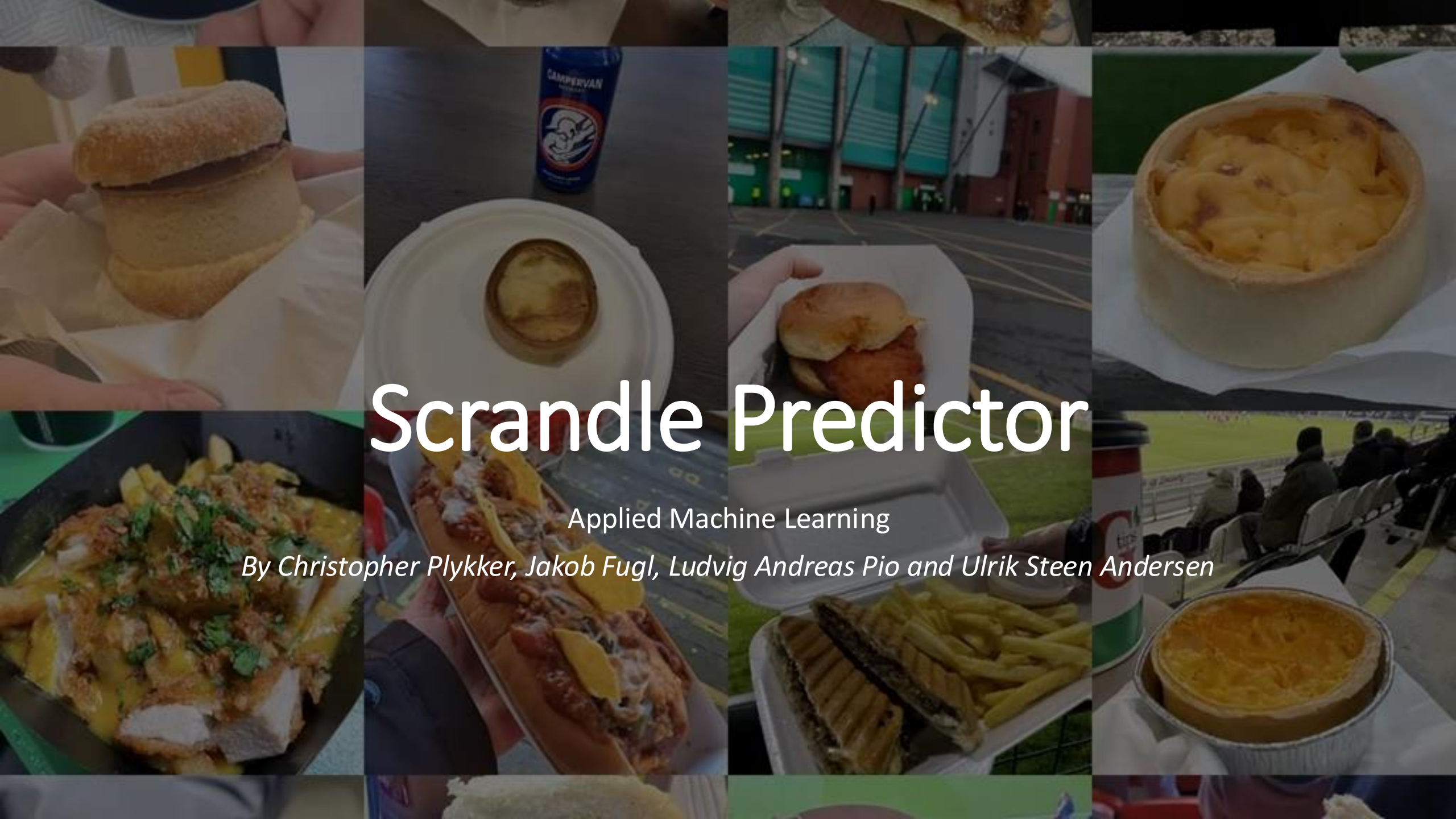


Scrandle Predictor

Applied Machine Learning

By Christopher Plykker, Jakob Fugl, Ludvig Andreas Pio and Ulrik Steen Andersen

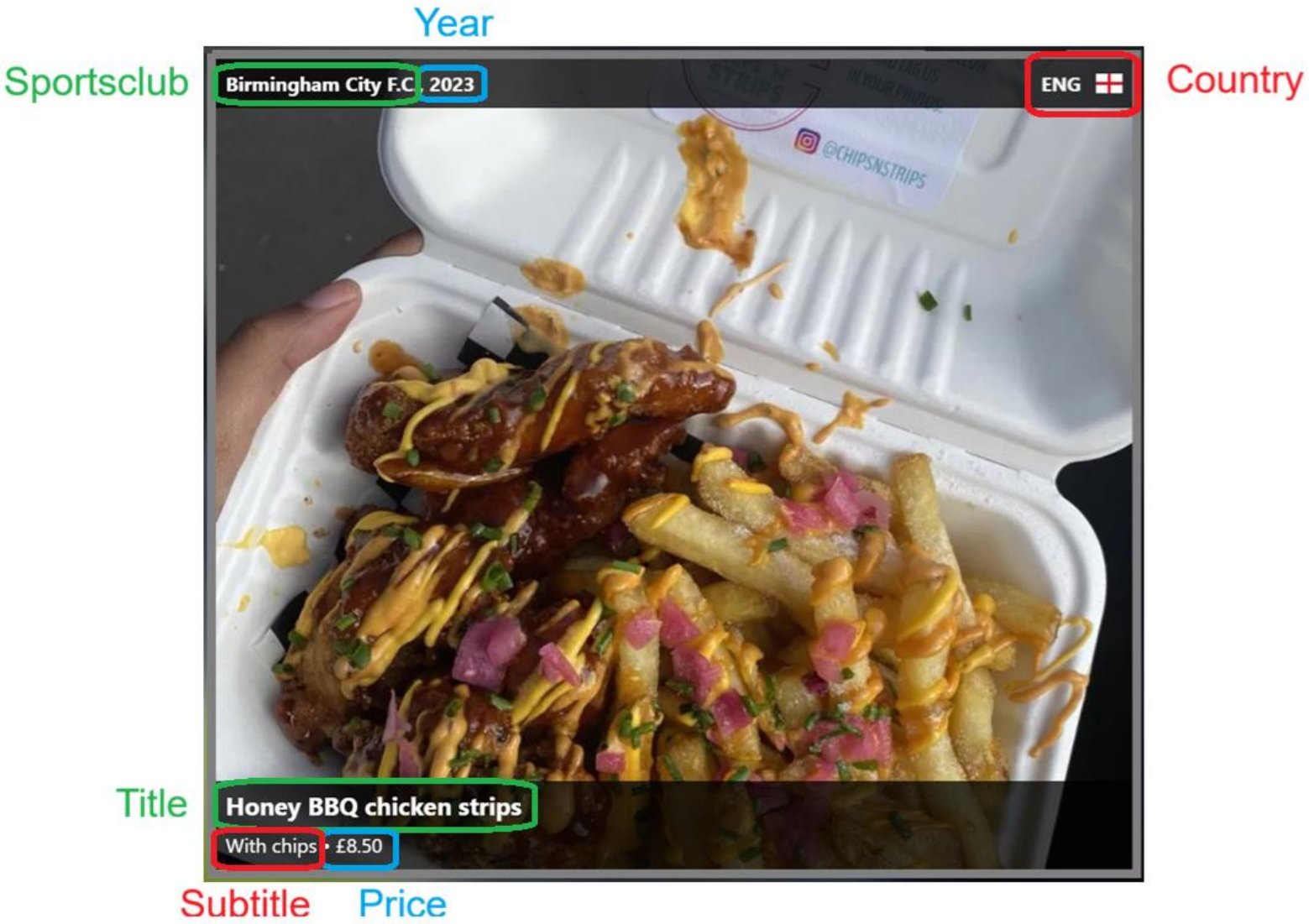
Data

Dates from **2025-04-20** to **2026-05-31** (407 days)

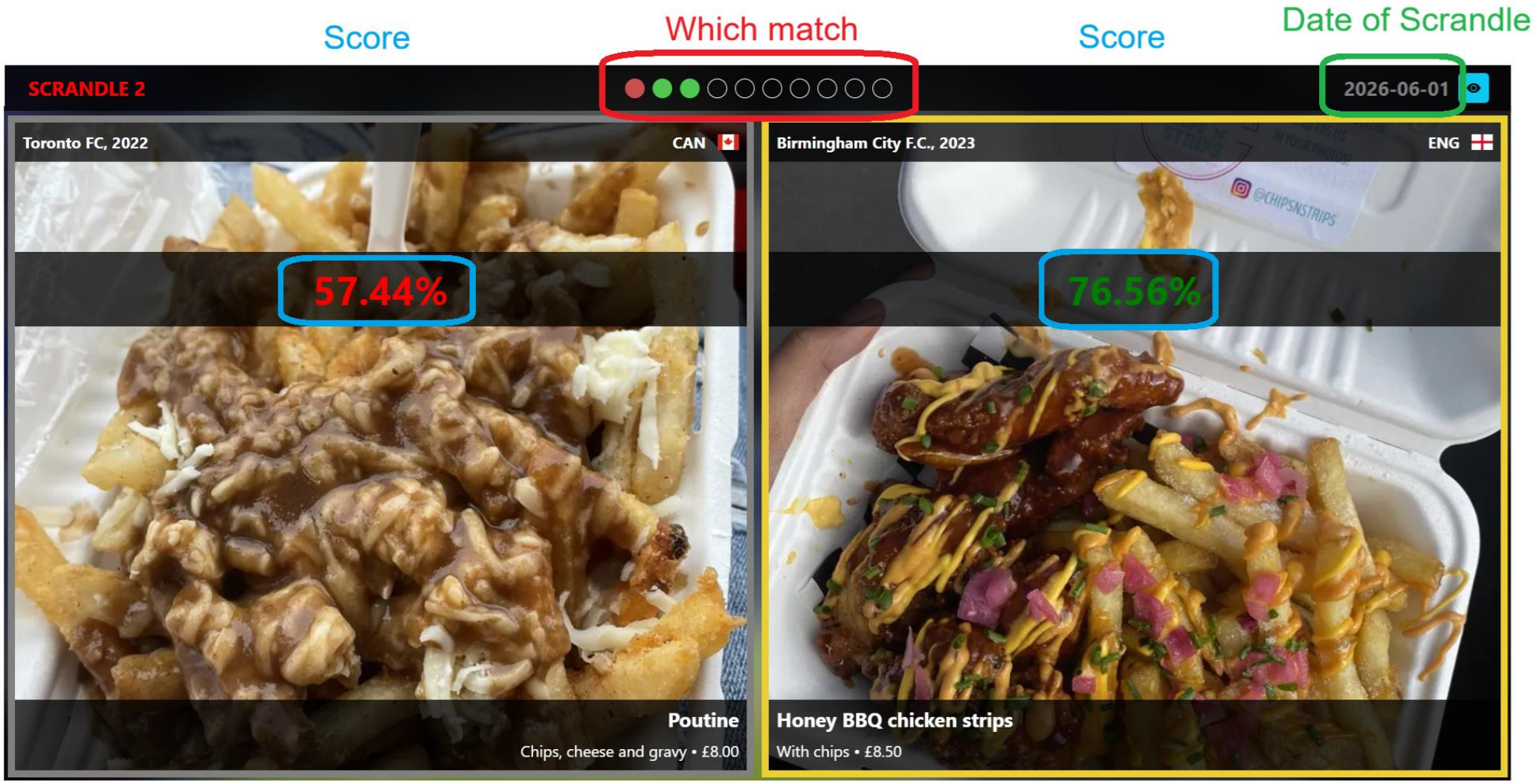
Each date has 10 matches
Each match has 2 foods
Each food has up to 10 metadata of which we work with these 8:

- Picture
- Title
- Subtitle
- Price
- Score
- Position (which date, which match and left or right)
- Sportsclub
- Country

An individual case of Scrandle



How the match is presented to the player



Preprocessing

Raw data

Freshly scraped from scrandle.com

407 days, 4070 matches, 8140 cases

Unique cases

Duplicate cases removed

4827 unique cases

Unique foods

Only foods with unique pictures remain

3193 unique foods, 8 metadata each

Data is WEIRD!

