

LAPORAN UAS DEEP LEARNING
IDENTIFIKASI MAKANAN SEHAT MAHASISWA TEKNIK UNIVERSITAS
BENGKULU



Disusun Oleh :

- | | |
|--------------------------------------|--------------------|
| 1. Nanditha Nabiilah Putri | (G1A021001) |
| 2. Erin Handayani Azzahra | (G1A021047) |
| 3. Farhani Ilham Hidayatullah | (G1A021081) |

Dosen Mata Kuliah:

Arie Vatresia, S.T., M.T.I., Ph.D

PROGRAM STUDI INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS BENGKULU
2024

1. Pendahuluan

Mahasiswa sering kali menghadapi tantangan dalam menjaga pola makan sehat di tengah jadwal akademik yang padat. Hal ini juga dirasakan oleh mahasiswa Teknik Universitas Bengkulu, yang cenderung memilih makanan cepat saji atau makanan yang kurang sehat karena keterbatasan waktu dan pilihan. Padahal, asupan makanan sehat sangat penting untuk mendukung kesehatan fisik dan mental, serta menjaga performa akademik mereka. Dalam konteks ini, teknologi berbasis *deep learning* dapat menjadi solusi inovatif untuk membantu mahasiswa mengenali dan memilih makanan yang sehat. Penelitian ini menggunakan model MobileNetV2, sebuah arsitektur *deep learning* yang ringan dan efisien, untuk mengidentifikasi makanan sehat yang sering dikonsumsi oleh mahasiswa Teknik Universitas Bengkulu. Dengan sistem ini, diharapkan mahasiswa dapat lebih mudah memahami pola makan sehat dan meningkatkan kualitas hidup mereka.

2. Analisa Model

Analisis model dalam penelitian ini berfokus pada penerapan arsitektur MobileNetV2, sebuah *convolutional neural network* (CNN) yang dirancang untuk tugas klasifikasi gambar dengan efisiensi tinggi. MobileNetV2 menggunakan *depthwise separable convolution* untuk mengurangi jumlah parameter dan mempercepat proses komputasi tanpa mengorbankan akurasi. Selain itu, *inverted residual blocks* yang menjadi ciri khas model ini memungkinkan efisiensi propagasi informasi melalui jaringan dengan mempertahankan fitur yang relevan dan menghilangkan fitur yang tidak signifikan. Model ini memanfaatkan *transfer learning* dari model pra-latih pada dataset besar seperti ImageNet, kemudian disesuaikan dengan dataset lokal yang mencakup makanan sehat khas mahasiswa Teknik Universitas Bengkulu. Dalam penelitian ini, model dilatih selama 25.000 epoch untuk memastikan proses optimasi parameter berlangsung secara mendalam, sehingga model mampu mengenali pola kompleks dari citra makanan. MobileNetV2 dievaluasi menggunakan metrik seperti akurasi, *precision*, *recall*, dan *F1-score*, yang menunjukkan bahwa model ini tidak hanya efisien tetapi juga andal dalam mengidentifikasi makanan sehat dengan tingkat akurasi yang tinggi.

3. Penjelasan Kode

3.1 Pengumpulan Data

Pengumpulan data untuk penelitian ini dilakukan dengan menggabungkan dua sumber utama, yaitu dataset makanan yang tersedia di platform Kaggle dan foto makanan

yang diambil secara mandiri. Dataset dari Kaggle menyediakan berbagai gambar makanan yang sudah dilabeli, mencakup berbagai kategori sehat. Data ini digunakan untuk memberikan dasar yang kuat dalam melatih model untuk mengenali berbagai jenis makanan. Selain itu, foto makanan diambil secara langsung oleh peneliti dengan menggunakan kamera secara langsung, yang mencakup makanan yang sering dikonsumsi oleh mahasiswa Teknik Universitas Bengkulu. Foto ini diambil dalam berbagai kondisi pencahayaan dan sudut pandang untuk mencerminkan variasi yang mungkin terjadi dalam dunia nyata. Setelah itu, kedua sumber data ini diproses dan digabungkan menjadi satu dataset terstruktur yang mencakup citra makanan, dan label jenis makanan. Dataset yang dihasilkan digunakan untuk melatih model MobileNetV2 agar dapat mengidentifikasi makanan sehat dengan akurasi tinggi.

