Hello!
My name is Seung Woo Choi,
and I am a Data Scientist.
You can find me at:
linkedin.com/in/seung-woo-choi/

Multiclass
Food Image
Classification
with CNNs

My Vision

Potential Applications

Recipe Recommender System

The user takes a picture of ANY food with his/her phone, and the model accurately predicts the food class and provides a generic recipe for the food class.

Uses: traveling in a foreign country, trying a new cuisine

Object Classification

The user takes a picture of ANY object with his/her phone, and the model accurately identifies the object.

Use: “eyes” for the visually impaired

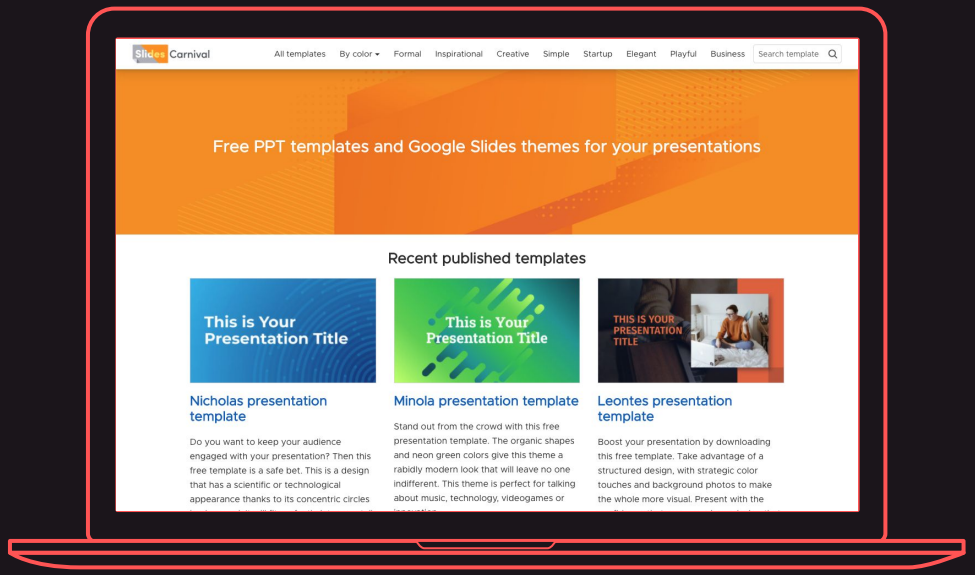
Dinner Recommender System

The user takes a picture of his/her dinner, and the model accurately predicts the food class and recommends a similar/different food class for tomorrow’s dinner.

Use: deciding what to eat for dinner tomorrow

A Reality Check...

Silicon Valley: Not Hotdog (Season 4, Episode 4 Clip)



Problem Statement

- Collect image data for 5 food classes:
  - Burger
  - Hotdog
  - Pizza
  - Taco
  - Sushi
- Build CNNs from scratch and pre-trained, SOTA models to predict multiclass food classes

About the Data

- **Data source:** Reddit's pushshift.io API
- **Collection method:** scraping urls
- **Image data:** .jpg and .png files
- **Ideal sample size:** 1,000 images/class
- **Actual sample size:** 253-282/class
- **Retention rate:** scraped 600/class → used < 300/class → **42-47% retention**
- **Efficiency: 26.7%** of ideal sample size
- **Data cleaning:** removed manually