Inconsistent metadata. Price can be:
{0 , “No price” , “ “ , “Free for soldiers”…}

Data contains jokers:
rats, birds and mushrooms are “foods”!

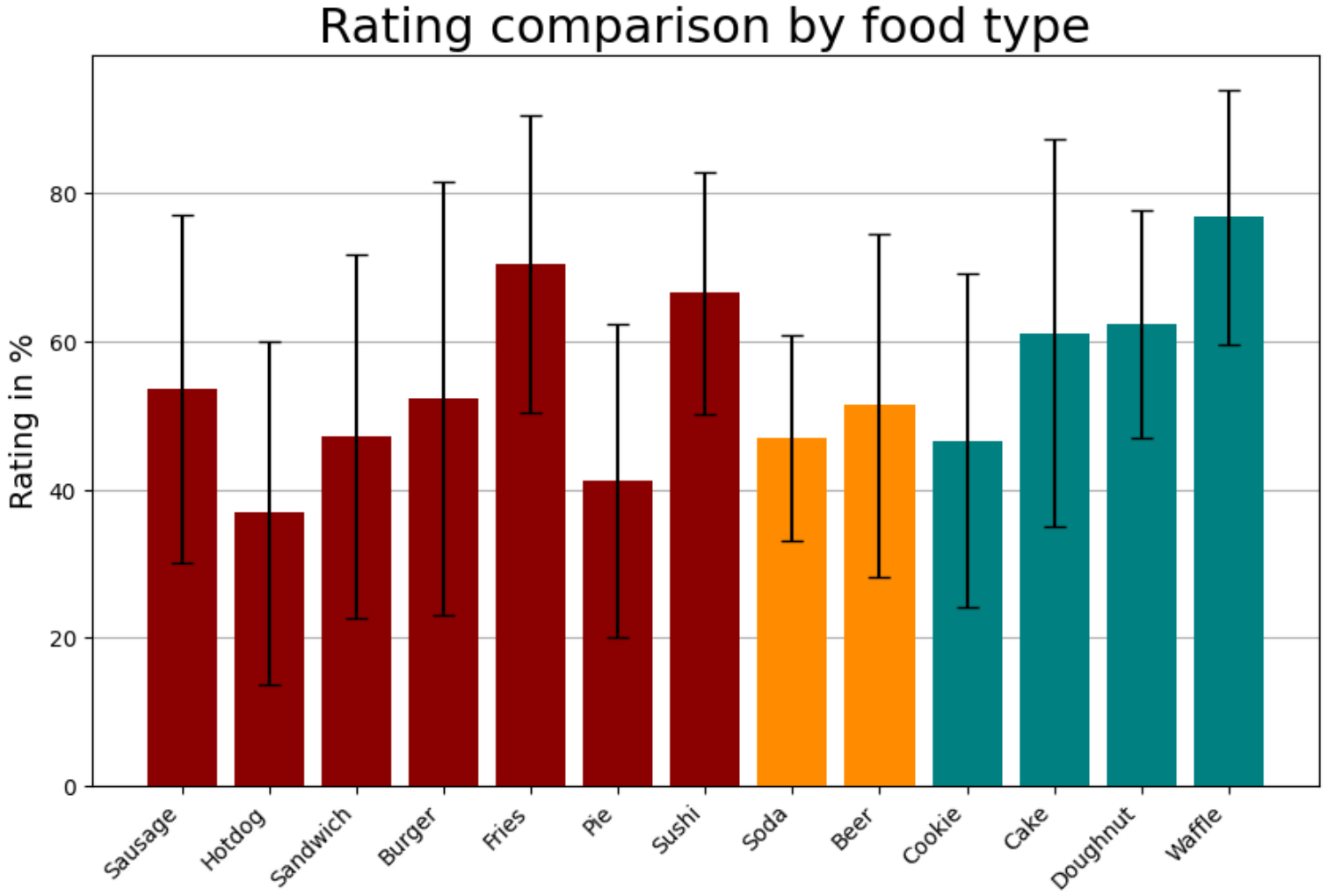
Data visualization, food types

Foods types, found from scraping titles and subtitles for keywords.

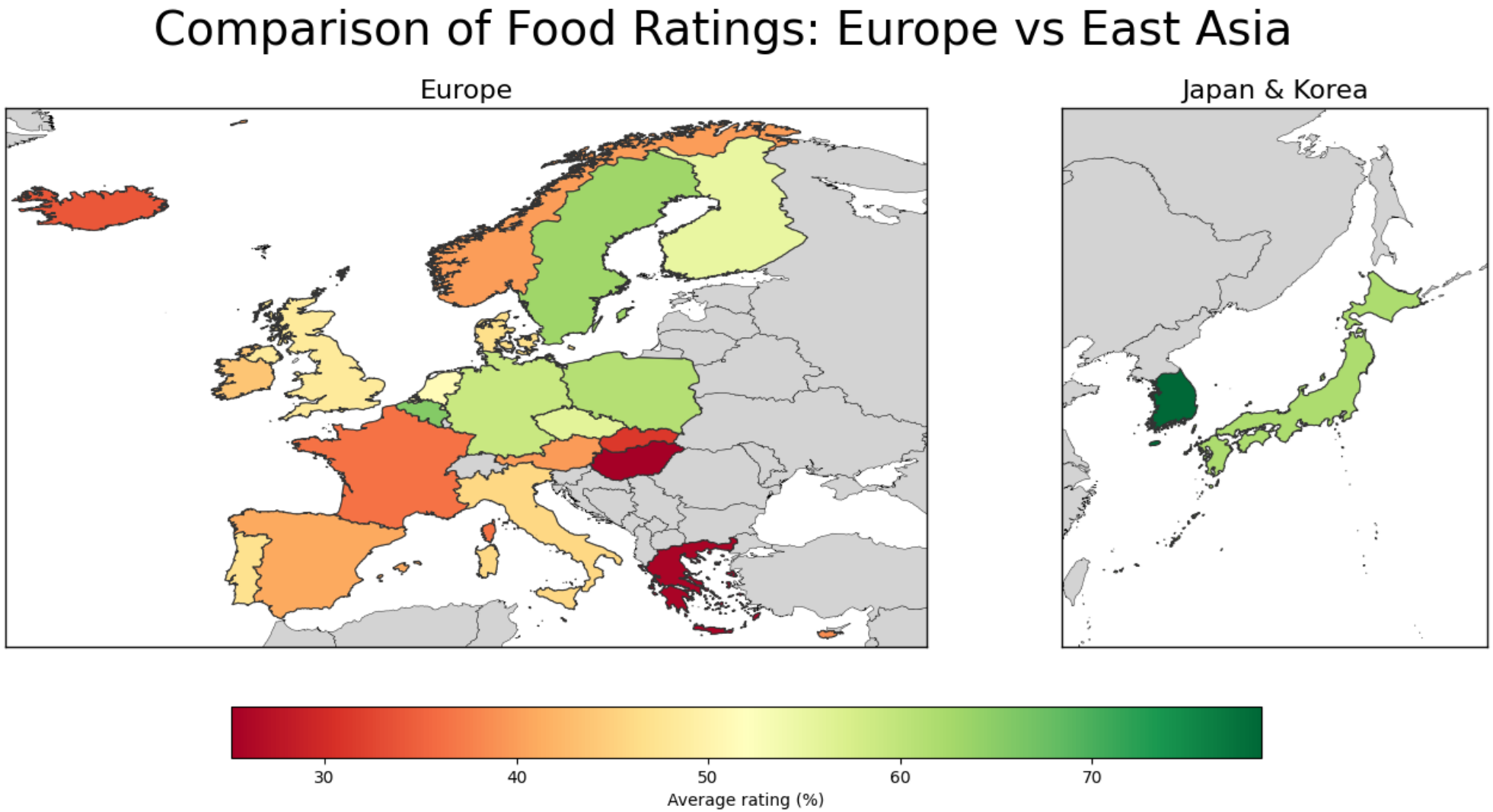
Red: Salty foods

Orange: Drinks

Teal: Sweets/cake



Data visualization, country averages



Example of Hungarian and Korean Scrandle

Hungarian zsíros deszka,

Average Hungarian rating = (25.12±12.78) %

Korean pork steak

Average Korean rating = (78.85±15.08) %

Image texts to tokens: preprocessing

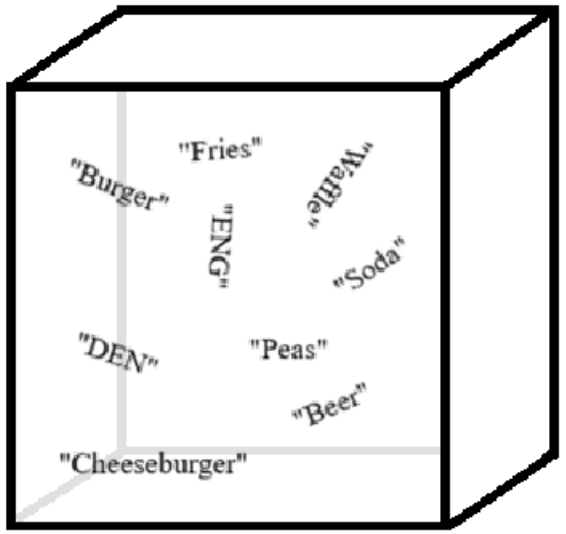
Concatenate separate image texts into one full image text -->
1 image text for each image

Title	Double cheeseburger
Subtitle	With sausage and bacon
Country	ENG
Club	Torquay United
Full image text	”Double cheeseburger with sausage and bacon ENG Torquay United”

Tokenization

- Word --> number called a ”token”: E.g. ”cheeseburger” --> "652"
- We used CountVectorizer from Scikit-Learn

Text data



CountVectorizer

Matrix	Token 652 (Cheeseburger)	...	Token n (Beer)
Image 1 (Text)	1 (Cheeseburger appears once in Image 1)	...	0 (Beer is not in Image 1)
...	...	...	...
Image m	...	...	...

CountVectorizer Representation of Text

Word	Token
double	1028
cheeseburger	652
with	3552
sausage	2816
…	…

”Double cheeseburger with sausage…”	Cheeseburger	…	Double	…	Sausage	…
Token	652	…	1028	…	2816	…
Token count	1	0	1	0	1	…

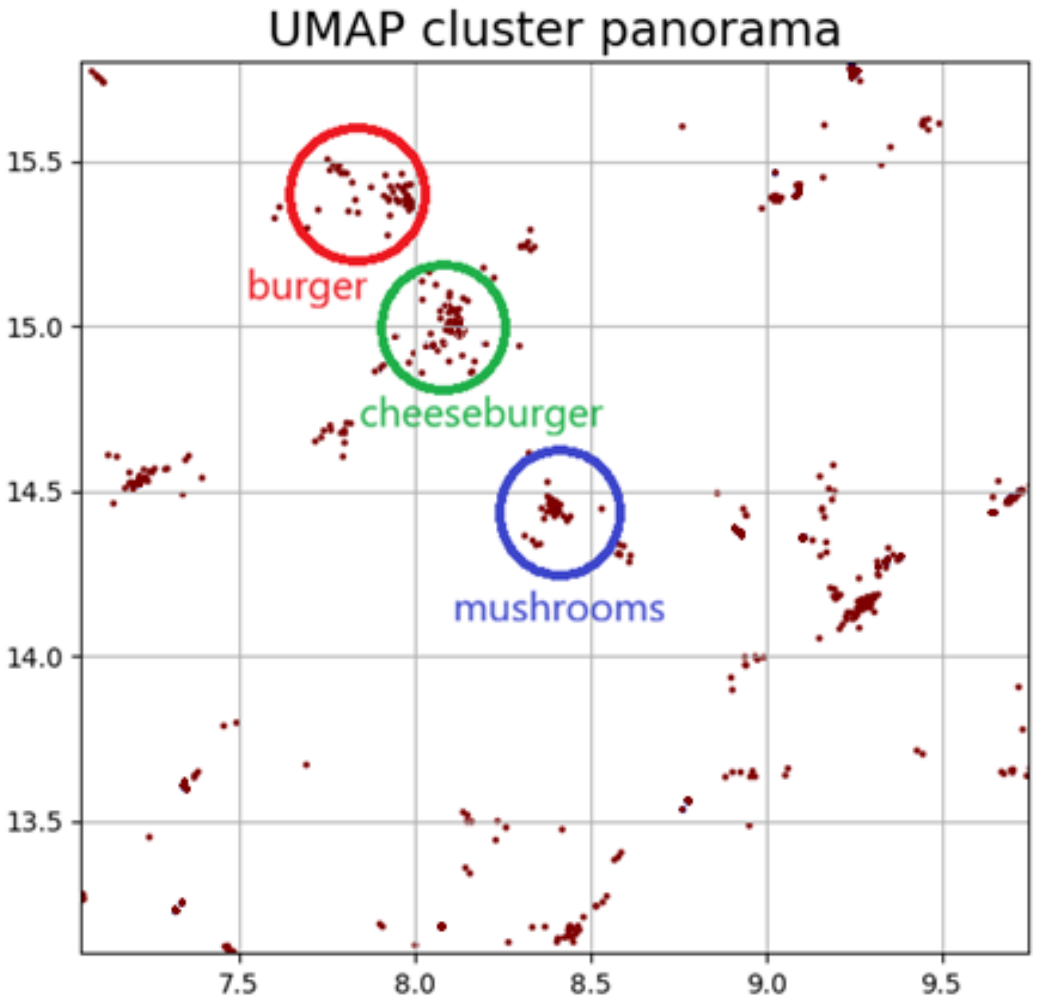
CountVectorizer Matrix as a Tabular Dataset