3.2 Preprocessing Data

```
# Install the Object Detection API (NOTE: This block takes about 10 minutes to finish executing)

# Need to do a temporary fix with PyYAML because Colab isn't able to install PyYAML v5.4.1
!pip install pyyaml==5.3
!pip install /content/models/research/

# Need to downgrade to TF v2.8.0 due to Colab compatibility bug with TF v2.10 (as of 10/03/22)
!pip install tensorflow==2.8.0

# Install CUDA version 11.0 (to maintain compatibility with TF v2.8.0)
!pip install tensorflow-io==0.23.1
!wget https://developer.download.nvidia.com/compute/cuda/repos/ubuntu1804/x86_64/cuda-ubuntu1804.pin
!mv cuda-ubuntu1804.pin /etc/apt/preferences.d/cuda-repository-pin-600
!wget http://developer.download.nvidia.com/compute/cuda/11.0.2/local_installers/cuda-repo-ubuntu1804-11-0-local_11.0.2-450.51.05-1_amd64.deb
!dpkg -i cuda-repo-ubuntu1804-11-0-local_11.0.2-450.51.05-1_amd64.deb
!apt-key add /var/cuda-repo-ubuntu1804-11-0-local/7f2af80.pub
!apt-get update && sudo apt-get install cuda-toolkit-11-0
!export LD_LIBRARY_PATH=/usr/local/cuda-11.0/lib64:$LD_LIBRARY_PATH
```

Gambar 3.1 Preprocessing Data

Kode pada gambar digunakan untuk mempersiapkan lingkungan pengembangan guna menjalankan Object Detection API dengan TensorFlow dan akselerasi GPU menggunakan CUDA. Langkah-langkahnya meliputi instalasi pustaka PyYAML versi 5.3 untuk mengatasi masalah kompatibilitas di Google Colab, penurunan versi TensorFlow ke 2.8.0 untuk mendukung kompatibilitas dengan API dan CUDA, serta instalasi TensorFlow I/O versi 0.23.1 untuk mendukung berbagai format data I/O. Selanjutnya, kode menginstal CUDA Toolkit 11.0, termasuk menambahkan repositori CUDA, mendownload dan menginstal file instalasi lokal CUDA, menambahkan kunci repositori, memperbarui sistem, serta menginstal toolkit CUDA. Terakhir, variabel lingkungan `LD_LIBRARY_PATH` diatur untuk memastikan TensorFlow dapat menemukan pustaka CUDA selama runtime, sehingga memungkinkan akselerasi GPU untuk proyek pembelajaran mesin.

```
!pip install roboflow -q

from roboflow import Roboflow
rf = Roboflow(api_key="UQ3lZ0w4sZMK520cd4hm")
project = rf.workspace("bangkit-7pwvz").project("foodies-qbezq")
version = project.version(3)
dataset = version.download("tfrecord")

[?]
...
0.0/81.5 kB ? eta -:--:--
81.5/81.5 kB 6.9 MB/s eta 0:00:00
0.0/66.8 kB ? eta -:--:--
66.8/66.8 kB 5.9 MB/s eta 0:00:00
0.0/751.2 kB ? eta -:--:--
751.2/751.2 kB 27.5 MB/s eta 0:00:00
ERROR: pip's dependency resolver does not currently take into account all the packages that are installed. This behaviour is the source of the following dependency conflicts.
dask 2024.10.0 requires cloudpickle>=3.0.0, but you have cloudpickle 2.2.1 which is incompatible.
tf-models-official 2.8.0 requires pyyaml<6.0,>=5.1, but you have pyyaml 6.0.2 which is incompatible.
loading Roboflow workspace...
loading Roboflow project...
Downloading Dataset Version Zip in Foodies-3 to tfrecord:: 100%|#####| 116925/116925 [00:01<00:00, 64404.02it/s]

Extracting Dataset Version Zip to Foodies-3 in tfrecord:: 100%|#####| 11/11 [00:00<00:00, 32.57it/s]
```

