URBAN VISUAL POLLUTION DATASET

Laporan Project Deep Learning

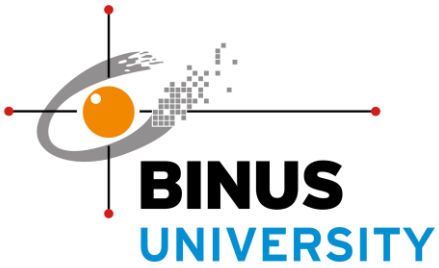
Oleh

Kelompok 3 LC09

2501979322 - Aprilia Jocelyne Gunawan

2502055204 - Elena Ghini Rachman

2502042095 - Patricia Laurencia Woge



PRODI COMPUTER SCIENCE

FAKULTAS SCHOOL OF COMPUTER SCIENCE

UNIVERSITAS BINA NUSANTARA

JAKARTA

2023

BAB 1
PENDAHULUAN

Dataset "Urban Visual Pollution" adalah sebuah data polusi visual di lingkungan perkotaan. Polusi visual terjadi ketika ada elemen visual yang mengganggu atau mengurangi kualitas nilai estetis suatu lingkungan. Dataset ini mungkin berisi informasi tentang faktor-faktor yang berkontribusi terhadap polusi visual, seperti reklame, tanda-tanda jalan, bangunan, dan sebagainya. Dataset berisi citra jalan yang diambil dari kendaraan yang bergerak. Polusi visual ini termasuk masalah yang baru dibandingkan pencemaran lainnya. Sehingga diperlukan analisa untuk mendefinisikan dan mengevaluasinya. Tujuan analisa ini adalah untuk memahami dan menganalisis faktor-faktor tersebut dan bagaimana mereka berkontribusi terhadap polusi visual di lingkungan perkotaan. Sehingga dengan pendekatan training dan testing diharapkan dapat mengidentifikasi gambar untuk mengklasifikasikan polusi visual.

Pertama-tama melakukan eksplorasi data dari train.csv dan test.csv untuk mengetahui lima data pertama, informasi data, dan deskripsi statistik variabel. Dari lima data pertama train.csv diketahui terdapat 7 variabel. Variabel class menunjukkan kelas atau kategori dari objek yang ada dalam gambar, variabel image_path merupakan path atau alamat file gambar yang terkait dengan objek, variabel name merupakan label atau nama dari objek yang ada dalam gambar, dan variabel xmax, xmin, ymax, ymin menunjukkan koordinat batas objek dalam gambar. Xmax adalah koordinat x maksimum, xmin adalah koordinat x minimum, ymax adalah koordinat y maksimum, dan ymin adalah koordinat y minimum.

	class	image_path	name	xmax	xmin	ymax	ymin
0	3.0	4a48c42c9579ec0399e6c5a3e825e765.jpg	GARBAGE	797.0	701.0	262.0	211.0
1	3.0	4a48c42c9579ec0399e6c5a3e825e765.jpg	GARBAGE	932.0	786.0	329.0	238.0
2	3.0	4a48c42c9579ec0399e6c5a3e825e765.jpg	GARBAGE	736.0	657.0	275.0	229.0
3	7.0	ea906a663da6321bcef78be4b7d1afff.jpg	BAD_BILLBOARD	986.0	786.0	136.0	0.0
4	8.0	1c7d48005a12d1b19261b8e71df7cafe.jpg	SAND_ON_ROAD	667.0	549.0	228.0	179.0

Gambar 1. Lima data pertama train.csv

Dari lima data pertama test.csv diketahui data ini hanya memiliki satu variabel yaitu variabel image_path yang berisi alamat file gambar yang terkait dengan data uji. Setiap baris menunjukkan satu file gambar yang akan digunakan untuk pengujian atau prediksi.

	image_path
0	953ab1447c46ecfef67ab14629cd70c7.jpg
1	e4ddbaa7970fca225a51288ce5f7d3f9.jpg
2	5b8120d69607a077b7583334be3ba18b.jpg
3	138b1dc82005b4c33e4886260649d313.jpg
4	0f91ec1533b845b13089f8cf4e0a36f7.jpg