Limitations

Data Collection

Google Images vs. Reddit

Sample Size

Ideal vs. restricted due to manual removal of incorrect images

Low Budget

AWS SageMaker

Processing Speed

Test runs for modeling

Hyperparameter tuning

Exploratory Data Analysis

Class Membership

Example: Burger and Hotdog

Example: Pizza and Taco

Example: Sushi

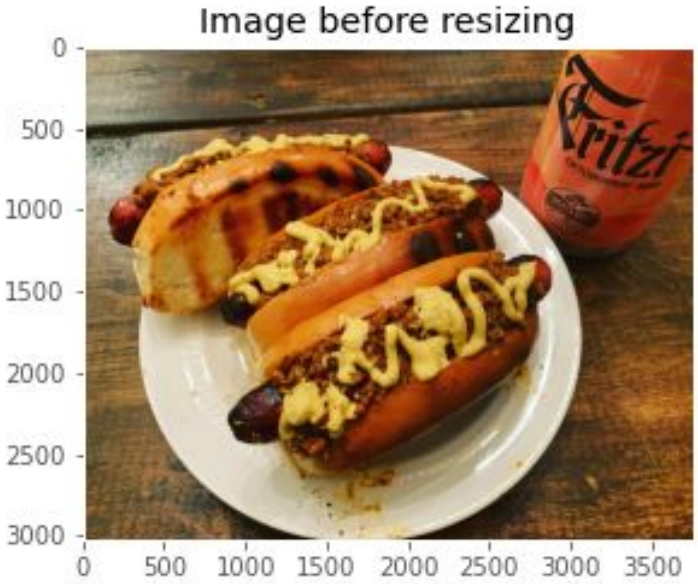
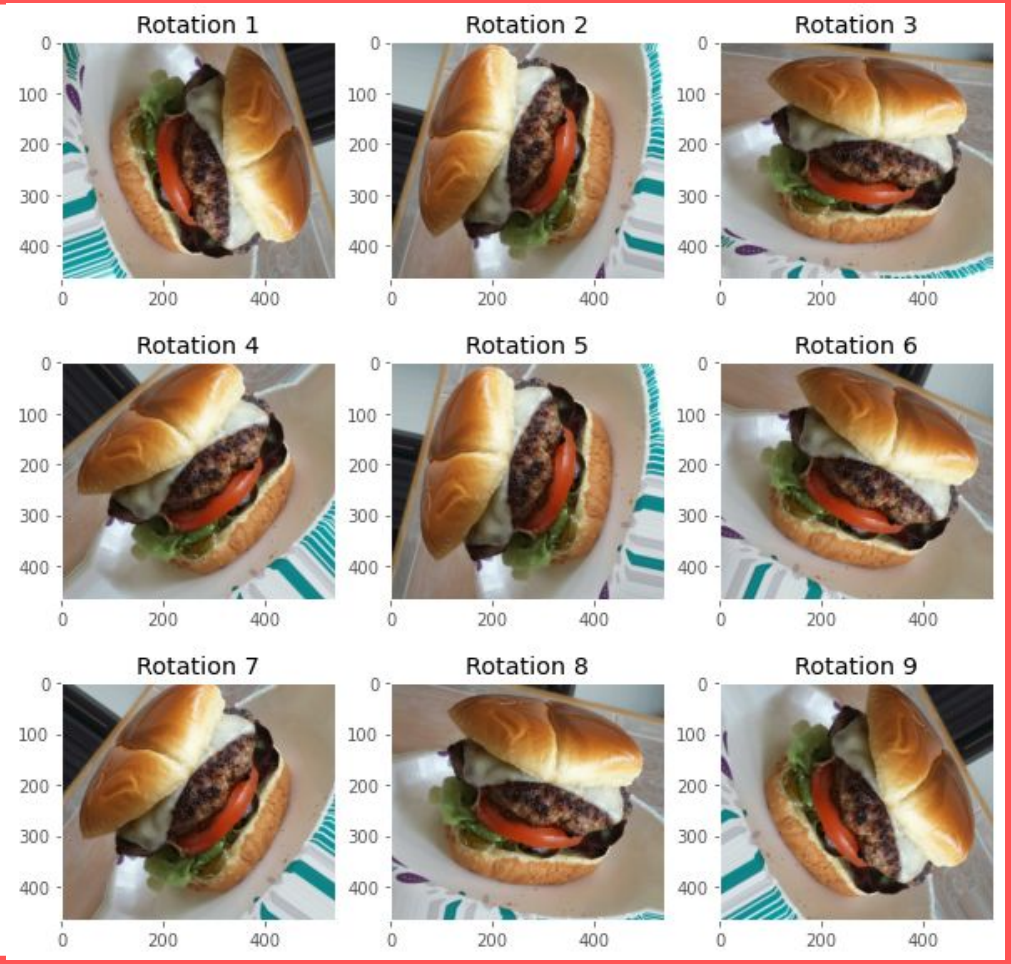


