LAMPIRAN

Lampiran 3. Persetujuan Etik

KEMENTERIAN PENDIDIKAN TINGGI, SAINS,
DAN TEKNOLOGI
UNIVERSITAS PENDIDIKAN GANESHA
LEMBAGA PENELITIAN DAN PENGABDIAN KEPADA MASYARAKAT
KOMISI ETIK PENELITIAN
Jalan Udayana Singaraja, Bali Kode Pos
81116Tlp. (0362) 22570 Fax. (0362) 25735
Laman: www.undiksha.ac.id

KETERANGAN KELAIKAN ETIK
(ETHICAL CLEARANCE)
No: 093/UN.48.16.04/PT/2025

Komite Etik Penelitian Universitas Pendidikan Ganesha, dalam upaya melindungi hak asasi dan kesejahteraan subjek penelitian serta menjamin bahwa penelitian berjalan sesuai dengan pedoman *Internasional Conference on Harmonisation – Good Clinical Practice (ICH-GCP)* dan aturan lainnya yang berlaku, telah mengkaji dengan teliti dan menyetujui proposal penelitian berjudul :
The Research Ethics Committee Universitas Pendidikan Ganesha, in an effort to protect the basic right and welfare of the research subject and to ensure that research operates in accordance with International Conference on Harmonisation – Good Clinical Practice (ICH-GCP) guidelines and other applicable and regulations, has thoroughly reviewed and approved a research proposal entitled :

"Pengenalan Pola Garis Telapak Tangan Menggunakan *Convolutional Neural Network* Dengan Arsitektur *Alexnet, Efficientnet-B7* Dan *Mobilenetv3*"

Registration Number	: 074/02/21/07/2025
Nama Peneliti Utama *Principal Researcher*	: Komang Ratna Mutu Manikam
Peneliti Lain *Other Researcher*	: 1. Dr. Luh Joni Erawati Dewi, S.T., M.Pd. 2. I Nyoman Saputra Wahyu Wijaya, S.Kom., M.Cs.
Nama Institusi *Institution*	: Fakultas Teknik dan Kejuruan, Undiksha.
Tempat Penelitian *Research location*	: Universitas Pendidikan Ganesha
Versi Dokumen (tanggal masuk) *Document Version*	: 21 Juli 2025

proposal tersebut dibebaskan pelaksanaannya.
hereby declare that the proposal is exempted.

Ditetapkan di : Singaraja
Issued in
Tanggal : 06 Agustus 2025
Date
Ketua
Chairman,

Mengetahui,
Plt. Kepala LPPM Undiksha,

I Gusti Lanang Agung Parwata
NIP 196906061994121001

Komang Hendra Setiawan
NIP. 198209302009121003

Keterangan/*notes:*
Persetujuan etik ini berlaku selama satu tahun sejak tanggal ditetapkan.
This ethical clearance is effective for one year from the due date.
Pada akhir penelitian, laporan pelaksanaan penelitian harus diserahkan ke Komite Etik Penelitian.
At the end of the research, progress and final summary report should be submitted to Research Ethics Committee.
Jika ada perubahan atau penyimpangan protokol dan/atau perpanjangan penelitian, harus mengajukan kembali permohonan kajian etik peneitian.
If there is any protocol modification or deviation and/or extension of the study, the Principal Investigator must resubmit the protocol for approval.
Jika ada kejadian serius yang tidak diinginkan (KTD) harus segera dilaporkan ke Komite Etik Penelitian.
Serious Adverse Events (SAE) should be immediately reported to the Research Ethics Committee.

Lampiran 4. Dokumentasi Pengambilan *Dataset* Pribadi

Lampiran 5. Potongan Kode Pembagian *Dataset* dengan *K-Fold Cross Validation*

```
for ds_name in DATASET_NAMES:
    src_root = BASE_DIR / ds_name
    out_root = BASE_DIR / f"{ds_name}_KFold"

    if CLEAN_OUTPUT and out_root.exists():
        shutil.rmtree(out_root)

    X, y, class_names = build_file_list(src_root)
    if not X:
        print(f"[SKIP] Dataset kosong: {src_root}")
        continue

    label_map = {p: lbl for p, lbl in zip(X, y)}

    print(f"\n=== {ds_name} | total gambar: {len(X)} |
kelas: {len(set(y))} ===")

    skf = StratifiedKFold(n_splits=K, shuffle=True,
random_state=SEED)

    for fold_idx, (train_idx, val_idx) in
enumerate(skf.split(X, y), start=1):
        fold_dir = out_root / f"fold_{fold_idx}"
        train_dir = fold_dir / "train"
        val_dir   = fold_dir / "valid"

        ensure_class_dirs(train_dir, class_names)
        ensure_class_dirs(val_dir, class_names)

        train_files = [X[i] for i in train_idx]
        val_files   = [X[i] for i in val_idx]

        copy_files(train_files, train_dir, label_map)
        copy_files(val_files, val_dir, label_map)

        print(f"[KFOLD] {ds_name} fold_{fold_idx}:
train={len(train_files)}, valid={len(val_files)}")
```