Gambar 2. Lima data pertama test.csv

Info data yang didapatkan dari train.csv adalah data ini memiliki total 19950 baris, dengan indeks yang dimulai dari 0 hingga 19949. Selain itu menampilkan informasi setiap kolom, seperti nama kolom, jumlah nilai non-null, dan tipe data tiap variabel. Kolom class memiliki 19950 nilai non-null dengan tipe data float64, Kolom image_path memiliki 19950 nilai non-null dengan tipe data objek (string). Kolom name memiliki 19950 nilai non-null dengan tipe data objek (string). Kolom xmax, xmin, ymax, ymin memiliki 19950 nilai non-null dengan tipe data float64.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 19950 entries, 0 to 19949
Data columns (total 7 columns):
 #   Column      Non-Null Count  Dtype  
---  ------      --------------  -----  
 0   class       19950 non-null  float64
 1   image_path  19950 non-null  object 
 2   name        19950 non-null  object 
 3   xmax        19950 non-null  float64
 4   xmin        19950 non-null  float64
 5   ymax        19950 non-null  float64
 6   ymin        19950 non-null  float64
dtypes: float64(5), object(2)
memory usage: 1.1+ MB
```

Gambar 3. Informasi data train.csv

Info data yang didapatkan dari test.csv adalah data ini memiliki total 2092 baris, dengan indeks yang dimulai dari 0 hingga 2091. Menampilkan informasi tentang kolom, termasuk nama kolom, jumlah nilai non-null, dan tipe data. Kolom image_path memiliki 2092 nilai non-null dengan tipe data objek (string).

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2092 entries, 0 to 2091
Data columns (total 1 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   image_path  2092 non-null   object
dtypes: object(1)
memory usage: 16.5+ KB
```

Gambar 4. Informasi data test.csv

Deskripsi statistik train.csv pada variabel class menunjukkan jumlah data dan rata-rata kelas yang ada. Selain itu, tidak ada informasi lain yang dapat diperoleh dari statistik deskriptif ini karena variabel class merupakan variabel kategori dan bukan numerik. Variabel xmax, xmin, ymax, dan ymin merupakan variabel numerik. Hal deskriptif statistic menunjukkan jumlah data, nilai rata-rata, standar deviasi, nilai minimum, nilai kuartil pertama, kuartil tengah, kuartil ketiga, dan nilai maksimum tiap variabel.

	class	xmax	xmin	ymax	ymin
count	19950.000000	19950.000000	19950.000000	19950.000000	19950.000000
mean	4.055388	572.813634	362.418647	314.202506	201.514937
std	2.502491	264.445257	264.246281	102.133871	85.536614
min	0.000000	30.000000	-397.000000	20.000000	-150.000000
25%	3.000000	357.000000	147.000000	242.000000	149.000000
50%	3.000000	567.000000	350.000000	296.000000	195.000000
75%	4.000000	789.000000	581.000000	372.000000	251.000000
max	10.000000	1384.000000	935.000000	698.000000	500.000000

Gambar 5. Deskriptif statistik train.csv

Hasil deskriptif statistic test.csv menunjukkan jumlah nilai unik dalam kolom image_path. Dalam hal ini, terdapat 2092 nilai unik, yang berarti setiap nilai dalam kolom ini adalah unik. Menunjukkan JUGA nilai yang paling sering muncul dalam kolom image_path. Terakhir menunjukkan frekuensi kemunculan nilai yang paling sering dalam kolom image_path.

	image_path
count	2092
unique	2092
top	953ab1447c46ecfef67ab14629cd70c7.jpg
freq	1