Image Before and After Resizing

Data Augmentation: Random Rotation

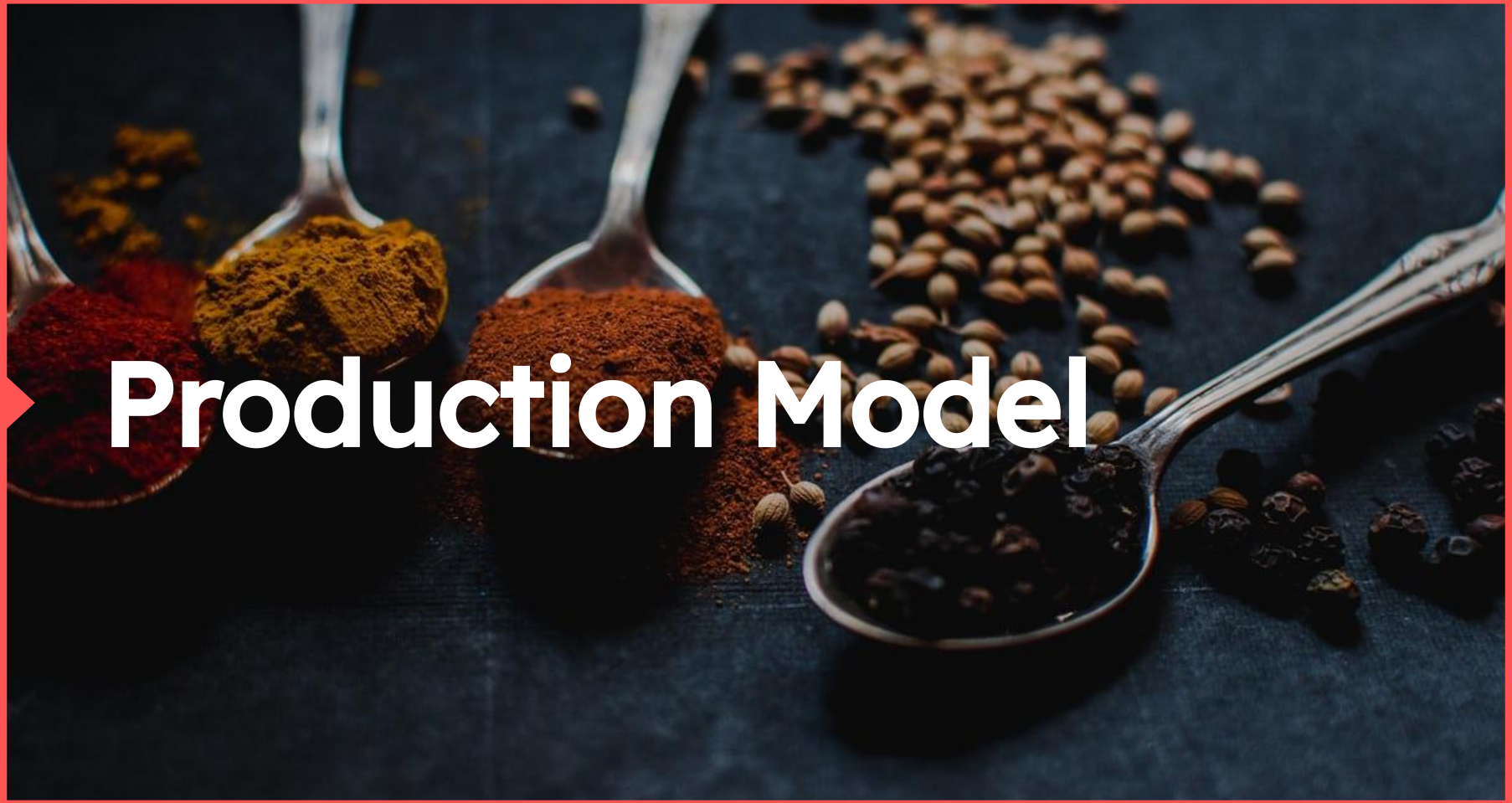


Modeling

Models

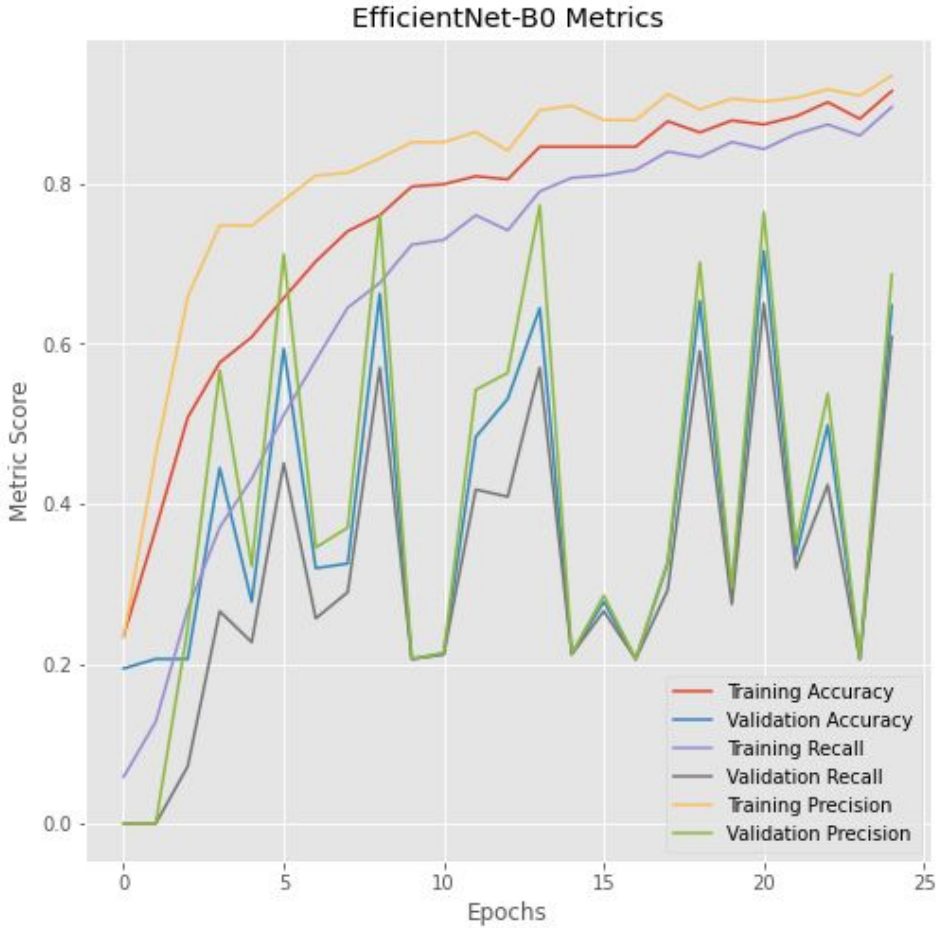
Null Model
Majority Class: **Burgers**
Baseline Accuracy: **21.06%**