Lampiran 6. Potongan Kode Augmentasi

```
shear_deg = 5.7
AUGS = [
    ("rot", transforms.RandomRotation(30)),

    ("zoom_out", transforms.RandomAffine(
        degrees=0,
        scale=(0.8, 0.8)
    )),

    ("zoom_in", transforms.RandomAffine(
        degrees=0,
        scale=(1.2, 1.2)
    )),

    ("shift_x", transforms.RandomAffine(
        degrees=0,
        translate=(0.10, 0.0)
    )),

    ("shift_y", transforms.RandomAffine(
        degrees=0,
        translate=(0.0, 0.10)
    )),

    ("shear", transforms.RandomAffine(
        degrees=0,
        shear=shear_deg
    )),

    ("bright", transforms.ColorJitter(
        brightness=(0.8, 1.2)
    )),
]

def augment_folder(folder: Path):
    for cdir in folder.iterdir():
        if not cdir.is_dir():
            continue

        imgs = [
            p for p in cdir.iterdir()
            if is_image_file(p) and "_aug_" not in
p.stem
        ]
```

```
        for img_path in imgs:
            img = Image.open(img_path).convert("RGB")

            for tag, transform in AUGS:
                aug_img = transform(img)
                out_name =
f"{img_path.stem}_aug_{tag}{img_path.suffix}"
                aug_img.save(cdir / out_name)
```

Lampiran 7. Potongan Kode Model Klasifikasi *AlexNet*

```
class AlexNetCustom(nn.Module):
    def __init__(self, num_classes):
        super().__init__()
        self.features = nn.Sequential(
            nn.Conv2d(3, 16, 3, padding=1), nn.ReLU(),
nn.MaxPool2d(2),
            nn.Conv2d(16, 32, 3, padding=1), nn.ReLU(),
nn.MaxPool2d(2),
            nn.Conv2d(32, 64, 3, padding=1), nn.ReLU(),
nn.MaxPool2d(2),
            nn.Dropout(0.5)
        )
        self.classifier = nn.Sequential(
            nn.Linear(64 * 28 * 28, 500),
            nn.ReLU(),
            nn.Dropout(0.5),
            nn.Linear(500, num_classes)
        )

    def forward(self, x):
        x = self.features(x)
        x = torch.flatten(x, 1)
        return self.classifier(x)
```

Lampiran 8. Potongan Kode Model Klasifikasi *EfficientNet*

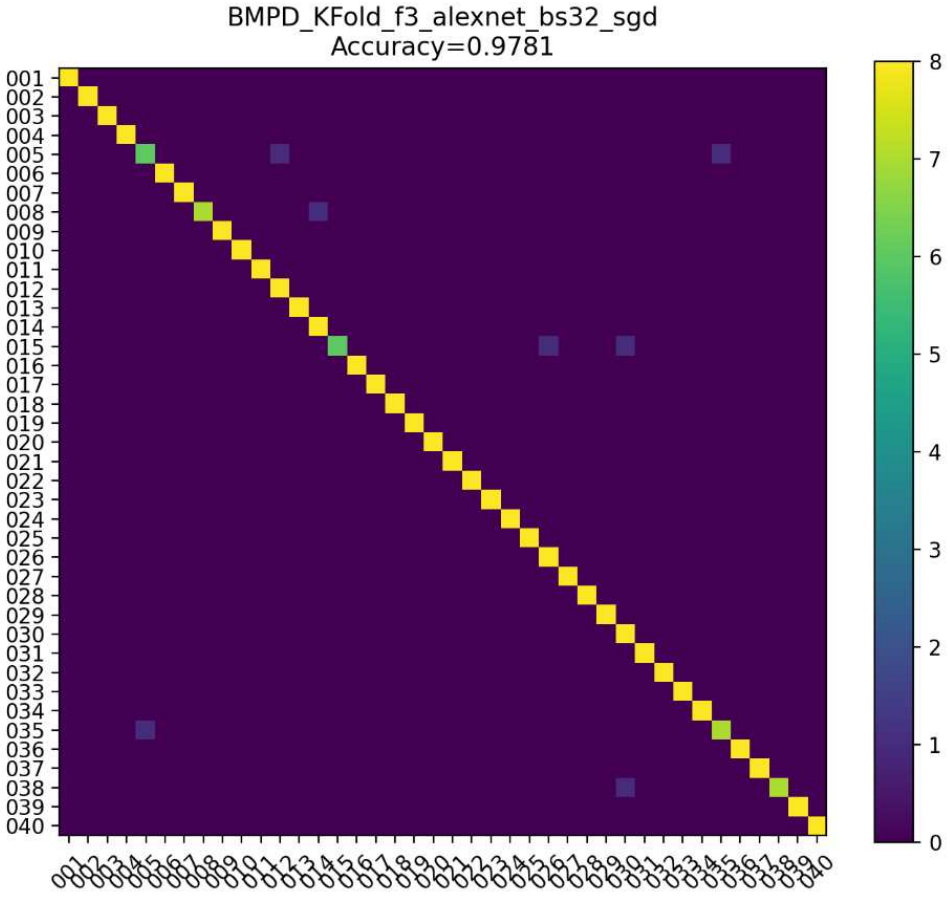
```
class EfficientNetB0(nn.Module):
    def __init__(self, num_classes=1000):
        super(EfficientNetB0, self).__init__()
        self.stem = nn.Sequential(
            nn.Conv2d(3, 32, 3, stride=2, padding=1,
bias=False),
            nn.BatchNorm2d(32),
            Swish()
        )
        self.blocks = nn.Sequential(
            MBConvBlock(32, 16, 3, 1, 1, 0.25),
            MBConvBlock(16, 24, 3, 2, 6, 0.25),
            MBConvBlock(24, 40, 5, 2, 6, 0.25),
            MBConvBlock(40, 80, 3, 2, 6, 0.25),
            MBConvBlock(80, 80, 5, 1, 6, 0.25),
            MBConvBlock(80, 112, 5, 2, 6, 0.25),
            MBConvBlock(112, 112, 5, 1, 6, 0.25),
            MBConvBlock(112, 112, 5, 1, 6, 0.25),
            MBConvBlock(112, 192, 5, 1, 6, 0.25),
            MBConvBlock(192, 192, 5, 1, 6, 0.25),
            MBConvBlock(192, 192, 5, 1, 6, 0.25),
            MBConvBlock(192, 192, 5, 1, 6, 0.25),
            MBConvBlock(192, 320, 3, 2, 6, 0.25),
            MBConvBlock(320, 1280, 1, 1, 6, 0.25),
        )
        self.head = nn.Sequential(
            nn.AdaptiveAvgPool2d(1),
            nn.Conv2d(1280, num_classes, 1, bias=True)
        )

    def forward(self, x):
        x = self.stem(x)
        x = self.blocks(x)
        x = self.head(x)
        x = torch.flatten(x, 1)
        return x
```