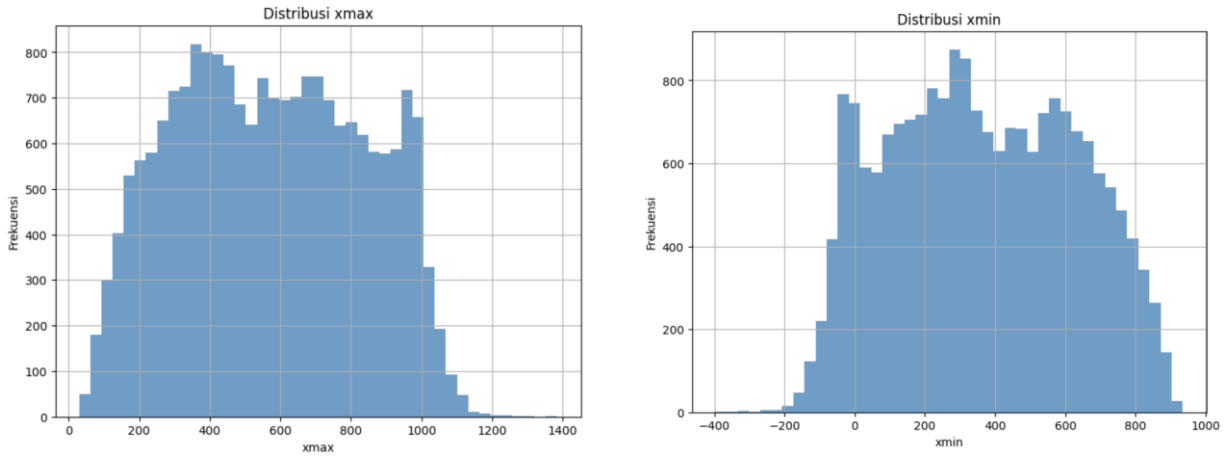
Gambar 6. Deskriptif statistik test.csv

Pada train.csv juga dilakukan analisa untuk mengetahui jumlah data tiap kelas

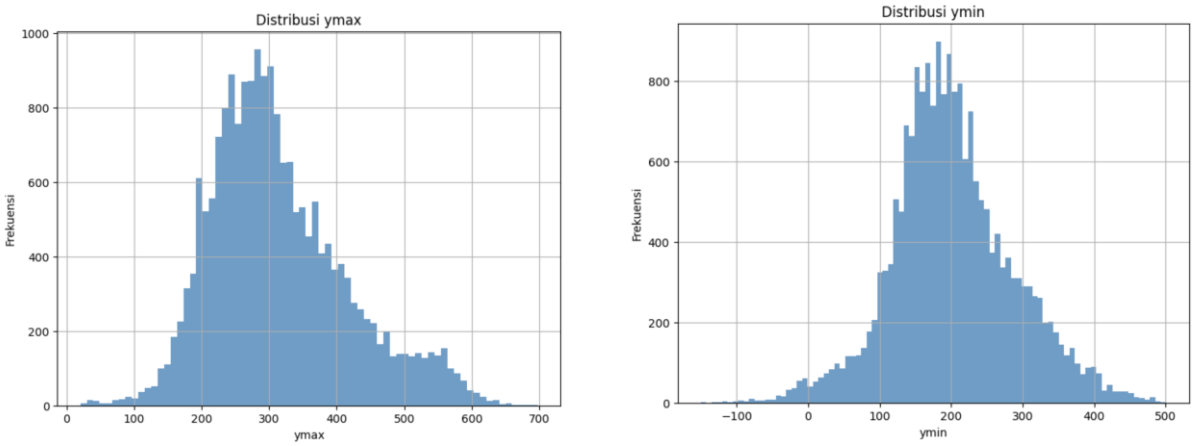
```
3.0     8597
4.0     2730
2.0     2625
9.0     2253
7.0     1555
0.0     1124
8.0      748
10.0     127
1.0      107
5.0       83
6.0        1
Name: class, dtype: int64
```

Gambar 7. Jumlah data tiap variabel class pada train.csv

Kedua dilakukan visualisasi data menggunakan plot atau grafik untuk memahami distribusi variabel dan korelasi antara variabel data train.csv. Menampilkan frekuensi distribusi tiap variabel dalam bentuk grafik histogram untuk variabel tipe numerik, yaitu xmax, xmin, ymax, dan ymin.



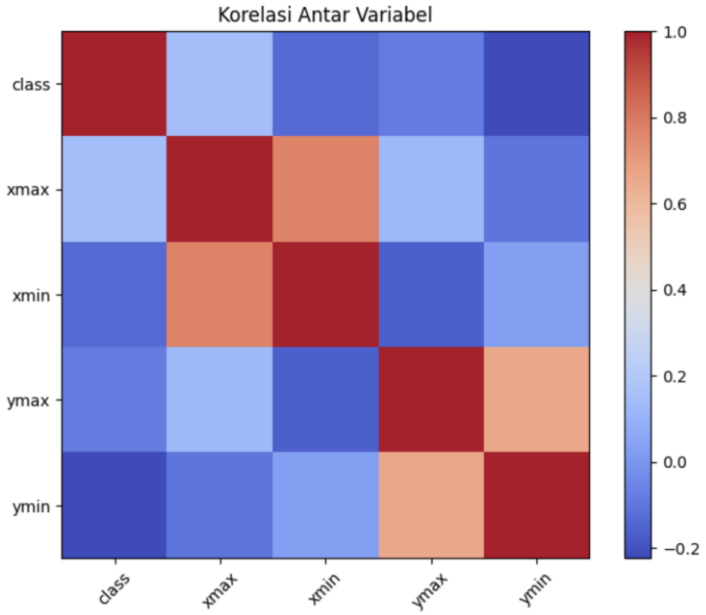
Gambar 8. Distribusi variabel xmax dan xmin pada train.csv