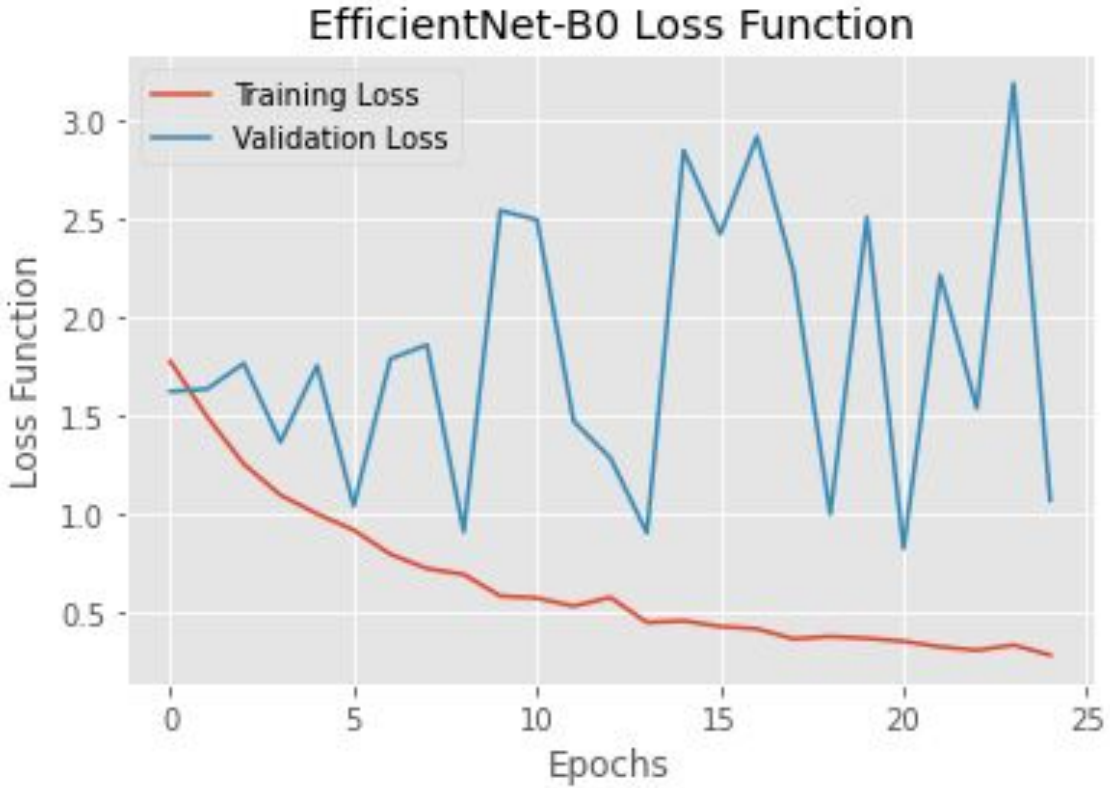
Model Type	Validation Accuracy	Vs. Null Model	Validation Loss
CNN with 9 layers	45.37%	2.15x	1.57
CNN with 15 layers	33.13%	1.57x	1.85
ResNet50 (IBM)	28.66%	1.36x	1.57
EfficientNet-B0 (Google)	64.78%	3.07x	1.06