Lampiran 9. Potongan Kode Model Klasifikasi *MobileNetV3*

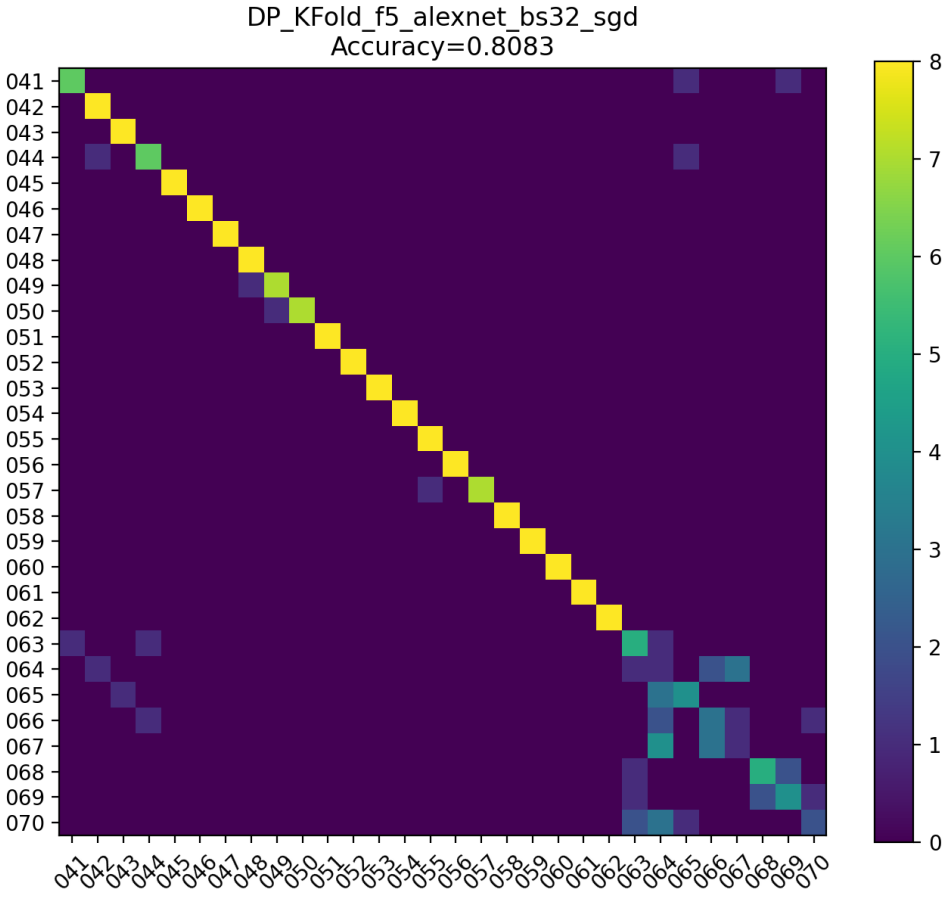
```
class MobileNetSmall(nn.Module):
    def __init__(self, num_classes=1000):
        super(MobileNetSmall, self).__init__()
        self.conv1 = nn.Sequential(
            nn.Conv2d(3, 16, 3, stride=2, padding=1,
bias=False),
            nn.BatchNorm2d(16),
            HardSwish()
        )
        self.bottlenecks = nn.Sequential(
            Bottleneck(16, 16, stride=2, t=1,
use_se=True, activation='RE'),
            Bottleneck(16, 16, stride=2, t=4,
activation='RE'),
            Bottleneck(16, 24, stride=1, t=3,
activation='RE'),
            Bottleneck(24, 24, stride=2, t=3,
use_se=True, activation='RE'),
            Bottleneck(24, 40, stride=1, t=3,
activation='HS', use_se=True),
            Bottleneck(40, 40, stride=1, t=3,
activation='HS', use_se=True),
            Bottleneck(40, 40, stride=1, t=6,
activation='HS', use_se=True),
            Bottleneck(40, 48, stride=1, t=2.5,
activation='HS', use_se=True),
            Bottleneck(48, 48, stride=1, t=2.3,
activation='HS', use_se=True),
            Bottleneck(48,96, stride=1, t=2.3,
activation='HS', use_se=True),
            Bottleneck(96, 96, stride=1, t=6,
activation='HS', use_se=True),
            Bottleneck(96, 96, stride=1, t=6,
activation='HS', use_se=True),
        )
        self.avgpool = nn.AdaptiveAvgPool2d(1)
        self.conv2 = nn.Sequential(
            nn.Conv2d(96, 576, 1, bias=False),
            nn.BatchNorm2d(576),
            HardSwish()
        )
        self.conv3 = nn.Sequential(
            nn.Conv2d(576, 1024, 1, bias=False),
            nn.BatchNorm2d(1024)
        )
        self.classifier = nn.Linear(1024, num_classes)
```

```
    def forward(self, x):
        x = self.conv1(x)
        x = self.bottlenecks(x)
        x = self.avgpool(x)
        x = self.conv2(x)
        x = self.conv3(x)
        x = torch.flatten(x, 1)
        x = self.classifier(x)
        return x
```