Gambar 9. Distribusi variabel ymax dan ymin pada train.csv

Melakukan juga analisa korelasi tiap variabel untuk mengetahui tingkat korelasi tiap variabel. Ditampilkan dengan melakukan plot heatmap, dengan skala warna yang umum

digunakan yaitu warna merah menunjukkan korelasi yang tinggi dan biru menunjukkan korelasi yang rendah. Dari hasil plot heatmap diketahui untuk korelasi variabel class memiliki korelasi paling tinggi dengan xmax dan paling rendah dengan ymin. Untuk variabel xmax memiliki korelasi paling tinggi dengan xmin dan paling rendah dengan ymin, Untuk variabel xmin memiliki korelasi paling tinggi dengan xmax, paling rendah dengan class dan ymax. Untuk variabel ymax memiliki korelasi paling tinggi dengan ymin dan paling rendah dengan xmin. Terakhir untuk variabel ymin memiliki korelasi paling tinggi dengan ymax dan paling rendah dengan class.



Gambar 10. Korelasi antar variabel tarin.csv

Data train.csv dan test.csv tidak memiliki nilai-nilai yang hilang dan kolom yang tidak relevan, oleh karena itu tidak dilakukan transformasi data.

```
class         0
image_path    0
name          0
xmax          0
xmin          0
ymax          0
ymin          0
dtype: int64
```

```
image_path    0
dtype: int64
```

Gambar 11. Tidak ada data yang hilang pada train.csv dan test.csv

Dalam mendapatkan model yang baik dilakukan evaluasi performa model menggunakan nilai akurasi. Modelling pertama untuk melatih model klasifikasi dataset ini menggunakan algoritma Adam (Adaptive Moment Estimation). Adam menggabungkan metode-metode dari algoritma Momentum dan RMSprop. Alasan menggunakan Adam dikarenakan Adam menggunakan metode adaptif yang memperbarui laju pembelajaran

berdasarkan momentum dan gradien kedua. Hal ini memungkinkan algoritma beradaptasi dengan cepat terhadap keadaan lokal maupun global dari dataset gambar sehingga meningkatkan efisiensi dan kecepatan konvergensi. Selain itu algoritma ini mampu menangani ruang parameter yang besar dan jumlah sampel yang banyak dengan efisien, memiliki mekanisme yang baik untuk menangani masalah gradien sehingga mempercepat proses pembelajaran pada dataset gambar, algoritma ini juga lebih toleran terhadap pilihan hiperparameter seperti laju pembelajaran awal, sehingga lebih mudah digunakan dan membutuhkan sedikit penyetelan.

Kemudian dilakukan tuning untuk mendapatkan model yang lebih baik menggunakan algoritma SGD dan RMSprop. Menggunakan algoritma SGD (Stochastic Gradient Descent) dikarenakan SGD secara stokastik memperbarui parameter model pada setiap langkah berdasarkan sampel data tunggal atau sekelompok kecil sampel data. Ini membuatnya lebih efisien secara komputasional. SGD juga dapat memiliki kecenderungan untuk melompati minimum lokal yang buruk yang mungkin terjadi pada data yang kompleks atau pada model yang kompleks sehingga memungkinkan SGD menemukan minimum lokal yang lebih baik dan memperbaiki model secara lebih efektif. Selain itu algoritma ini juga dapat mengatasi perubahan tiba-tiba dalam gradien atau ketidakstabilan pada tahap pelatihan.