…	Token n - 1	Token n	Token n +1	…
Image m - 1	0	0	0	…
Image m	0	1	0	…
Image m + 1	0	0	0	…
…	…	…	…	…

- Tokens are dimensions in the data. N tokens --> N-dimensional
- Each image is a data point. M images --> M data points
- This is a tabular dataset of dimension N and size M that can be fed to a neural network

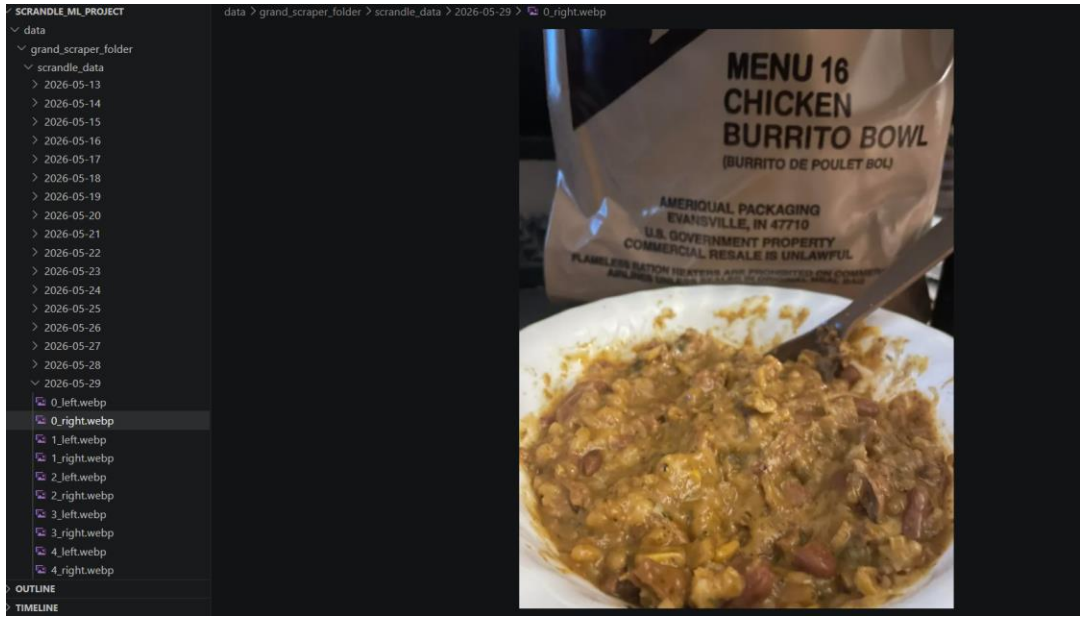
Dimensionality Reduction

- Dimensionality Reduction (UMAP) on the tokens is able to separate categories such as mushrooms, burgers, cheeseburgers
- There are a few outliers in the food clusters however
- E.g. nachos were observed in the mushroom cluster



Outlier Removal – Two Methods

1. Manual Removal



- Easy to implement
- Slow
- Manual
- Sometimes hard to tell what is an outlier

2. Removal Via Clustering

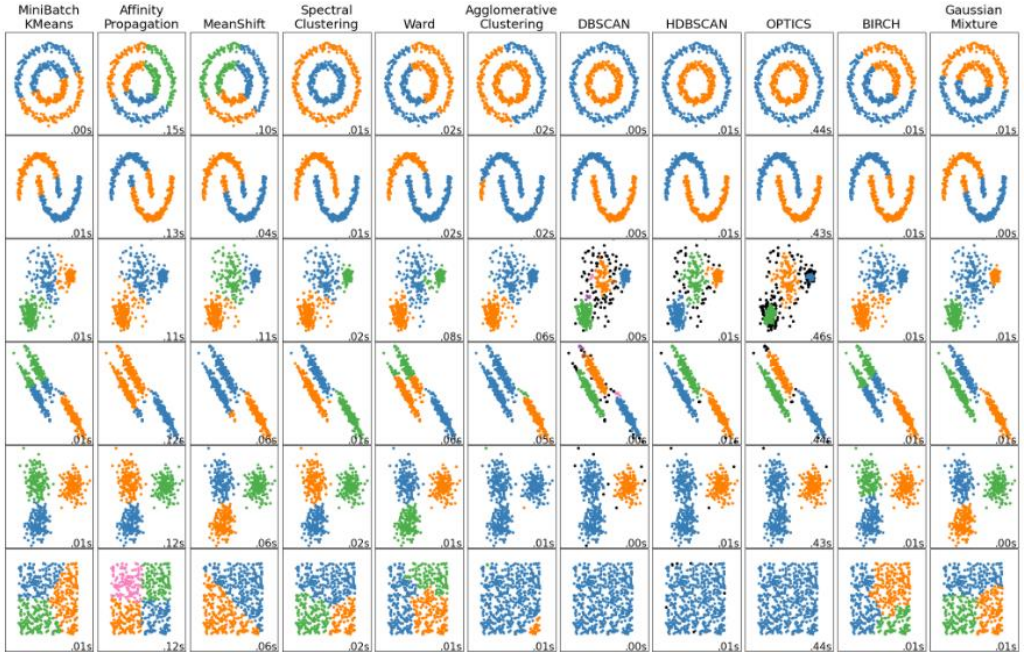
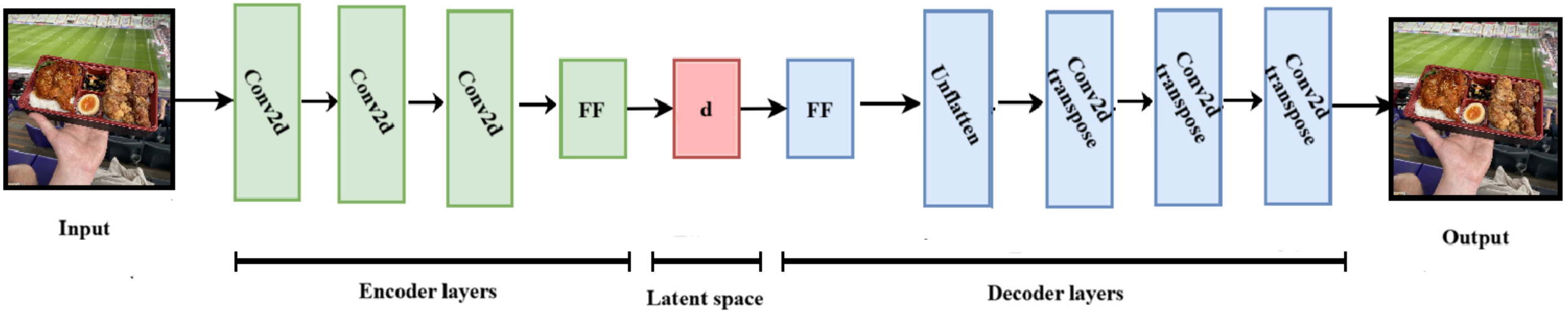


Image taken from: https://scikit-learn.org/stable/auto_examples/cluster/plot_cluster_comparison.html

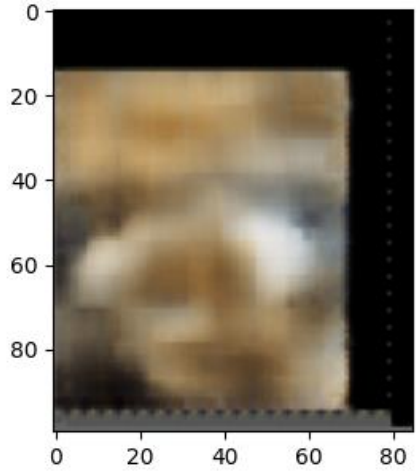
- Fast
- Automatic
- Requires variational autoencoder (VAE) or similar for images

VAE Model and Challenges

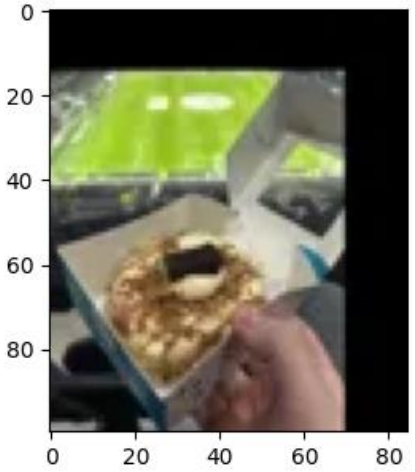


Above image is a modified version of the one found at: https://commons.wikimedia.org/wiki/File:VAE_Basic.png

Recreation



Original



- Normalisation
- Image resolution
- Conv2D vs. Conv2Dtranspose
- GPU(NVIDIA GeForce RTX 4050)
- Struggles with the colour green:
  "*Green? What do you mean?*"
- Padding and output padding