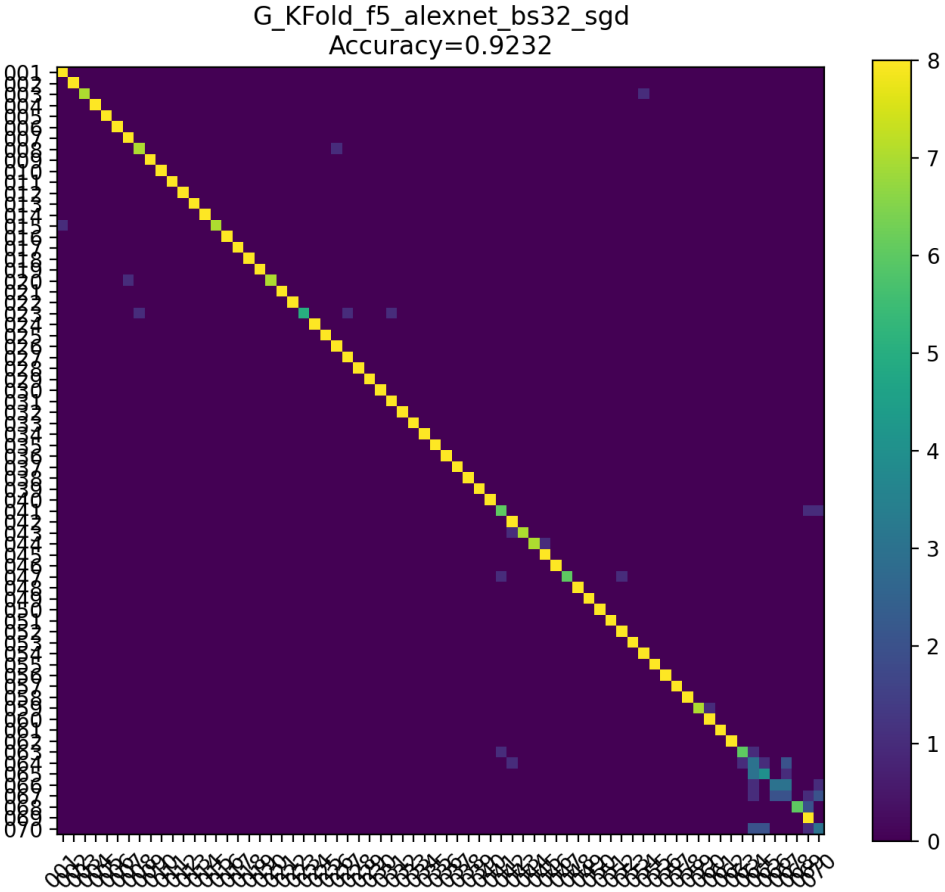
Lampiran 10. *Confusion Matrix AlexNet* pada Validasi Klasifikasi *Dataset BMPD*



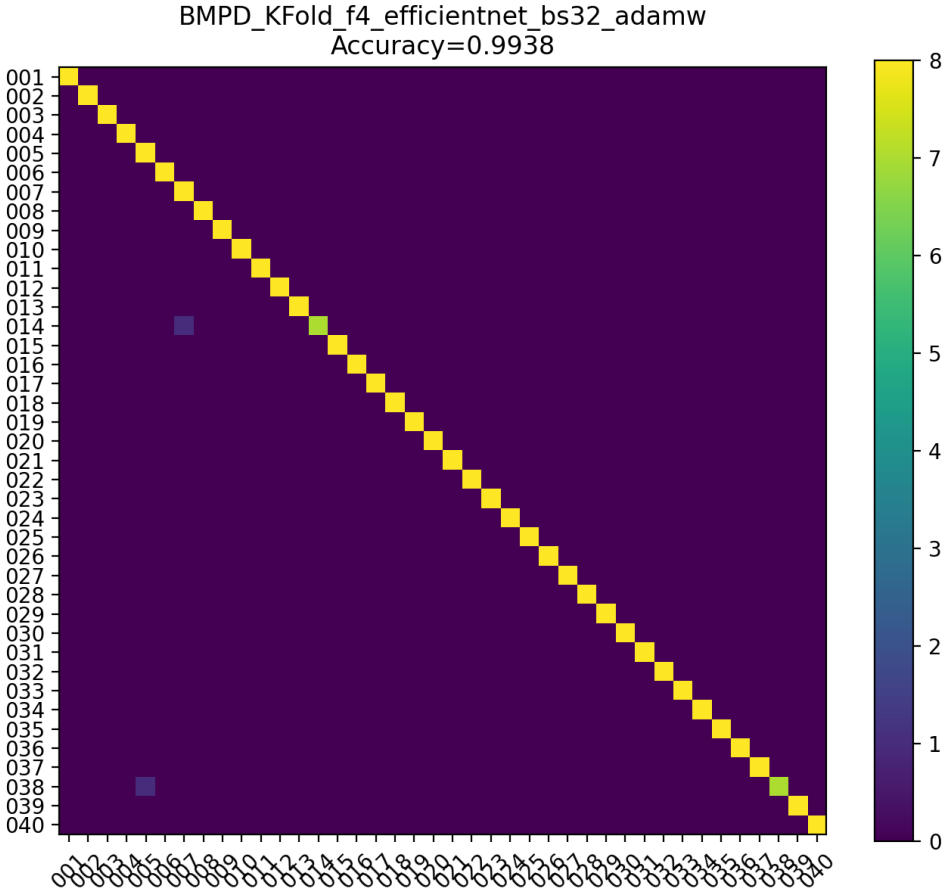
Lampiran 11. *Confusion Matrix AlexNet* pada Validasi Klasifikasi *Dataset* DP