Alasan menggunakan algoritma RMSprop (Root Mean Square Propagation) karena menggunakan teknik adaptif untuk menyesuaikan laju pembelajaran pada setiap parameter model. Hal ini membantu algoritma mengatasi tantangan seperti perbedaan skala antar fitur atau gradien yang berfluktuasi secara signifikan. Dengan menyesuaikan laju pembelajaran secara adaptif, RMSprop dapat membantu model menemukan minimum lokal dengan lebih cepat dan stabil. RMSprop juga menggunakan teknik root mean square (RMS) untuk menghitung perubahan laju pembelajaran berdasarkan gradien historis. Dengan melakukan normalisasi, membantu menstabilkan pelatihan model dan mengurangi masalah gradien yang meledak atau menghilang. Selain itu dapat menemukan pola dan fitur yang relevan dalam data gambar sehingga membantu meningkatkan akurasi dan performa model dalam klasifikasi dataset gambar.

BAB 2
METODOLOGI

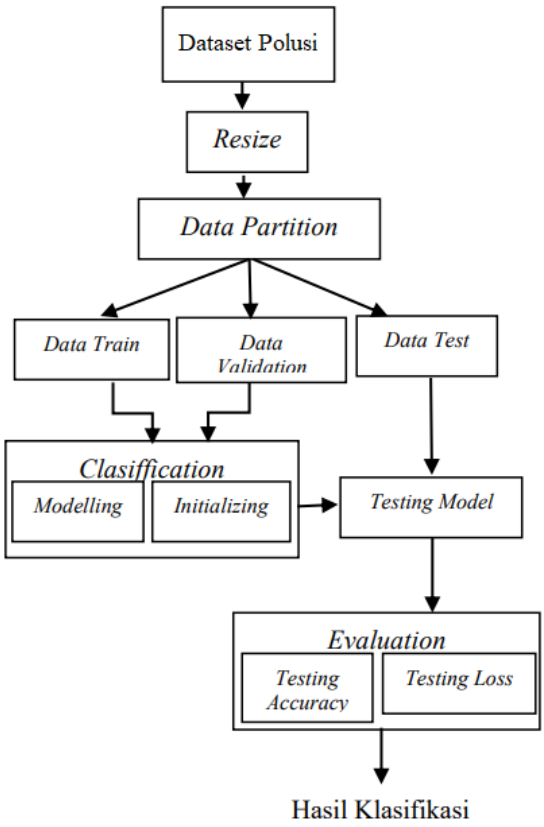
Klasifikasi gambar dilakukan untuk menganalisis polusi visual di lingkungan perkotaan. Sehingga dilakukan metode eksperimental untuk mengetahui hasilnya. Data yang diambil adalah data sekunder, yakni dari Kaggle. Untuk mencapai hasil yang baik, dilakukan eksperimen menggunakan arsitektur dari CNN dengan model pertama dengan MobileNetV2 dan model kedua dengan EfficientNetB0. Dengan pra-pemrosesan gambar yang melibatkan resizing dan normalisasi dikedua model tersebut. Model-model klasifikasi gambar tersebut akan dibandingkan hasilnya pada evaluation yang disertai dengan jumlah parameter kurang dari 10M.

Pada model pertama, MobileNetV2 didasarkan pada struktur residu terbalik, dimana koneksi residual berada di antara layer bottleneck dan meningkatkan kinerja model seluler canggih pada berbagai tugas dan tolok ukur serta di seluruh spektrum ukuran model yang berbeda. MobileNetV2 yang membuatnya berbeda dari jenis model deep learning lainnya, akurasi prediksi yang diberikan MobileNetV2 sangat tinggi meskipun tidak terlalu banyak merugikan komputasi dan memori dalam hal biaya (Lum dkk., 2020).

Metode kedua menggunakan EfficientNetB0, metode yang digunakan dengan skala secara seragam atau "compound scaling". Pendekatan ini melibatkan peningkatan resolusi gambar, kedalaman jaringan, dan lebar jaringan secara seragam dan proporsional. Dengan meningkatkan semua aspek ini secara bersamaan, model mencapai efisiensi yang lebih tinggi daripada model sebelumnya. Skala secara seragam ini dilakukan dengan menggabungkan tiga faktor skala: lebar jaringan (width), kedalaman jaringan (depth), dan resolusi gambar input (resolution). Dalam EfficientNetB0, faktor-faktor ini dikendalikan menggunakan parameter skala komposit (compound scaling parameter) yang disebut "phi". Parameter phi digunakan untuk mengubah ukuran model dengan proporsi tertentu. Misalnya, jika phi = 1, maka model tersebut akan memiliki ukuran baseline. Jika phi ditingkatkan, maka model tersebut akan menjadi lebih besar dengan resolusi yang lebih tinggi, lebih dalam, dan lebih lebar.

a. Desain Sistem

   Sebelum eksperimen dilakukan, mendesain terlebih dahulu untuk rancangan sistem dalam pengolahan citra dengan mengklasifikasikan polusi diperkotaan dengan cara sebagai berikut.