Processing the Images – The Worst Example

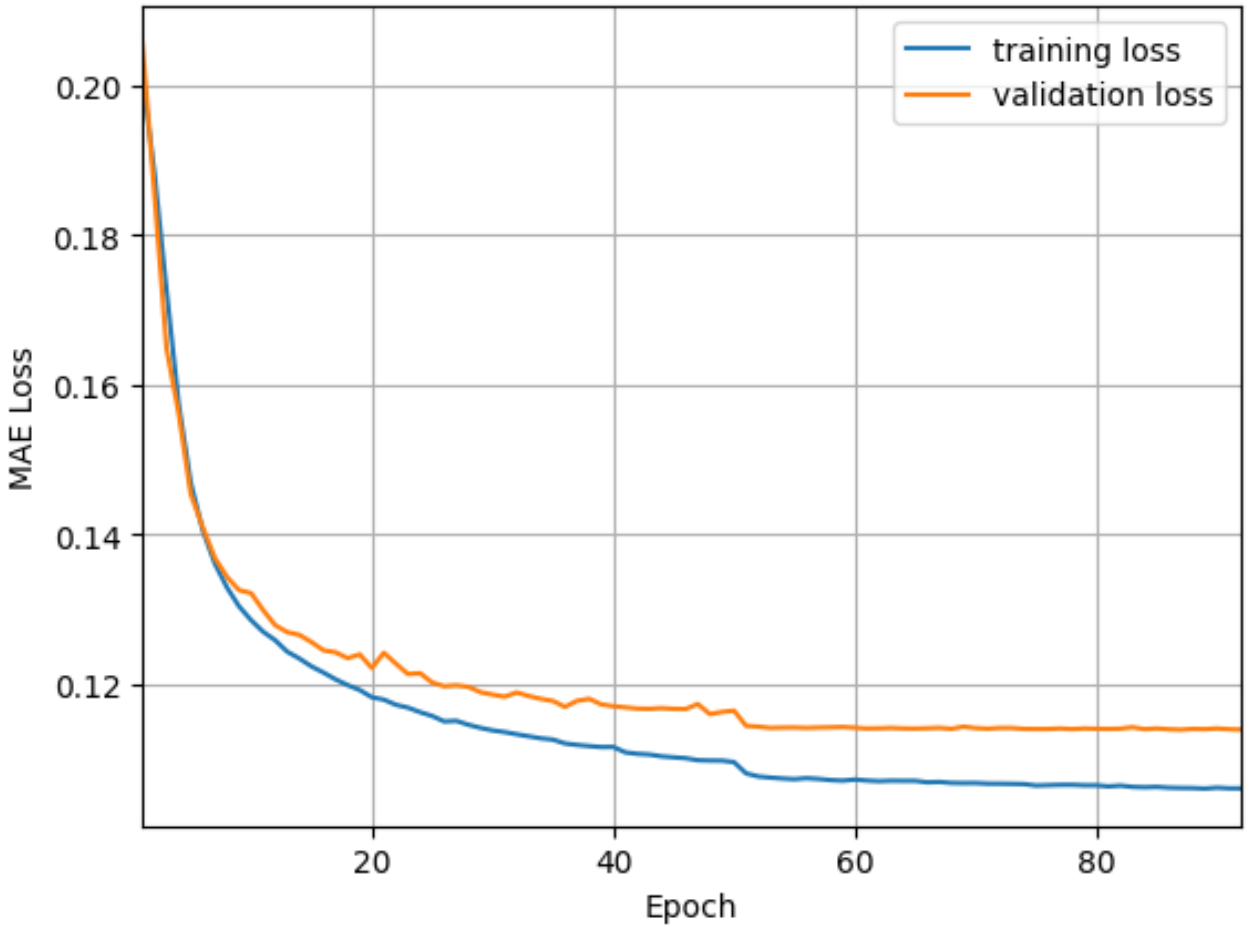
Scrandle.com image

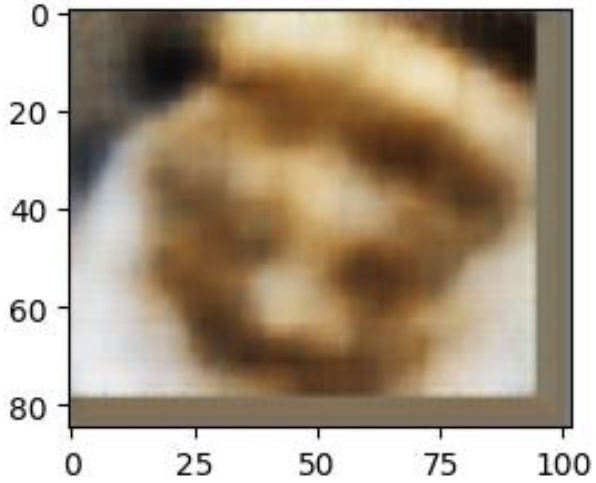
Scraped image

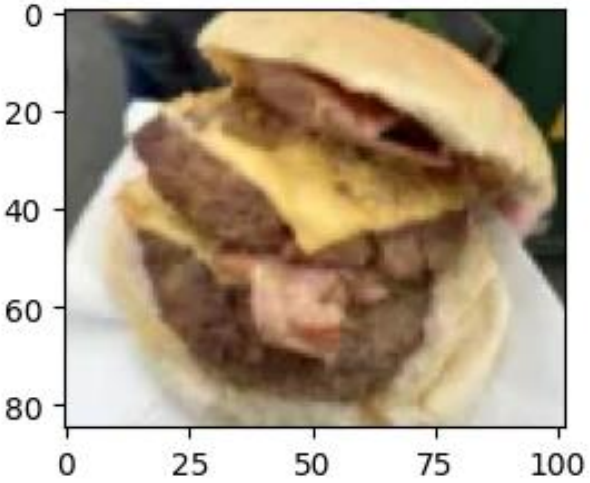
Padded to match largest possible image

Centred and set to scrandle.com aspect ratio

VAE Results



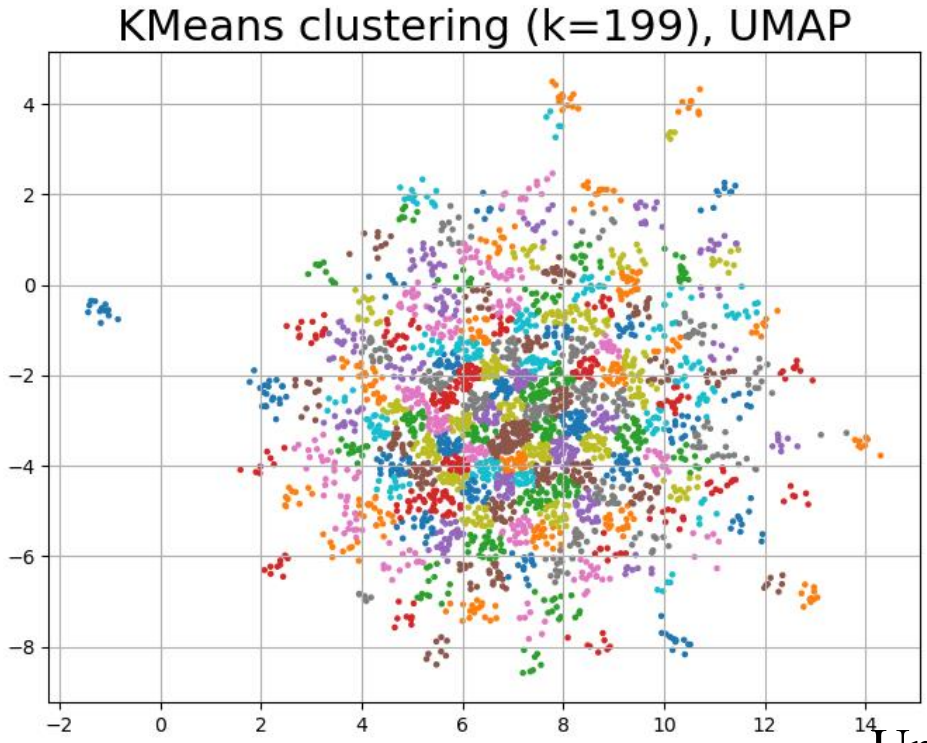


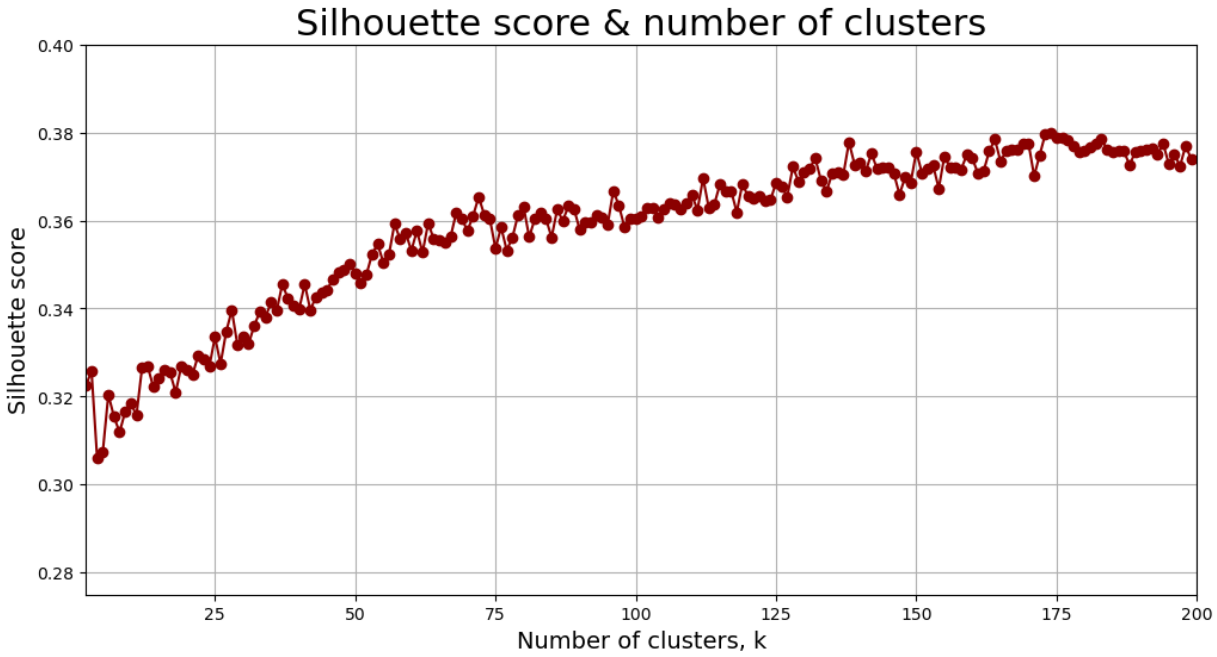


- Using batch size of 96 And 145 latent space dimensions.
- Better results, despite higher loss
- Still issues with green
- Still borders
- Solution: Use ResNet, redo decoder, HP optimization

Clustering And Dimensionality Reduction With Images and Tokens:

Modern Art on a Scatterplot





Unsatisfactory results:

No clear patterns with regards to price, food type, country, rating etc.

Not able to determine outliers from this result

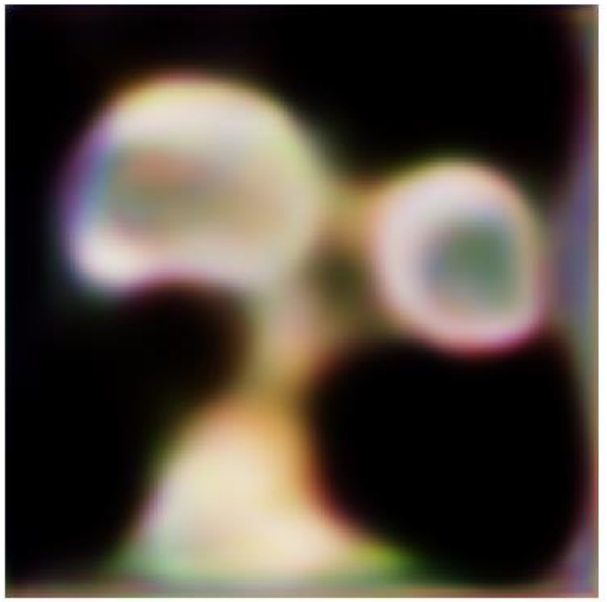
Generative adversarial network (GAN)

- Our aim was to try to generate our own pictures for Scrandle.
- Methods: using PyTorch, ResNet
- Problems:

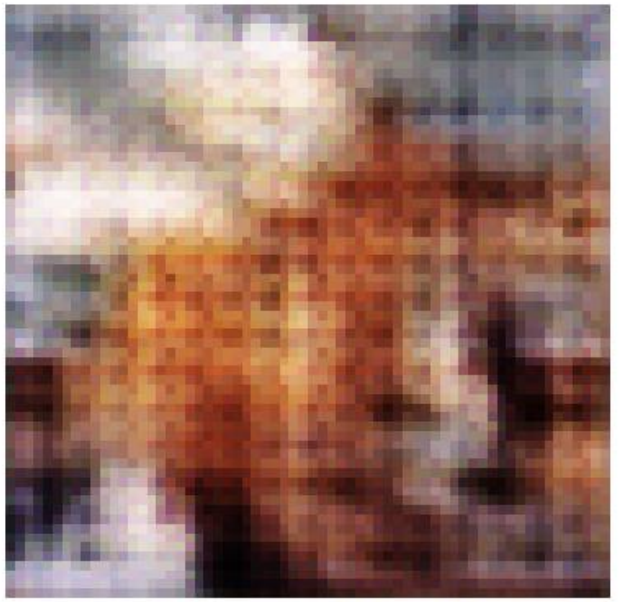
Took hours to run, even using GPU or Erda

ResNet improved pictures but increased runtime.

Increased resolution from 32x32 to 128x128 did not help clarity of images.