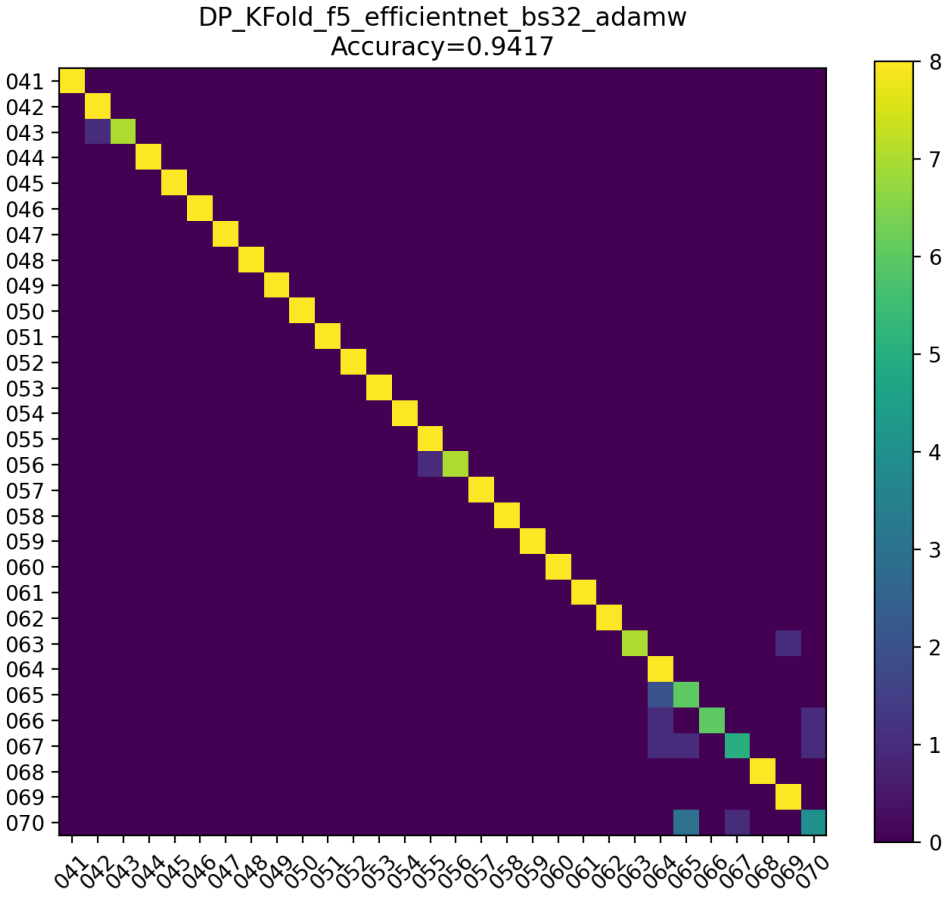
Lampiran 12. *Confusion Matrix AlexNet* pada Validasi Klasifikasi *Dataset* Gabungan



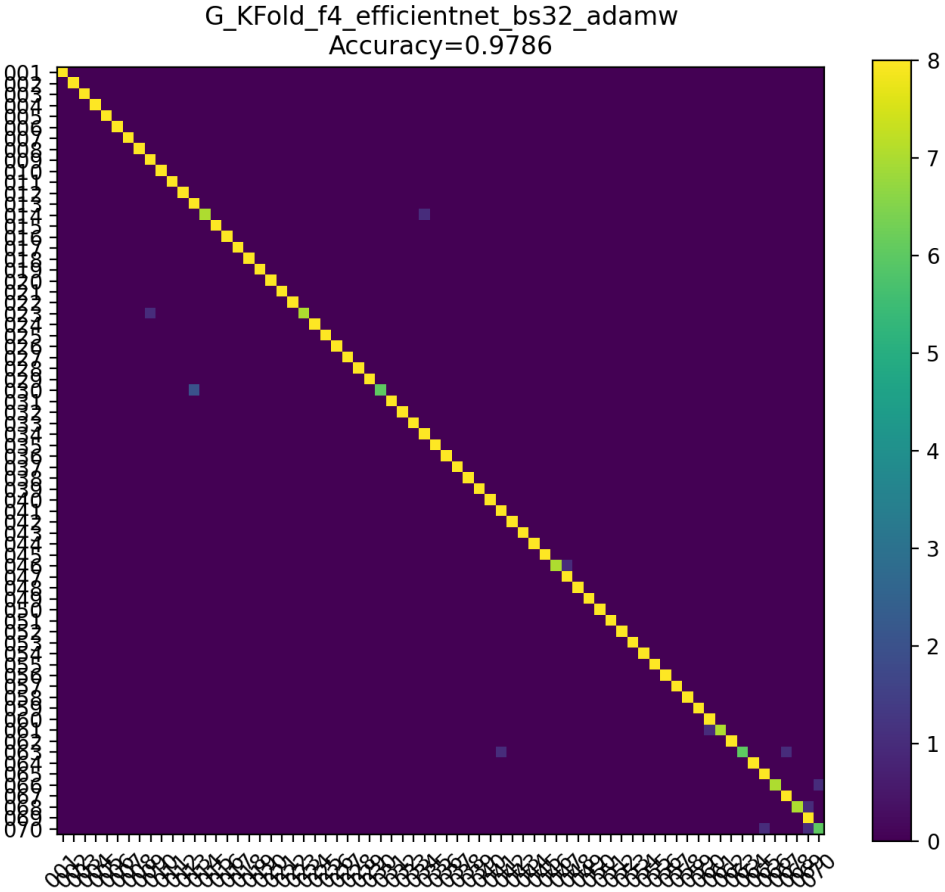
Lampiran 13. *Confusion Matrix EfficientNet* pada Validasi Klasifikasi *Dataset* BMPD