Gambar 3.2 Memuat Dan Memproses Dataset

Kode pada gambar 3.2 merupakan kode untuk memproses dataset. Dataset didapat dari roboflow yang sebelumnya telah dilabeli sesuai dengan kelas yang ada. Pada identifikasi ini merupakan pembelajaran terarah yaitu data telah belajar dengan dilabeli pada Roboflow. Pada dataset ini terdiri dari 8 kelas.

```
### This creates a "labelmap.txt" file with a list of classes the object detection model will detect.
%bash
cat <<EOF >> /content/labelmap.txt
Ayam
Daging
Ikan
Nasi
Sayur
Tahu
Telur
Tempe
EOF
```

Gambar 3.3 Preprocessing Data

Kode pada gambar 3.3 merupakan perintah untuk labelmap. labelmap berfungsi untuk memetakan atau menghubungkan label numerik ke label tekstual dalam konteks pemrosesan data, seperti dalam klasifikasi gambar atau deteksi objek. Dalam pembelajaran ini terdapat 8 kelas diantaranya adalah ayam, daging, ikan, nasi, sayur, tahu, telur dan tempe.

3.3 Pelatihan Model

```
# Change the chosen_model variable to deploy different models available in the TF2 object detection zoo
chosen_model = 'ssd-mobilenet-v2'

MODELS_CONFIG = {
    'ssd-mobilenet-v2': {
        'model_name': 'ssd_mobilenet_v2_320x320_coco17_tpu-8',
        'base_pipeline_file': 'ssd_mobilenet_v2_320x320_coco17_tpu-8.config',
        'pretrained_checkpoint': 'ssd_mobilenet_v2_320x320_coco17_tpu-8.tar.gz',
    },
    'efficientdet-d0': {
        'model_name': 'efficientdet_d0_coco17_tpu-32',
        'base_pipeline_file': 'ssd_efficientdet_d0_512x512_coco17_tpu-8.config',
        'pretrained_checkpoint': 'efficientdet_d0_coco17_tpu-32.tar.gz',
    },
    'ssd-mobilenet-v2-fpn-lite-320': {
        'model_name': 'ssd_mobilenet_v2_fpn-lite_320x320_coco17_tpu-8',
        'base_pipeline_file': 'ssd_mobilenet_v2_fpn-lite_320x320_coco17_tpu-8.config',
        'pretrained_checkpoint': 'ssd_mobilenet_v2_fpn-lite_320x320_coco17_tpu-8.tar.gz',
    },
    'ssd_efficientdet-d7_1536': {
        'model_name': 'ssd_efficientdet_d7_1536x1536_coco17_tpu-32',
        'base_pipeline_file': 'ssd_efficientdet_d7_1536x1536_coco17_tpu-32.config',
        'pretrained_checkpoint': 'ssd_efficientdet_d7_1536x1536_coco17_tpu-32.tar.gz',
    },
}

# The centernet model isn't working as of 9/10/22
# 'centernet-mobilenet-v2': {
#     'model_name': 'centernet_mobilenetv2fpn_512x512_coco17_od',
#     'base_pipeline_file': 'pipeline.config',
#     'pretrained_checkpoint': 'centernet_mobilenetv2fpn_512x512_coco17_od.tar.gz',
# }

model_name = MODELS_CONFIG[chosen_model]['model_name']
pretrained_checkpoint = MODELS_CONFIG[chosen_model]['pretrained_checkpoint']
base_pipeline_file = MODELS_CONFIG[chosen_model]['base_pipeline_file']
```