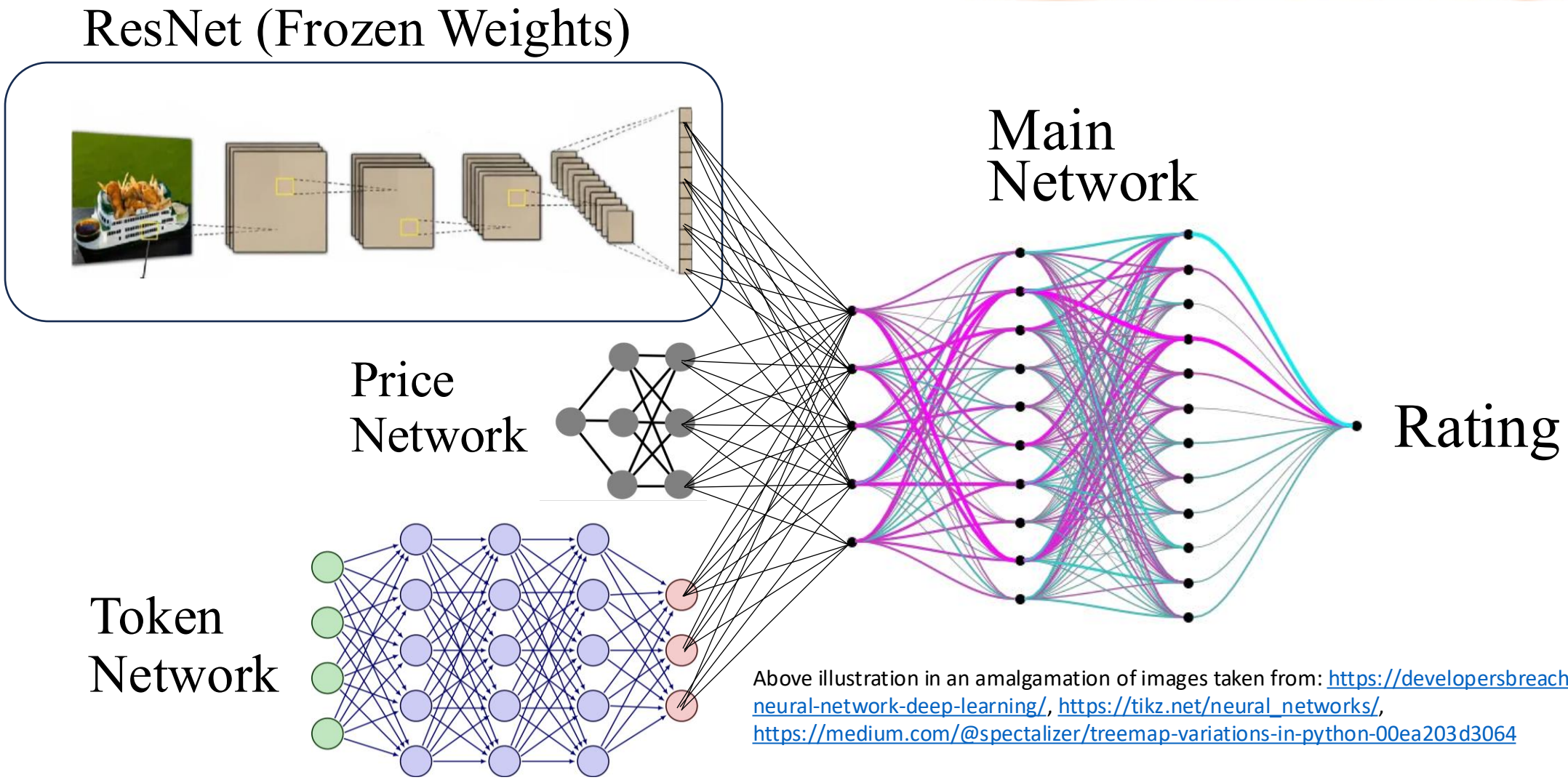


GAN "Burger" after 27 iterations using ResNet



GAN "Burger" after 120 iterations without ResNet

Structure of NN Model



Above illustration in an amalgamation of images taken from: https://developersbreach.com/convolution-neural-network-deep-learning/, https://tikz.net/neural_networks/, https://medium.com/@spectalizer/treemap-variations-in-python-00ea203d3064

Data and Training

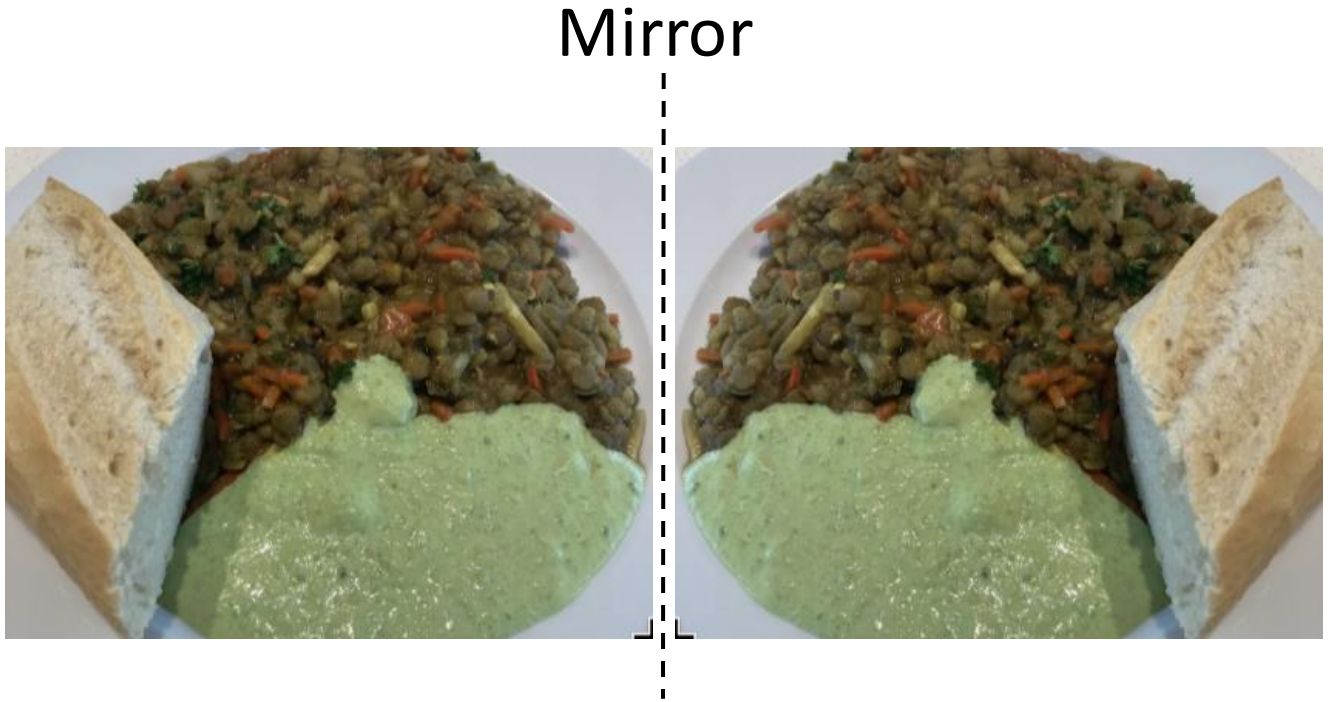
Mirroring used to double data entries

Samples increases: 3073 → 6146

Explicit Separation between Training/Test

L1Loss (MAE) Mean Absolute Error Loss

Optuna used to find number of layers and sizes of layers.



OPTUNA

Wow, the Optuna logo is from the Optuna website,what a coincidence!: https://optuna.readthedocs.io/en/stable/

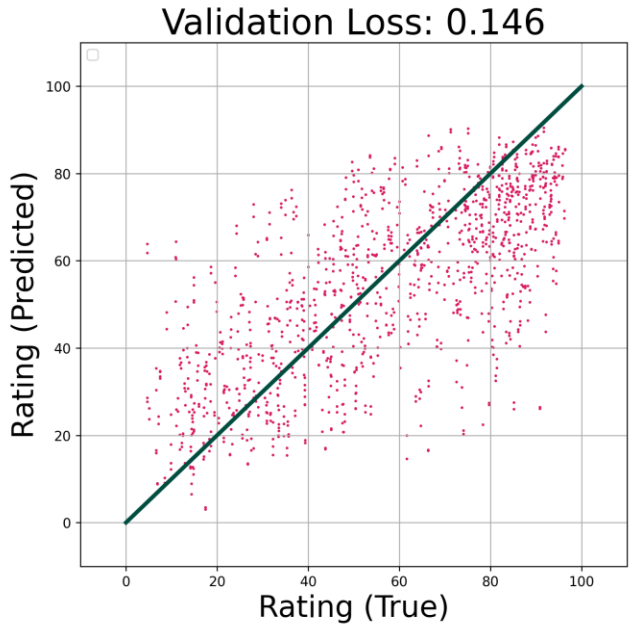
Training Results

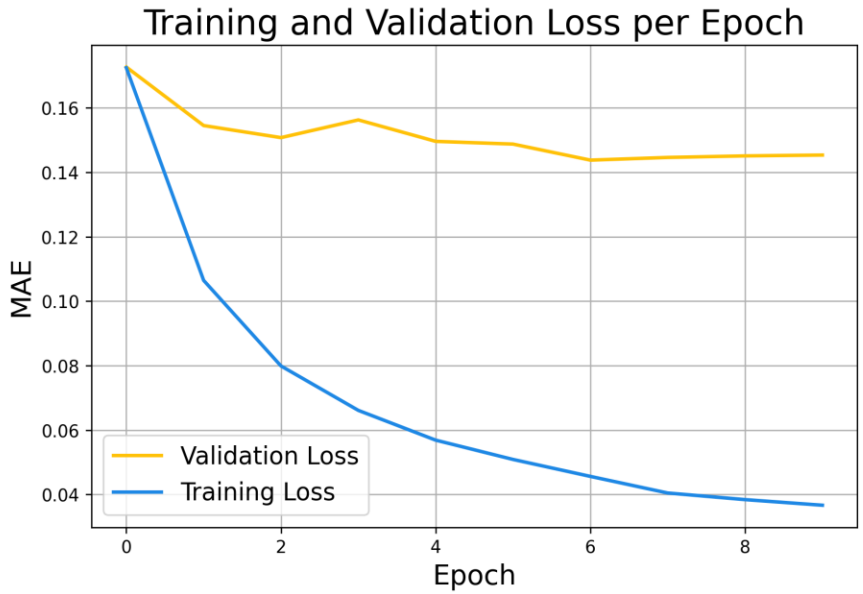
Clearly Overtrained, but this might be a benefit.

Training Loss reflect the ability to recall and already seen scrandle entry.

Validation Loss reflect the ability to score a 'never before seen' scrandle entry.

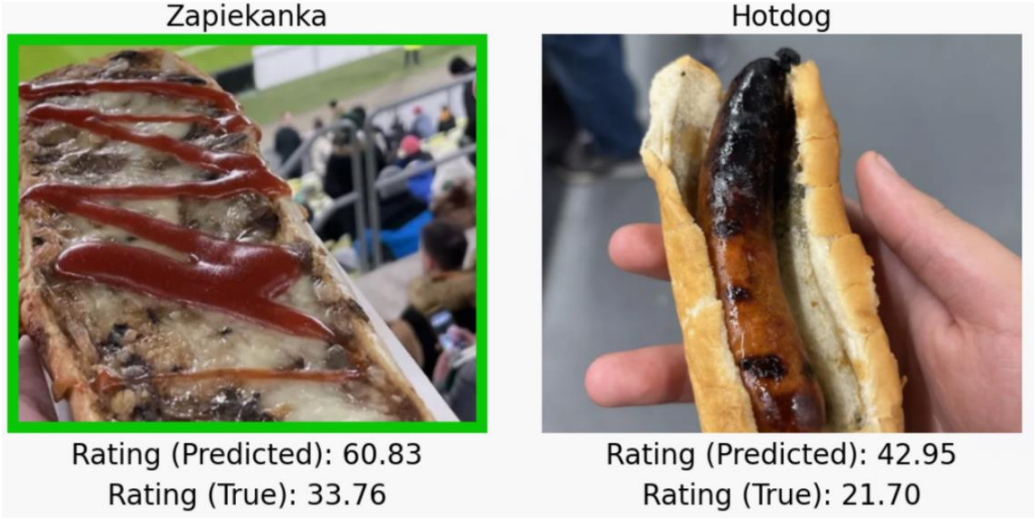
Training struggles to generalize the model. We might be data limited





Scrandle Bot

Load	Load data from Scrandle.com
Use	Use model to predict rating score
Pick	Pick the food item with the highest predicted rating



Metadata

Price: ”invaluable”

Title: “Troels C. Petersen”

Subtitle: ”Particle physicist”

Club: ”Copenhagen”

Country: ”DEN”

Troels C. Petersen

Rating (Predicted): 22.12

Homemade salmon with peas

Rating (Predicted): 48.00

Metadata

Price: 1.0£

Title: “Homemade salmon with peas”

Subtitle: ”Donated salmon (from mum)”

Club: ”Copenhagen”

Country: ”DEN”

Picture of Troels taken from:
https://www.nbi.dk/~petersen/Teaching/AppliedMachineLearning2026.html
Salmon and peas made by Christopher.