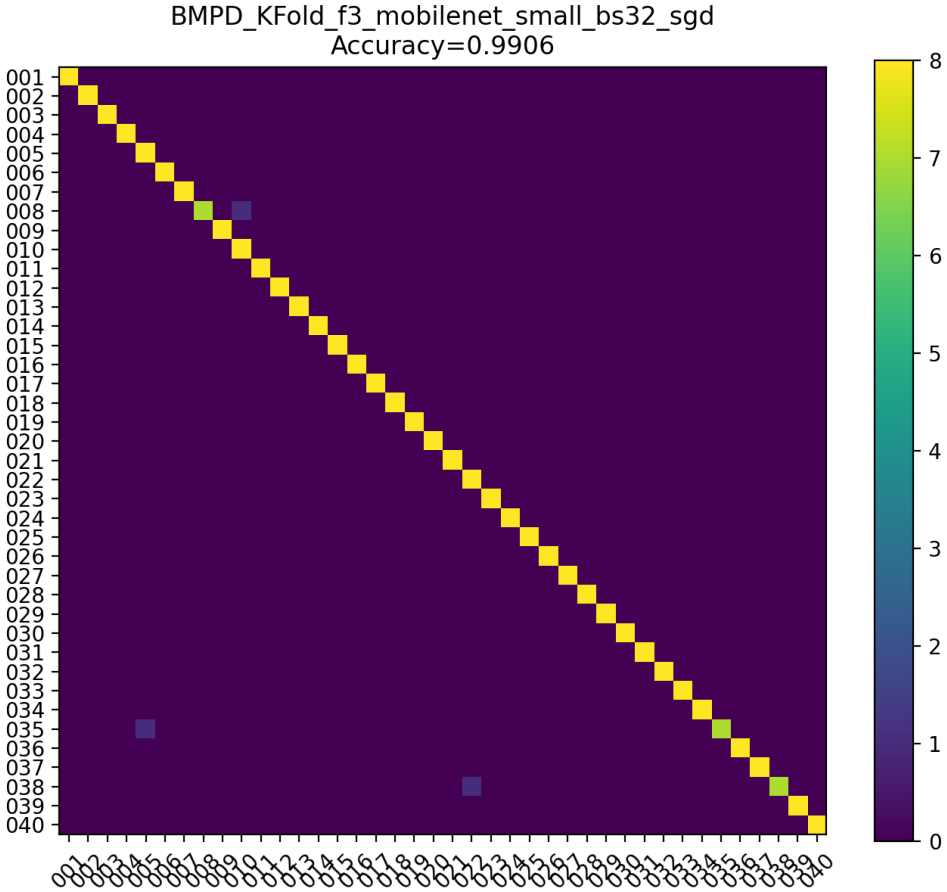
Lampiran 14. *Confusion Matrix EfficientNet* pada Validasi Klasifikasi *Dataset* DP