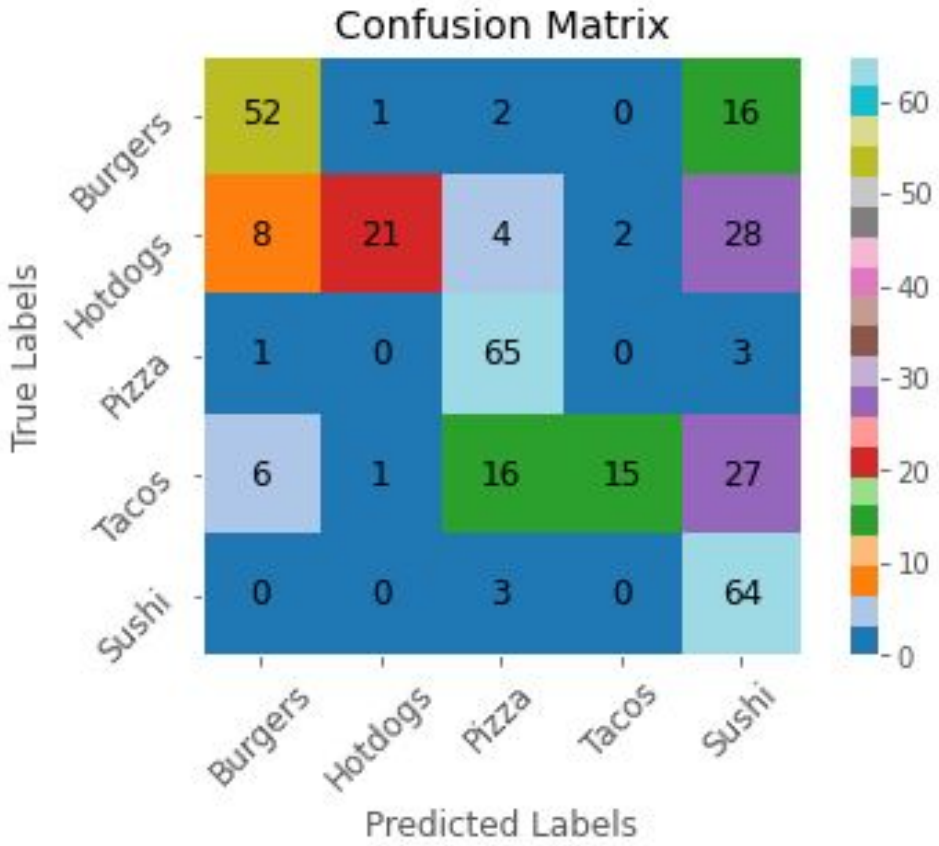
Production Model

EfficientNet-B0: Classification Metrics





EfficientNet-B0: Loss Function



EfficientNet-B0: Confusion Matrix

Conclusion

5

Number of classes

253 to 282

Number of images per class

64.78%

Accuracy on unseen data

Thanks!

Any questions?