Gambar 12. Desain Sistem

Pengolahan dataset polusi dilakukan sesuai pada gambar diatas dengan penjelasan langkah-langkah sebagai berikut:

1. Citra polusi diperkotaan sebelumnya harus melalui proses pre-processing yaitu resize citra dengan ukuran 224x224 piksel yang tujuannya untuk memperkecil banyak piksel citra dan mendapatkan resolusi citra yang sesuai sehingga inputan yang dibaca oleh sistem mudah terbaca, mudah dalam proses pengolahan citra serta mencegah terjadinya proses perlambatan saat mengeksekusi citra yang lama ketika program dijalankan akibat besarnya jumlah piksel pada citra pada ruang penyimpanan citra yang diproses (Zarkasi & Ubaya, 2016).
2. Tahap selanjutnya, dilakukan partisi data yang membagi data menjadi tiga data yaitu data train, data test dan data validasi. Data train untuk merakit model atau melatih model. Eksperimen kali ini tidak bisa diolah seluruh data train sehingga dilakukan tratified sampling buat ambil sample tiap kelas gambar sebanyak 200 tiap kelas. Data test digunakan setelah proses latihan selesai untuk menguji model. Data test membuat model bahkan manusia tidak untuk melihat sampel ketika proses pelatihan data sehingga dengan ini disebut sebagai unseen data. Terakhir, data validasi untuk generalisasi supaya model bisa mengenal pola secara genetik sehingga dengan data validasi digunakan untuk mengoptimalkan ketika ingin melatih model (Putra, 2019).

Para tahap ini eksperimen dilakukan dengan mempartisi data dengan pembagian data train sebesar 60%, 20% untuk data test dan 20% data validation.

3. Lalu, eksperimen dilakukan mengklasifikasi polusi dengan mengumpulkan objek/entitas yang sama dan memisahkannya apabila tidak sama (Firasari et al., 2020). Klasifikasi dilakukan untuk generalisasi struktur/model yang diketahui supaya diaplikasikan pada beberapa data baru (Suyanto, 2017). Proses ini dengan model EfficientNetB0 dan MobileNetV2 untuk pendeteksian polusi di perkotaan. Dimana klasifikasi polusi dilakukan ke dalam sejumlah kategori polusi yang terdiri dari sekitar 9 kelas polusi. Untuk hasil akurasi dan klasifikasi akan dibahas pada poin hasil dan pembahasannya
4. Tahap selanjutnya, dilakukan testing model selanjutnya dilakukan klasifikasi yang akan dilakukan proses testing dengan data test yang sebelumnya sudah dipartisi.
5. Hasil klasifikasi polusi di lingkungan perkotaan dilakukan evaluasi pengukuran model. Metode evaluasi digunakan untuk perbandingkan hasil klasifikasi yang selesai dilakukan oleh sistem dengan hasil klasifikasi yang sebenarnya (Purnama, 2019). Evaluasi dilakukan dengan memvisualisasikan ke bentuk confusion matrix.
6. Evaluasi yang diuji pada eksperimen kali ini adalah akurasi yang digunakan sebagai gambaran keakuratan model dan kinerja model dalam mengklasifikasikan polusi diperkotaan. Sehingga eksperimen kali ini menunjukan tingkat prediksi dengan nilai yang sebenarnya/aktual (Putra, 2019).