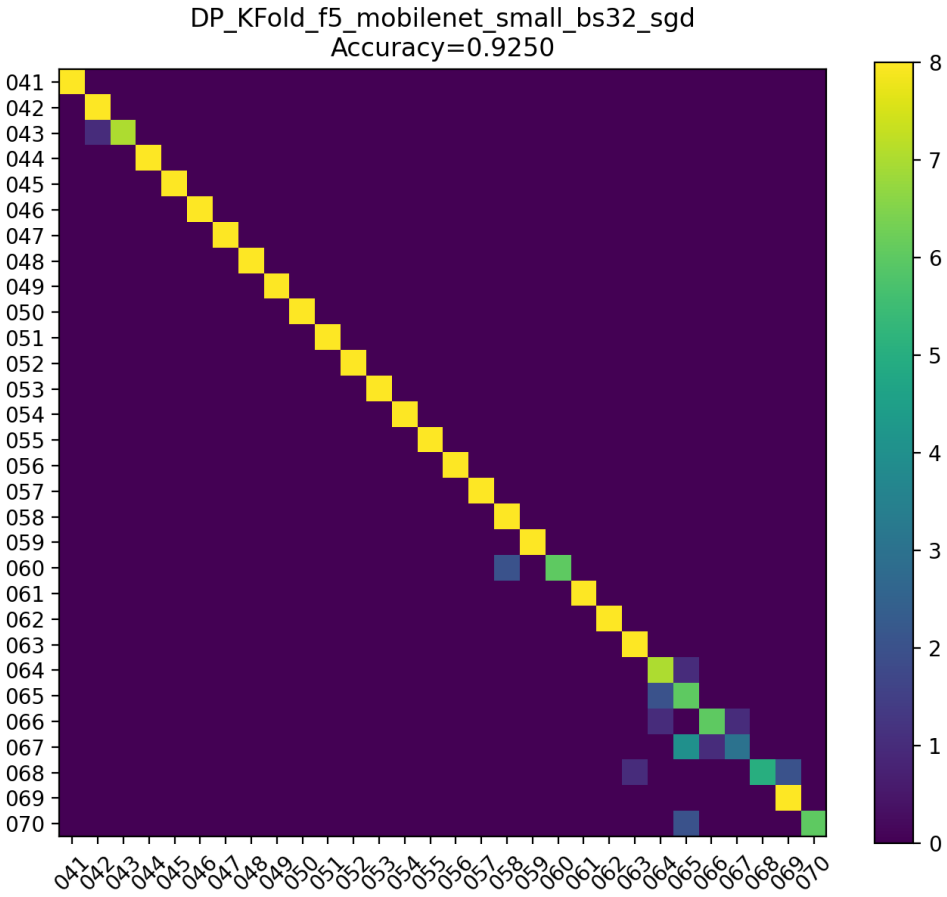
Lampiran 15. *Confusion Matrix EfficientNe*t pada Validasi Klasifikasi *Dataset* Gabungan



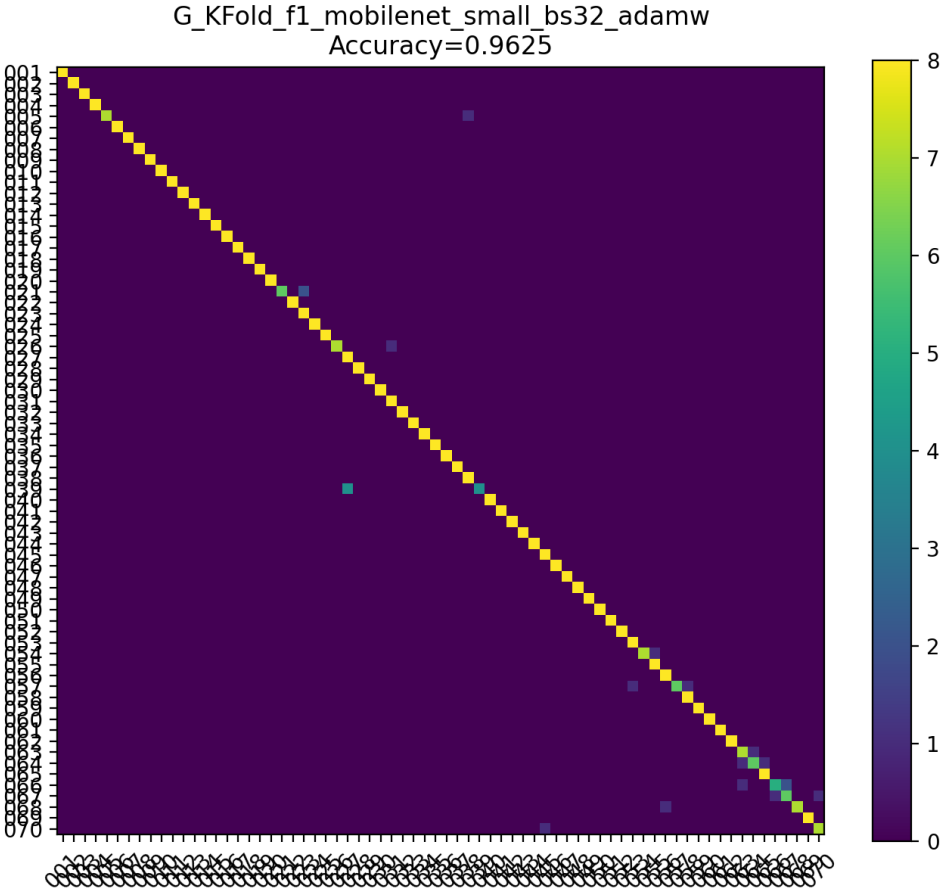
Lampiran 16. *Confusion Matrix MobileNetV3* pada Validasi Klasifikasi *Dataset* BMPD