Logo source: Awesome logo was generated by ChatGPT

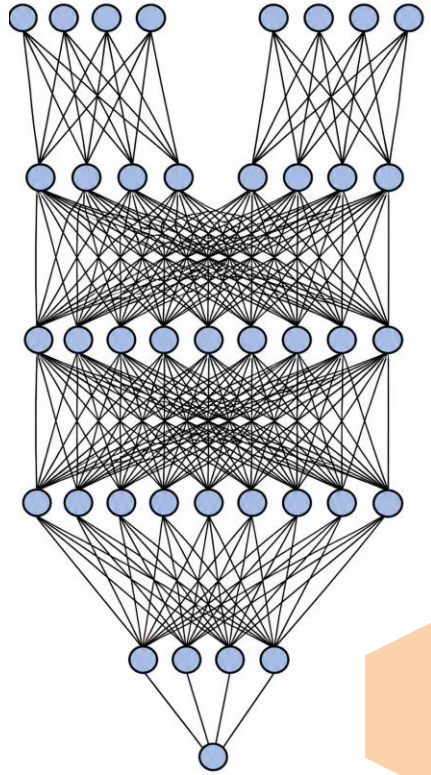
Alternative Approach

Input:
- Image
- Price
- Tokens

Direct Classification

Advantage:
Samples increases: 3073 → 9440256

Disadvantage:
Cannot remember ratings scores from already seen images



Classifier
Left/Right

Image of Neural Network Source: Generated by ChatGPT and edited in Pinta

Conclusion and Outlook

- Our model achieved an MAE validation loss of 0.146.
- Data is complicated, inconsistent and clearly from humans

- Future:
  - Perhaps train on matches of foods, not just individual foods themselves.
  - Clustering could be used as an outlier detector

Appendix

"Sausage" from Frankfurt, Germany

Scran From 09-06-2026

The Following 10 Slides are Backup if Scrandle on the 11-06-2026 is Primarily Jokes

Round 1

The hawks burger

Rating (Predicted): 69.22

Rating (True): 63.38

Brookie burger

Rating (Predicted): 36.14

Rating (True): 72.55

Round 2

Cheese burger

Rating (Predicted): 68.95

Rating (True): 59.01

Chicken teriyaki bowl

Rating (Predicted): 56.56

Rating (True): 74.29

Round 3

Sensational popcorn

Rating (Predicted): 34.57

Rating (True): 28.39

Roast chicken dinner

Rating (Predicted): 58.73

Rating (True): 54.12

Round 4

Grilled cheese with chicken

Rating (Predicted): 48.50

Rating (True): 69.97

Helmet nachos

Rating (Predicted): 39.06

Rating (True): 41.39

Round 5

Chicken shawarma

Rating (Predicted): 64.60

Rating (True): 31.94

Steak, potato and Guinness pie

Rating (Predicted): 45.84

Rating (True): 22.12

Round 6

Lamb meat and chips box

Rating (Predicted): 64.88

Rating (True): 63.68

Turkey, pork and stuffing pie

Rating (Predicted): 63.53

Rating (True): 19.89

Round 7

Indian box

Rating (Predicted): 61.61

Rating (True): 55.66

Pie

Rating (Predicted): 23.41

Rating (True): 19.78

Round 8

Chilli cheese fries

Rating (Predicted): 62.23

Rating (True): 38.02

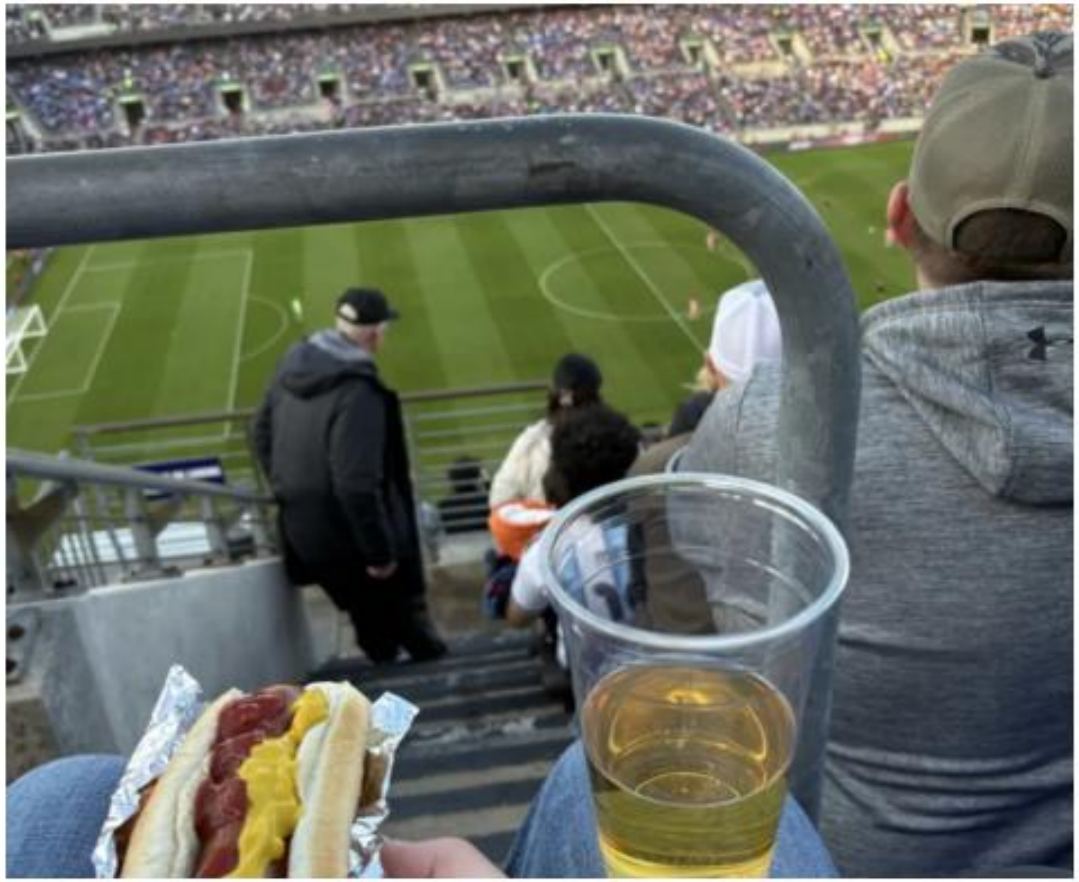
Double bacon n cheeseburger

Rating (Predicted): 58.47

Rating (True): 58.25

Round 9

Hot dog and beer



Rating (Predicted): 32.18

Rating (True): 32.87

Ahi tuna hawaiian bowl

Rating (Predicted): 33.35

Rating (True): 58.14

Round 10

Cheesy chips

Rating (Predicted): 20.49

Rating (True): 19.89

Pie, chips and gravy

Rating (Predicted): 67.26

Rating (True): 47.19

Code available at our Github page

https://github.com/JakobFugl/scrandle_ml_project.git

Project Statement

All participants contributed equally

Comments on GAN

• GAN using ResNet:

- Used image resolution of 128 * 128.
- Batch size of B = 32
- Latent noise dimension of 128

• Generator:

Input noise vector of dimensions (B, 128) is passed through a connected layer and reshaped to
(B, 256, 8, 8). Therefrom 4 upsampling stages are applied, each of upsampling with scale factor 2. Therefrom followed by a convolution with kernel size 3*3, padding 1, stride 1, batch normalization and ReLU.

- After each convolution, a residual block is applied consisting of 2 convolutional layers with kernel size 3*3 and padding 1.
- Channel progression through the network is: 256 →128 → 64 → 32 → 3, Spatial dimensions increase as: 8*8 → 16*16 → 32*32 → 64*64 → 128*128
- Final output of: **(B, 3, 128, 128).**

• Discriminator:

- Input image of dimensions **(B, 3, 128, 128)** is normalized using ImageNet statistics.
- Image is then passed through a **ResNet18.**
- Final fully connected layer is replaced to map: (B, 512) to **(B, 1)** producing a single logit output.
- The discriminator outputs a score indicating whether input is real or generated. It is good at discerning what is real or generated, but horrible at generating images.

Comments on GAN, without ResNet

- GAN without ResNet:

Used image resolution of **128 * 128**.

Batch size of **B = 32**

Latent noise dimension of **128**

- Generator:

Input noise vector of dimensions **(B, 128)** is passed through a connected layer and reshaped to (B, 256, 8, 8).

Therefrom 4 upsampling stages are applied, each upsampling with scale factor 2. Therefrom followed by a convolution with kernel size 3*3, padding 1, stride 1 batch normalization and ReLU.

After each convolution, a block of 2 convolutional layers with kernel size 3*3 and padding 1 is applied.

Channel progression through the network is: 256 to 128 to 64 to 32 to 3. The spatial increases with 88 to 1616 to 3232 to 6464 to 128*128 to a final output of **(B, 3, 128, 128)**

- Discriminator:

Input image of dimensions **(B, 3, 128, 128)** is passed through a series of convolutional layers.

Each layer consists of, Convolution with kernel size 3*3, padding 1, stride 2, increasing number of kernels, as with the Generator.

Final feature map is flattened and goes via a layer mapping to an output **(B, 1)**

The discriminator outputs a "score" indicating whether input is real or generated, as with the ResNet version.

Comments on data

- All matches on Scrandle.com are unique. Some individual foods may have several matches and show up multiple times, with different rating scores.

- Many Fridays contain "jokers": military food, rats, birds, mushrooms, Halloween paintings etc.
- Human factors may influence the data: perhaps Koreans are more willing to show their nice foods, whereas Hungarians want to show their weird and funny food.

Tokenization notes

- Almost all entries in the CountVectorizer matrix are 0 or 1. Very few are 2 or 3. This is because the same word very rarely appears twice or more in the same image text
- Since most entries are 0 in the matrix, sparse matrix representation is used.
- Dimensionality reduction did separate e.g. mushrooms and cheeseburgers, but mushrooms were not further removed from cheeseburgers than burgers from cheeseburgers. Therefore it did not separate outliers (mushrooms) from non-outliers as hoped.
- NaNs were replaced with a blank space before tokenization

More Details for the Different VAE Models

NOTE: All VAE models use PyTorch. All VAE models, except ResNet, optimized with optuna

VAE with padded images:

- Used 105 by 85 resolution images
- Batch size of 36, 91 latent space dimensions
- 2500 of the images are training. Rest are validation. Not shuffled.
- Encoder convolutional layers description: 3 layers. First has 8 kernels of spatial extent 3, padding of 0, and stride of 2.

Second has 16 kernels of spatial extent 3, padding of 0, and stride of 2. Third has 32 kernels of spatial extent 3, padding
Of 0, and stride of 2.

- FF(Feed Forward) part of the encoder network consists of flattening convolutional output to 1D input layer of 11*9*32 inputs, then going to

A layer with 128 nodes, and finally to the latent space dimension.

- Decoder: Unflattens image to size (32,11,9) first.

Otherwise: opposite structure to Encoder using ConvTranspose2d. Image size issues “fixed” using output padding of (1,1)
For first layer, (0,1) for second, and (1,0) for the last one. Output padding causes weird borders on all reconstructions.

- ReLU used as activation layers, sigmoid as final layer. Weight decay of 1e-05.

VAE with correct size images(centred and set to scrandle.com aspect ratio):

- Used 105 by 85 resolution images
- Batch size of 36, 91 latent space dimensions
- Training set is 80% of data, validation set is 20% of data. Shuffled.
- Differences from padded image version: Decoder has swapped 11 and 9 in unflattener as aspect ratio changed.(flattened dimension is the same)
- FF(Feed Forward) part of the network is, the same as for the one with padded images.
- Learning rate was 0.001 for first 50 epochs, 0.0001 for remaining 42.
- ReLU used as activation layers, sigmoid as final layer. Weight decay of 1e-05.