BAB 3
HASIL DAN ANALISA

Link code:
https://colab.research.google.com/drive/1UMiLA5LihJ41Xq8ROND9cofGPD9Ih4Ty?usp=sharing

Berikut code dari model CNN yang digunakan:

```
# Model MobileNetV2
base_model_1 = MobileNetV2(weights='imagenet', include_top=False, input_shape=(224, 224, 3))
x = base_model_1.output
x = GlobalAveragePooling2D()(x)
output_1 = Dense(11, activation='softmax')(x)
model_1 = Model(inputs=base_model_1.input, outputs=output_1)

# Model EfficientNetB0
base_model_2 = EfficientNetB0(weights='imagenet', include_top=False, input_shape=(224, 224, 3))
x = base_model_2.output
x = GlobalAveragePooling2D()(x)
output_2 = Dense(11, activation='softmax')(x)
model_2 = Model(inputs=base_model_2.input, outputs=output_2)
```

Gambar 13. Code Model

```
# Latih model 1
model_1.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
model_1.fit(train_images, train_labels, validation_data=(val_images, val_labels), epochs=10)
```

Gambar 14. Latih Model MobileNetV2

```
# Latih model 2
model_2.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
model_2.fit(train_images, train_labels, validation_data=(val_images, val_labels), epochs=10)
```

Gambar 15. Latih Model 2 EfficientNttB0

Parameter yang digunakan sebelum dilakukan tuning:
optimizer : Adam
epoch : 10
learning rate : 0.001 (*by default)*
batch size : 32 (*by default)*

Diperoleh hasil akurasi pada evaluasi sebesar 12% pada model MobileNetV2 dan 9% pada EfficientNetB0. Akurasi yang rendah pada kedua model tersebut mungkin disebabkan oleh faktor jumlah data yang terbatas karena digunakan sampling untuk mengurangi jumlah data, sehingga akurasi model terpengaruh. Oleh karena itu, dilakukan tuning hyperparameter seperti learning rate, batch size, dan optimizer yang digunakan untuk mencoba meningkatkan akurasi model.

Pada proses tuning ini, dilakukan Early Stopping dengan memantau val_accuracy dan memberikan patience=3 untuk menghentikan pelatihan jika tidak ada peningkatan dalam 3 epoch berturut-turut.

```
from tensorflow.keras.callbacks import EarlyStopping

early_stopping = EarlyStopping(monitor='val_accuracy', patience=3)
```

Gambar 16. Early Stopping

Berikut hasil akurasi model evaluasi setelah tuning parameter

Parameter					**Akurasi Model Evaluasi**	
Optimizer	**Learning rate**	**Momentum**	**Epoch**	**Batch size**	**MobileNetV2**	**EfficientNetB0**
Adam	0.01	-	10	16	7.5%	7.5%
	0.1				10.5%	10%
SGD	0.01	0.9			10.5%	29.3%
		0.5			10.5%	37.9%
RMSprop	0.001	-			10.5%	25%
	0.01				17.9%	10.7%

Tabel 1. Perbandingan Akurasi Parameter Tuning

Pada tuning parameter ini, seluruh batch size diganti menjadi 16 dari yang sebelumnya 32. Hal ini dilakukan karena dilakukan sampling sehingga dataset yang digunakan pada model pelatihan relatif kecil, sehingga ukuran batch yang lebih kecil mungkin lebih sesuai dengan harapan membantu model menggeneralisasi lebih baik dan mencegah overfitting.

Dari hasil eksperimen tuning parameter dilakukan, berikut adalah analisis terhadap hasil evaluasi model pada kedua arsitektur yang digunakan (MobileNetV2 dan EfficientNetB0):

1. Model MobileNetV2:
   - ➢ Awalnya, model MobileNetV2 dengan parameter default (optimizer: Adam, learning rate: 0.001, batch size: 32) menghasilkan akurasi evaluasi sebesar 12%.
   - ➢ Setelah tuning parameter dengan mengurangi batch size menjadi 16 dan tetap menggunakan optimizer Adam dengan meningkatkan learning rate menjadi 0.01, akurasi evaluasi turun menjadi 7.5%.
   - ➢ Ketika learning rate ditingkatkan menjadi 0.1, akurasi evaluasi meningkat menjadi 10.5%.
   - ➢ Pergantian optimizer menjadi SGD dengan learning rate 0.01 dan momentum 0.9 menghasilkan akurasi evaluasi yang masih sama, yaitu 10.5%.