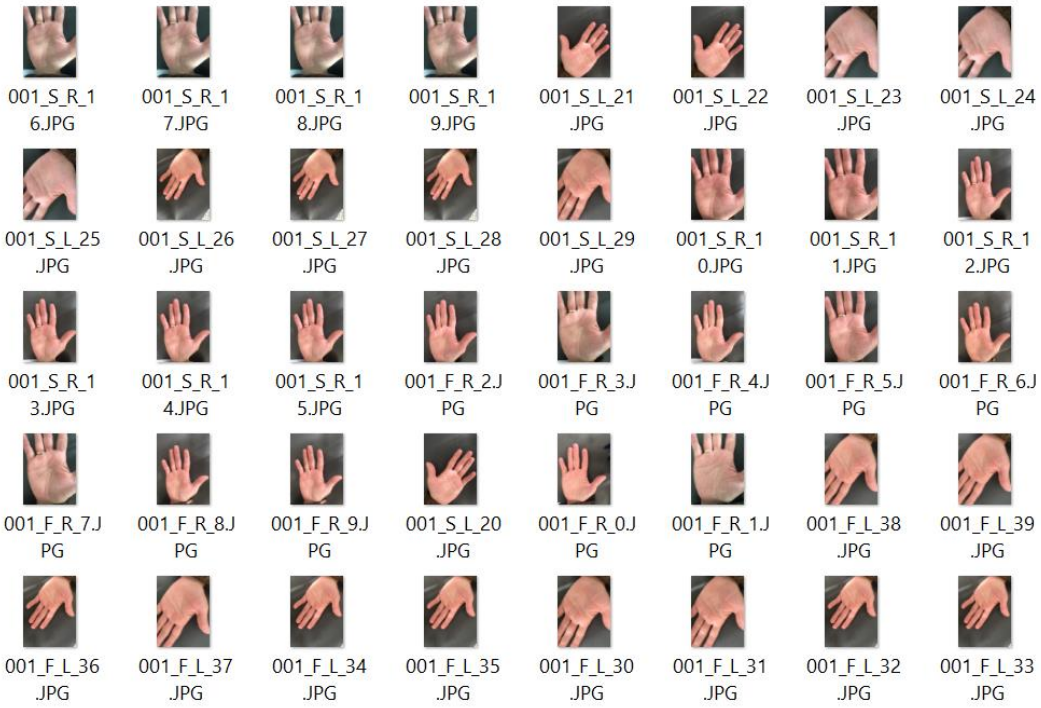
Lampiran 17. *Confusion Matrix MobileNetV3* pada Validasi Klasifikasi *Dataset* DP



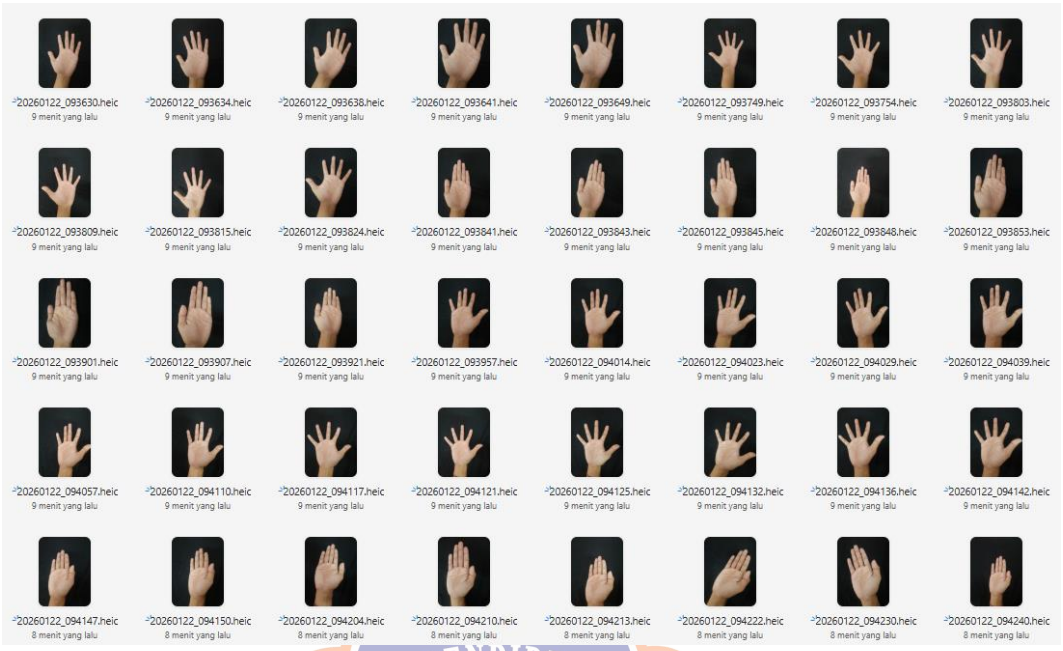
Lampiran 18. *Confusion Matrix MobileNetV3* pada Validasi Klasifikasi *Dataset* Gabungan



Lampiran 19. Contoh Gambar dari *Dataset* BMPD



Lampiran 20. Contoh Gambar dari *Dataset* Pribadi (DP)



RIWAYAT HIDUP

Komang Ratna Mutu Manikam lahir di Bukti, pada tanggal 17 September 2004. Penulis beralamat di Desa Bukti, Kecamatan Kubutambahan, Kabupaten Buleleng, Provinsi Bali.

Penulis menyelesaikan pendidikan dasar di SD Negeri 3 Bukti pada tahun 2016. Kemudian penulis melanjutkan pendidikan menengah pertama di SMP Negeri 1 Kubutambahan dan lulus pada tahun 2019. Kemudian melanjutkan pendidikan menengah atas di SMA Negeri 1 Sawan dan lulus di tahun 2022. Pada tahun yang sama, melanjutkan ke perguruan tinggi Universitas Pendidikan Ganesha pada Program Studi S1 Ilmu Komputer. Pada semester genap tahun 2026 penulis telah menyelesaikan Skripsi yang berjudul "Pengenalan Citra Telapak Tangan Menggunakan *Convolutional Neural Network* dengan Arsitektur *AlexNet, EfficientNet,* dan *MobileNetV3"*. Selanjutnya pada semester ganjil tahun 2026 penulis telah resmi lulus dari Universitas Pendidikan Ganesha.