Gambar 3. 4 Pelatihan model

Gambar 3.4 menetapkan model yang akan digunakan (ssd-mobilenet-v2) dan menyediakan konfigurasi untuk berbagai model deteksi objek lainnya. Setiap model memiliki nama model, file konfigurasi pipeline, dan checkpoint pelatihan yang terkait. Model yang dipilih kemudian dapat digunakan dalam proses pelatihan atau inferensi sesuai kebutuhan

```
# Create "mymodel" folder for holding pre-trained weights and configuration files
mkdir /content/models/mymodel/
cd /content/models/mymodel/

# Download pre-trained model weights
import tarfile
download_tar = 'http://download.tensorflow.org/models/object_detection/tf2/20200711/' + pretrained_checkpoint
wget (download_tar)
tar = tarfile.open(pretrained_checkpoint)
tar.extractall()
tar.close()

# Download training configuration file for model
download_config = 'https://raw.githubusercontent.com/tensorflow/models/master/research/object_detection/configs/tf2/' + base_pipeline_file
wget (download_config)

/content/models/mymodel
--2024-12-24 19:32:41-- http://download.tensorflow.org/models/object_detection/tf2/20200711/ssd_mobilenet_v2_320x320_coco17_tpu-8.tar.gz
Resolving download.tensorflow.org (download.tensorflow.org)... 74.125.195.207, 172.253.117.207, 142.251.188.207, ...
Connecting to download.tensorflow.org (download.tensorflow.org)|74.125.195.207|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 46842990 (44M) [application/x-tar]
Saving to: 'ssd_mobilenet_v2_320x320_coco17_tpu-8.tar.gz'

ssd_mobilenet_v2_32 100%[=====] 43.91M 164MB/s in 0.3s

2024-12-24 19:32:41 (164 MB/s) - 'ssd_mobilenet_v2_320x320_coco17_tpu-8.tar.gz' saved [46842990/46842990]

--2024-12-24 19:32:42-- https://raw.githubusercontent.com/tensorflow/models/master/research/object_detection/configs/tf2/ssd_mobilenet_v2_320x320_coco17_tpu-8.config
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.110.133, 185.199.111.133, 185.199.109.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.110.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Saving to: 'ssd_mobilenet_v2_320x320_coco17_tpu-8.config'
```

Gambar 3. 5 pelatihan model

Kode yang ditampilkan pada gambar berfungsi untuk menyiapkan direktori kerja, mengunduh model pra-latih, dan file konfigurasi yang dibutuhkan untuk melatih model deteksi objek. Pertama, folder bernama mymodel dibuat di dalam direktori /content/models/, dan direktori kerja dipindahkan ke folder tersebut.

➤ Train Model

```
# Run training!
python /content/models/research/object_detection_model_main_tf2.py \
  --pipeline_config_path=pipeline_file \
  --model_dir=model_dir \
  --logtostderr \
  --num_train_steps=num_steps \
  --sample_1_of_n_eval_examples=1

/usr/local/lib/python3.10/dist-packages/tensorflow_addons/utils/tfa_eol_msg.py:23: UserWarning:
TensorFlow Addons (TFA) has ended development and introduction of new features.
TFA has entered a minimal maintenance and release mode until a planned end of life in May 2024.
Please modify downstream libraries to take dependencies from other repositories in our TensorFlow community (e.g. Keras, Keras-CV, and Keras-MLP).
For more information see: https://github.com/tensorflow/addons/issues/2807

warnings.warn(

/usr/local/lib/python3.10/dist-packages/tensorflow_addons/utils/ensure_tf_install.py:53: UserWarning: TensorFlow Addons supports using Python ops for all TensorFlow versions above or equal to 2.13.0 and strictly below 2.16.0
The version of TensorFlow you are currently using is 2.8.0 and is not supported.
Some things might work, some things might not.
If you were to encounter a bug, do not file an issue.
If you want to make sure you're using a tested and supported configuration, either change the TensorFlow version or the TensorFlow Addons's version.
You can find the compatibility matrix in TensorFlow Addon's readme:
https://github.com/tensorflow/addons

warnings.warn(
2024-12-24 19:35:11.638564: W tensorflow/core/common_runtime/gpu/gpu_bfc_allocator.cc:39] Overriding allow_growth setting because the TF_FORCE_GPU_ALLOW_GROWTH environment variable is set. Original config value was 0.
INFO:tensorflow:Using MirroredStrategy with devices (/job:localhost/replica:0/task:0/device:GPU:0,)
I1224 19:35:11.677533 138917068081280 mirrored_strategy.py:374] Using MirroredStrategy with devices (/job:localhost/replica:0/task:0/device:GPU:0,)
INFO:tensorflow:Maybe overwriting train_steps: 10000
I1224 19:35:11.681168 138917068081280 config_util.py:552] Maybe overwriting train_steps: 10000
INFO:tensorflow:Maybe overwriting use_bfloat16: False
I1224 19:35:11.681333 138917068081280 config_util.py:552] Maybe overwriting use_bfloat16: False
WARNING:tensorflow:From /usr/local/lib/python3.10/dist-packages/object_detection_model_lib_v2.py:563: StrategyBase.experimental_distribute_datasets_from_function (from tensorflow.python.distribute.distribute_lib) is deprecated.
>>>
{'loss/localization_loss': 0.856738626,
 'loss/regulation_loss': 0.08663035,
 'loss/total_loss': 0.9220744}
```