- ➢ Pengurangan momentum menjadi 0.5 dengan parameter lain yang sama pada SGD tidak mengubah akurasi evaluasi, masih 10.5%.
- ➢ Dengan menggunakan optimizer RMSprop dan learning rate 0.001, akurasi evaluasi tetap 10.5%.
- ➢ Namun, dengan learning rate RMSprop ditingkatkan menjadi 0.01, akurasi evaluasi meningkat menjadi 17.9%.

2. Model EfficientNetB0:

- ➢ Pada awalnya, model EfficientNetB0 dengan parameter default menghasilkan akurasi evaluasi sebesar 9%.
- ➢ Setelah tuning parameter dengan batch size 16 dan learning rate 0.01 menggunakan optimizer Adam, akurasi evaluasi turun menjadi 7.5%.
- ➢ Peningkatan learning rate menjadi 0.1 meningkatkan akurasi evaluasi menjadi 10%.
- ➢ Dengan optimizer SGD, learning rate 0.01, dan momentum 0.9, akurasi evaluasi meningkat drastis menjadi 29.3%.
- ➢ Pengurangan momentum menjadi 0.5 dengan parameter lain yang sama pada SGD menghasilkan peningkatan lebih lanjut pada akurasi evaluasi, yaitu 37.9%.
- ➢ Ketika menggunakan optimizer RMSprop dengan learning rate 0.001, akurasi evaluasi menjadi 25%.
- ➢ Namun, dengan learning rate RMSprop ditingkatkan menjadi 0.01, akurasi evaluasi turun menjadi 10.7%.

Sehingga dapat diperoleh *insight*:

1. Model MobileNetV2:

- Peningkatan learning rate dari 0.001 menjadi 0.1 tidak menghasilkan peningkatan akurasi evaluasi yang signifikan, menunjukkan bahwa learning rate yang lebih tinggi menurunkan kinerja model.
- Penggunaan optimizer SGD dengan momentum 0.9 tidak memberikan perbaikan dalam akurasi evaluasi dibandingkan dengan optimizer Adam.
- Peningkatan akurasi evaluasi terbesar diperoleh saat menggunakan optimizer RMSprop dengan learning rate 0.01.
- Pengurangan batch size tidak menghasilkan peningkatan yang signifikan dalam akurasi evaluasi.

2. Model EfficientNetB0:

- Model EfficientNetB0 cenderung memiliki akurasi evaluasi yang lebih tinggi dibandingkan dengan MobileNetV2 dalam eksperimen ini.
- Peningkatan akurasi evaluasi yang signifikan diperoleh saat menggunakan optimizer SGD dengan momentum 0.5, menunjukkan bahwa momentum yang lebih rendah dapat meningkatkan performa model ini.
- Peningkatan learning rate pada optimizer RMSprop tidak memberikan perbaikan yang signifikan dalam akurasi evaluasi.

BAB 4
KESIMPULAN

Dalam keseluruhan analisis, dapat disimpulkan bahwa peningkatan learning rate dan penggunaan optimizer dengan momentum yang lebih rendah (pada model EfficientNetB0) memberikan perbaikan dalam akurasi evaluasi. Namun, hasil akurasi yang dicapai masih rendah, sehingga mungkin perlu mempertimbangkan penyesuaian lain pada jumlah data yang digunakan untuk melatih model agar mendapatkan hasil yang lebih baik.

REFERENSI

Lum, K. Y., Goh, Y. H., dan Lee, Y. B., 2020, American Sign Language Recognition Based on MobileNetV2, Advances in Science, Technology and Engineering Systems Journal 5 (6), 481–488.

Firasari, E., Khasanah, N., Khultsum, U., Kholifah, D. N., Komarudin, R., & Widyastuty, W. (2020). Comparation of K-Nearest Neighboor (K-NN) and Naive Bayes Algorithm for the Classification of the Poor in Recipients of Social Assistance. Journal of Physics: Conference Series, 1641(1). https://doi.org/10.1088/1742- 6596/1641/1/012077.

Putra, J. W. G. (2019). Pengenalan Konsep Pembelajaran Mesin dan Deep Learning. Computational Linguistics and Natural Language Processing Laboratory, 4, 1–235. https://www.researchgate.net/publication/3237 00644.

Purnama, B. (2019). Pengantar Machine Learning. Informatika.

Zarkasi, A., & Ubaya, H. (2016). Vision Sebagai Pengolahan Citra Api. Konferensi Nasional Teknologi Informasi & Aplikasinya, 4, 39–44.