More Details for the Different VAE Models

VAE using correct size images and ResNet:

- Used 105 by 85 resolution images.
- Batch size of B=36
- Latent dimensions of 91
- 3193 images, 80% are training, rest are validation.
- Weight decay of 1e-5

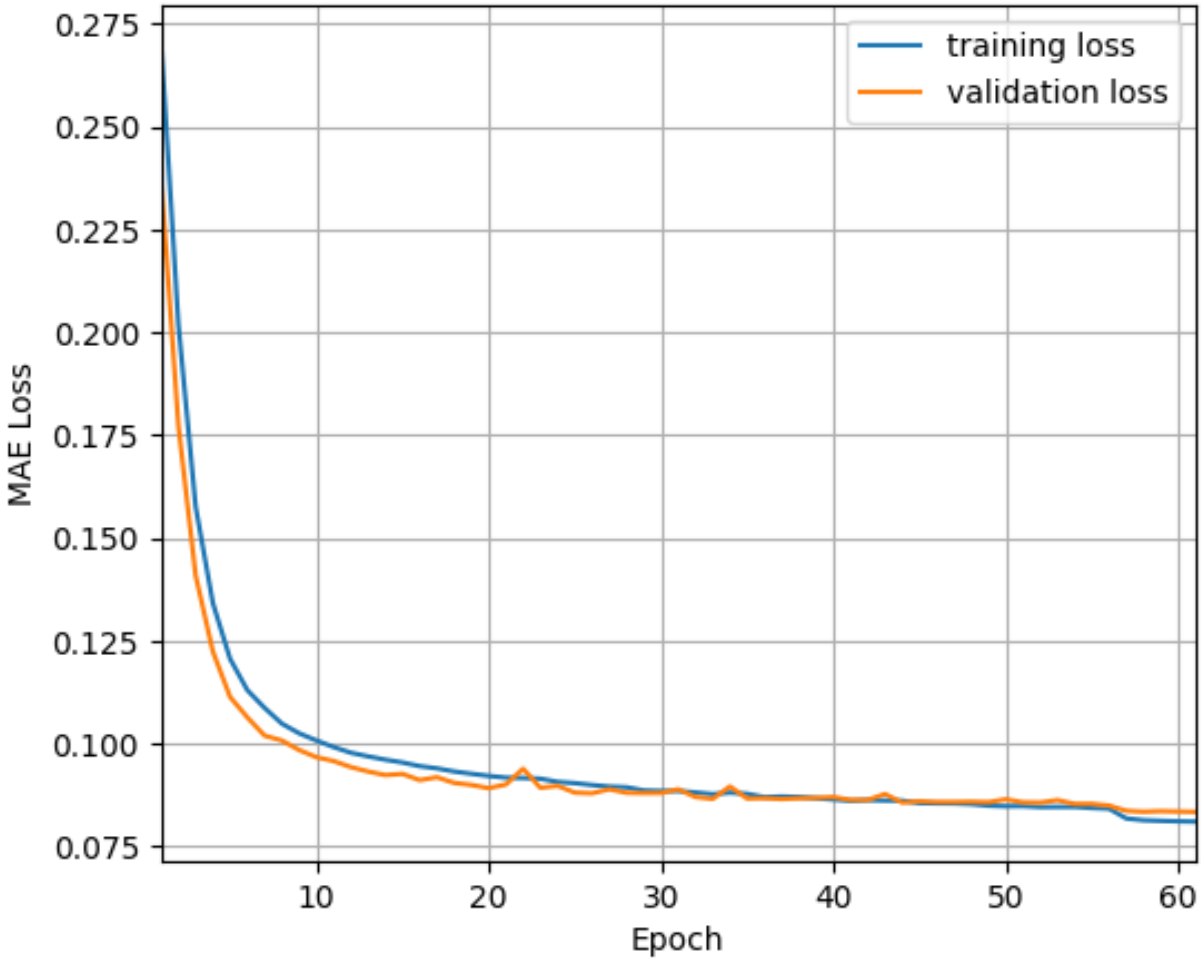
- Encoder:

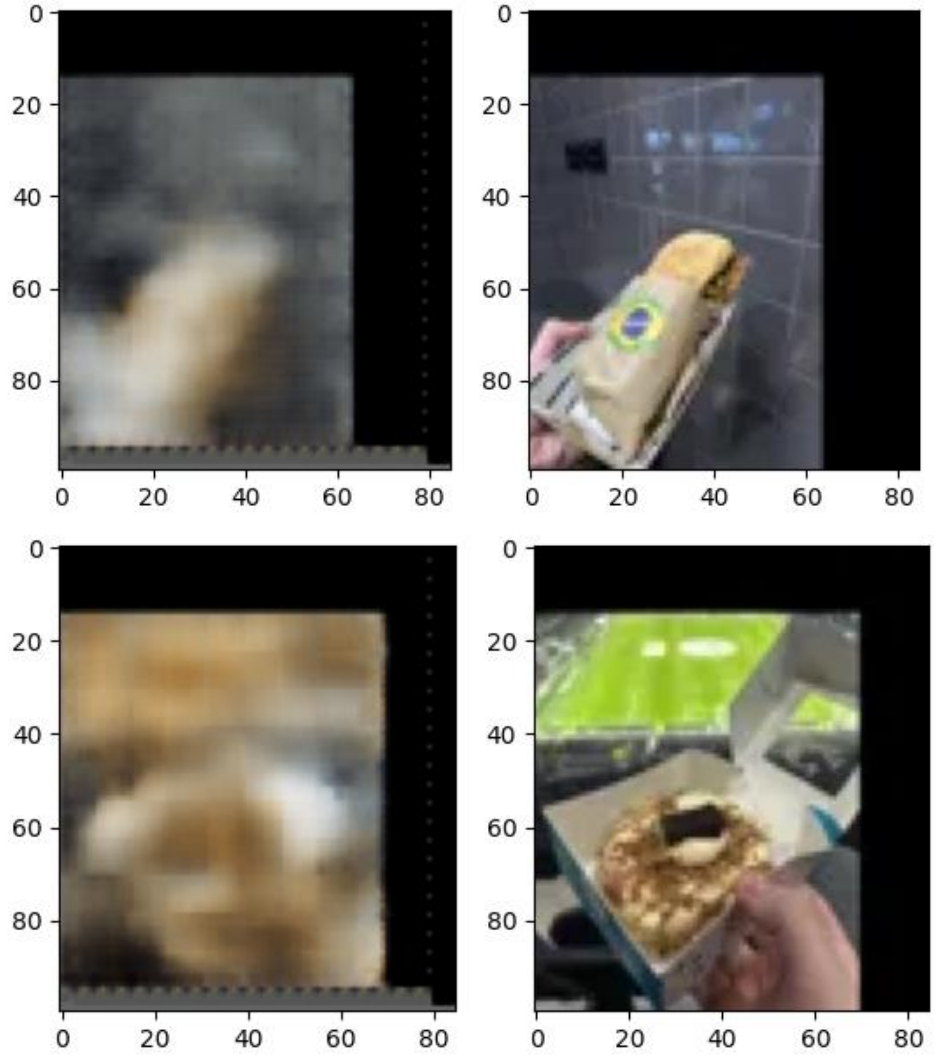
Input image of dimensions **(B, 3, 105, 85)** is passed through ResNet50 with weights from "IMAGENET1K_V2" and flattened to (B, 2048) and therefrom compressed (B, 2048) to (B, 128) to a latent vector of **(B, 91)**.

- Decoder:

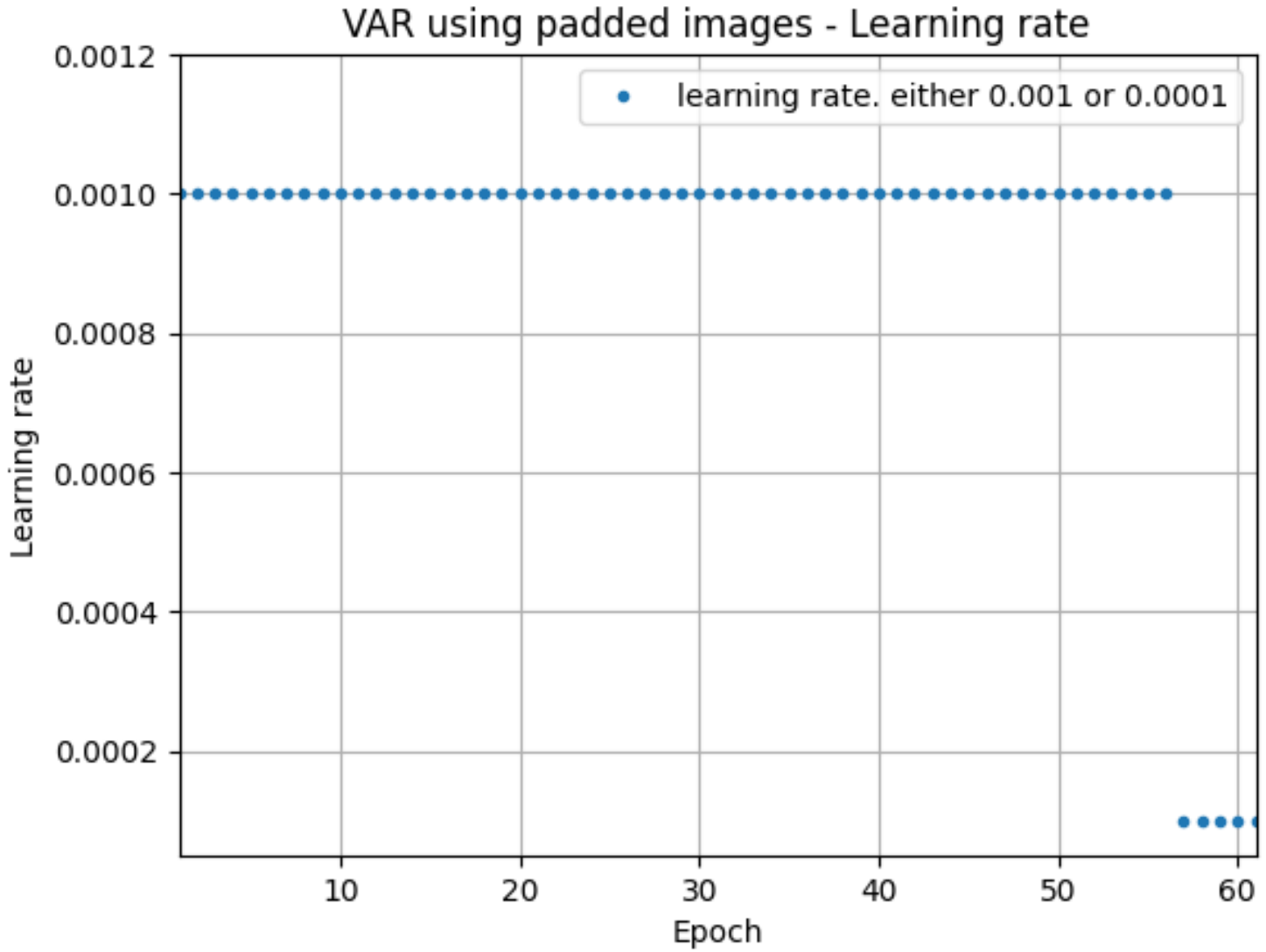
Input latent vector of **(B, 91)** is upsampled to (B, 128) to (B, 256*5*6) and "unflattened" to (B, 256, 5, 6).
Therefrom 3 layers of convolution to (B, 32, 48, 40) all with a stride of 2, padding of 1 and kernel size of 4*4; kernels being {128, 64, 32}.
Thereafter a final layer of stride 1, padding 0, 3 kernels and a kernel size of 38*63 to match the original dimensions to **(B, 3, 85, 105)**.
Activation layers were ReLU

VAE results with padded images:

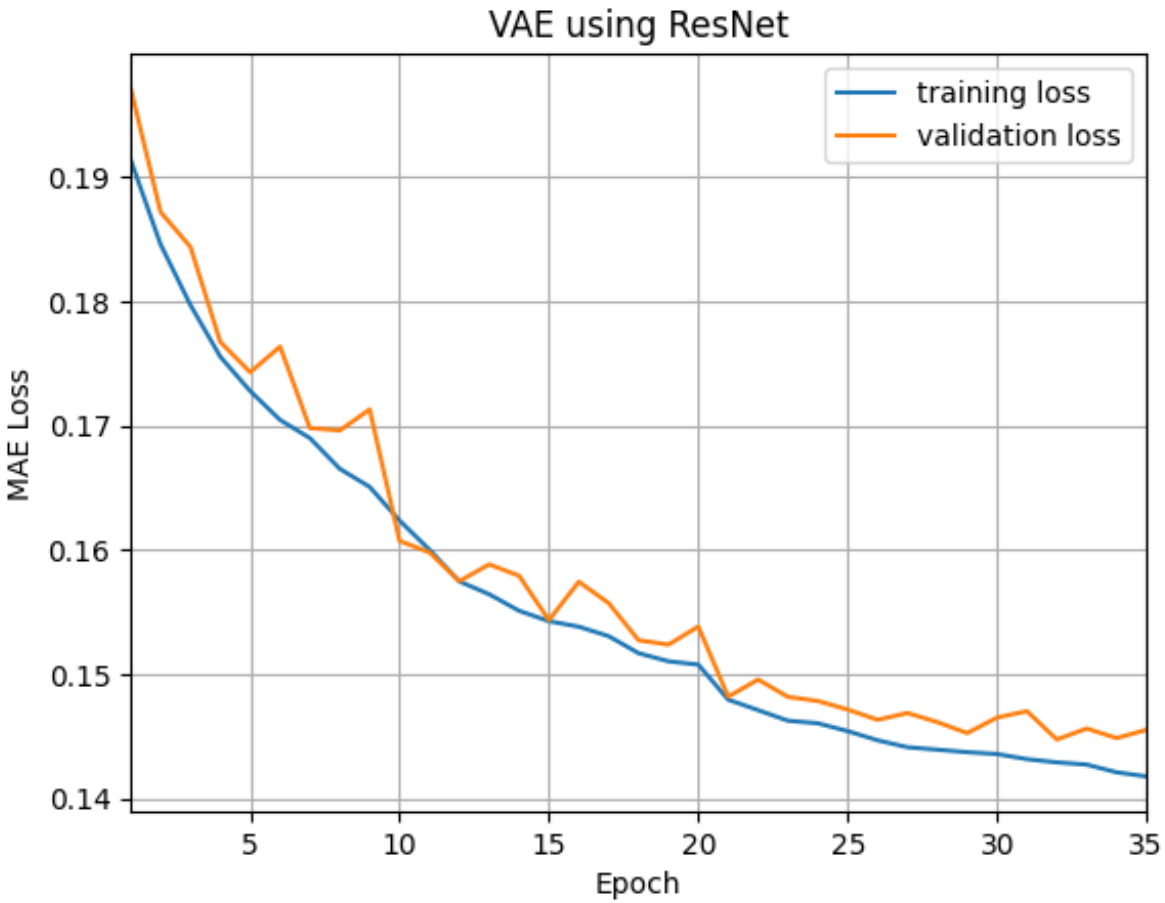


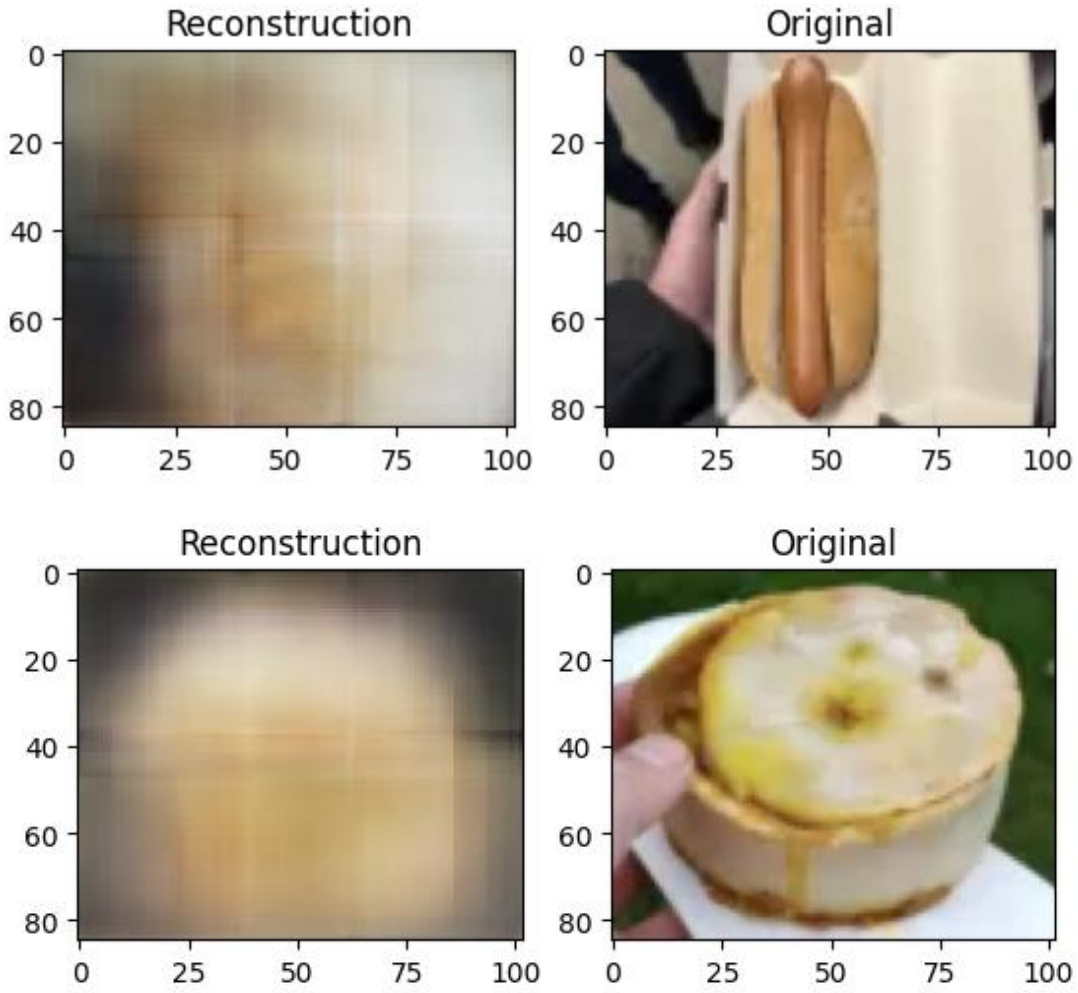


VAE results with padded images:

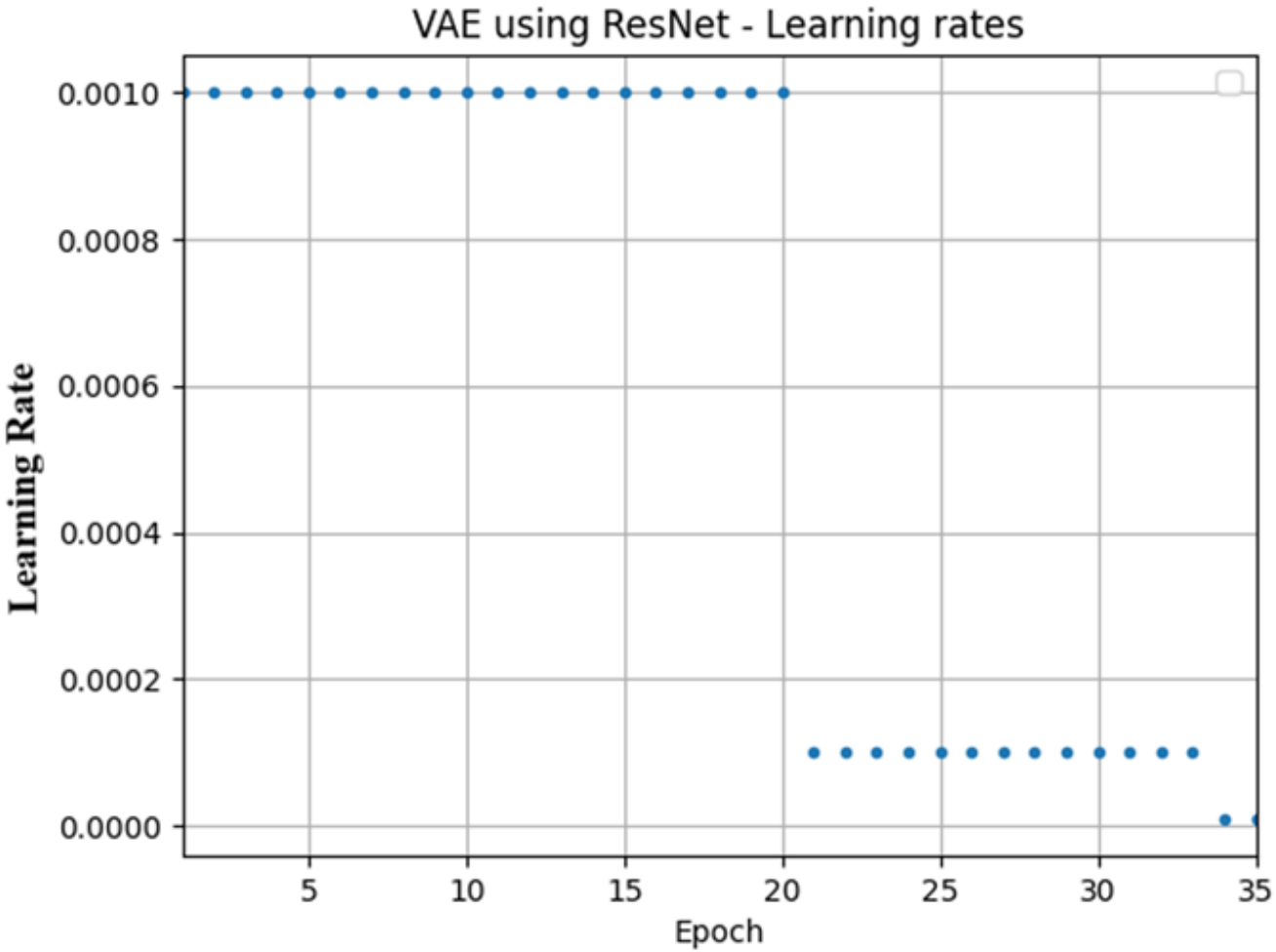


VAE results with ResNet Model:





VAE results with ResNet model:



Regression Model for Rating:

Regression Model:

- Used 105 by 85 resolution images
- Used PyTorch
- Using ReLU between each layer.
- Used ResNet50 (Weight are frozen)
- Sigmoid on the last output layer on Main/Head Neural Network
- (Using mirroring) number samples are: (6146) 3073
- ResNet50, Price Neural Network and Token Neural Network all feeds into the Main/Head Neural Network
- Samples in training dataset: (4916) 2458
- Samples in valdiation dataset: (1230) 615

- Optuna Optimized the following:
- Batch Size: 80
- Learning Rate: 5.109e-4
- Weight Decay: 2.0378e-7
- Dropout Rate: 0.2814
- Number of Layers in Price Neural Network: 2 (Excluding Input Layer) | And their sizes: {16,8}
- Number of Layers in Token Neural Network: 2 (Excluding Input Layer) | And their sizes: {1024,256}
- Number of Layers in Main/Head Neural Network: 2 (Excluding Input/Output Layer) | And their size: {512, 128}

Scrandle developer

- We tried to contact the developer of Scrandle, "danbo". They asked what we wanted and ended their e-mail by:

"*An egg is always an adventure; the next one may be different.*"

- Oscar Wilde

- We responded and they ghosted us; perhaps they were bewildered and intimidated by our response, which ended with:

"*Yesterday is history, tomorrow is a mystery, but today is a gift. That is why it is called the present.*"

- Master Oogway

Oogway from Kung Fu Panda, Dreamworks 2008, 31:09
Oscar Wilde from https://www.britannica.com/biography/Oscar-Wilde

Clustering and Dimensionality Reduction details

- Token only dimensionality reduction done using UMAP with input: n_components=2, n_neighbors=50, random_state=42

For Combined images and tokens:

- Combined the tokenization model and the 3193 unique images. The tokens had a weight of 1, the images a weight of 0.01.
- UMAP used with a minimum distance of 0.9 and number of neighbors of 30.
- The clusters found using KMeans, Gaussian Mixture and Spectral from sklearn, did not make sense to us; there was no consistency in either price, country of origin, type of food etc. For KMeans, clustering plateaued at a silhouette score of 0.38.
- Clustering could **not** be used to find outliers and was deemed unsuccessful with the amount of data we had and our methodology.