Gambar 3. 6 train model

Pada gambar 4.3, num_steps adalah jumlah total langkah yang digunakan untuk melatih model. Pada model yang akan dilatih diatur num_steps sebanyak 40000. Semakin banyak langkah maka akan semakin lama model akan dilatih. Proses pelatihan dapat dihentikan di tengah proses pelatihan. Kemudian batch_size adalah jumlah gambar yang digunakan per langkah pelatihan. Pada model ini batch_size yang digunakan adalah 16 dikarenakan pre-trained model yang digunakan adalah SSD MobileNetV2. Batch size 16 bagus untuk model ssd karena

memungkinkan pelatihan yang efisien, memanfaatkan kapasitas memori GPU yang tersedia, dan memberikan stabilitas gradien yang cukup untuk konvergensi yang efektif. Dalam mengatur batch_size dapat disesuaikan berdasarkan GPU yang digunakan dalam pelatihan.

3.4 Evaluasi Model

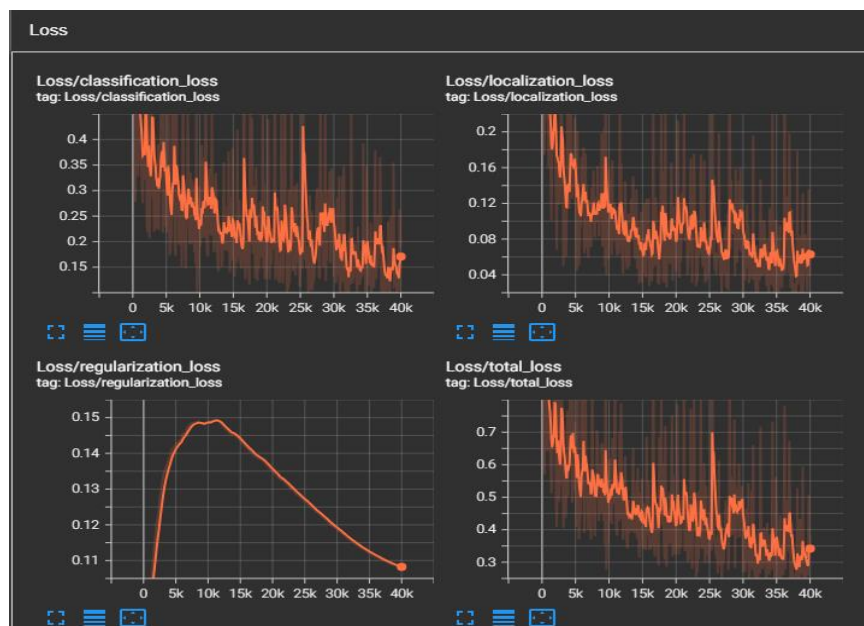
➤ Akurasi model

mAP Results	
Class	Average mAP @ 0.5:0.95
Ayam	87.52%
Daging	80.00%
Ikan	81.38%
Nasi	78.17%
Sayur	68.49%
Tahu	85.27%
Telur	86.62%
Tempe	81.43%
Overall	81.11%

Gambar 3. 7 MAP

Pada pembelajaran menggunakan SSD MobileNetV2 menggunakan evaluasi MAP atau Mean Average Precision. mAP memberikan penilaian yang lebih menyeluruh dibandingkan hanya menggunakan akurasi biasa karena mempertimbangkan beberapa hal yaitu Lokasi deteksi (bounding box), Keakuratan klasifikasi, Tingkat ambang batas IoU.

➤ Grafik visual



Gambar 3. 8 Grafik pelatihan

Model yang telah dilatih menunjukkan perbaikan performa selama proses pelatihan, dapat dilihat dengan penurunan nilai-nilai loss seiring bertambahnya jumlah iterasi. Meskipun terdapat fluktuasi pada nilai-nilai loss, tetapi bukan masalah dan merupakan hal yang wajar. Hal itu menunjukkan bahwa model sedang belajar dan meningkatkan kemampuannya untuk melakukan tugas-tugasnya. Prediksi akurasi

➤ Testing Model



Gambar 3. 9 Testing Model

Terlihat model telah dapat mendeteksi komponen makanan dari gambar yang telah diupload. Pada gambar tersebut terlihat *bounding box* yang memberikan hasil deteksi berupa nasi, ayam, sayur dan tempe. Masing-masing makanan tersebut dideteksi dengan tingkat kepercayaan diri sebesar 97% untuk nasi, 79% untuk ayam, 97% dan 94% untuk masing-masing tempe dan 97% untuk sayur. Dengan hasil deteksi dan tingkat kepercayaan diri sebesar itu, maka dapat dikatakan model telah berhasil melakukan deteksi makanan dengan cukup baik dan akurat.

4. Kesimpulan

Berdasarkan penelitian yang dilakukan terhadap pelatihan dan pengujian model CNN *transfer learning*, yaitu MobileNetV2, dapat disimpulkan bahwa arsitektur MobileNetV2 dalam *deep learning* memberikan performa yang sangat baik dalam identifikasi makanan sehat mahasiswa Teknik Universitas Bengkulu. Pelatihan MobileNetV2 dijalankan sebanyak 25.000 *epoch* dengan menggunakan dataset yang mencakup gambar makanan dari Kaggle dan foto makanan mandiri, yang setiap epoch memproses gambar untuk mengoptimalkan akurasi. Model ini mencapai tingkat akurasi yang tinggi, menunjukkan kemampuannya dalam mengenali dan mengklasifikasikan makanan sehat dengan akurat. Selama proses pelatihan, terjadi penurunan yang signifikan pada nilai *loss* yang menunjukkan bahwa model telah belajar dengan baik untuk mengidentifikasi pola dalam gambar makanan. Nilai *classification loss* dan *localization loss* menunjukkan hasil yang memadai, yang berarti model efektif dalam mengklasifikasikan makanan dengan presisi yang baik. Fluktuasi pada nilai *loss* tetap menunjukkan tren penurunan, menandakan bahwa model sedang berproses dengan optimal. Secara keseluruhan, model yang telah dilatih menunjukkan perbaikan performa yang signifikan sepanjang proses pelatihan, yang dapat dilihat melalui penurunan nilai-nilai *loss* seiring bertambahnya jumlah iterasi. Dibandingkan dengan model lain yang sudah diuji, MobileNetV2 terbukti menjadi pilihan optimal dalam mengidentifikasi makanan sehat, dengan hasil akurasi yang konsisten baik pada pelatihan maupun pengujian.