LAMPIRAN-LAMPIRAN

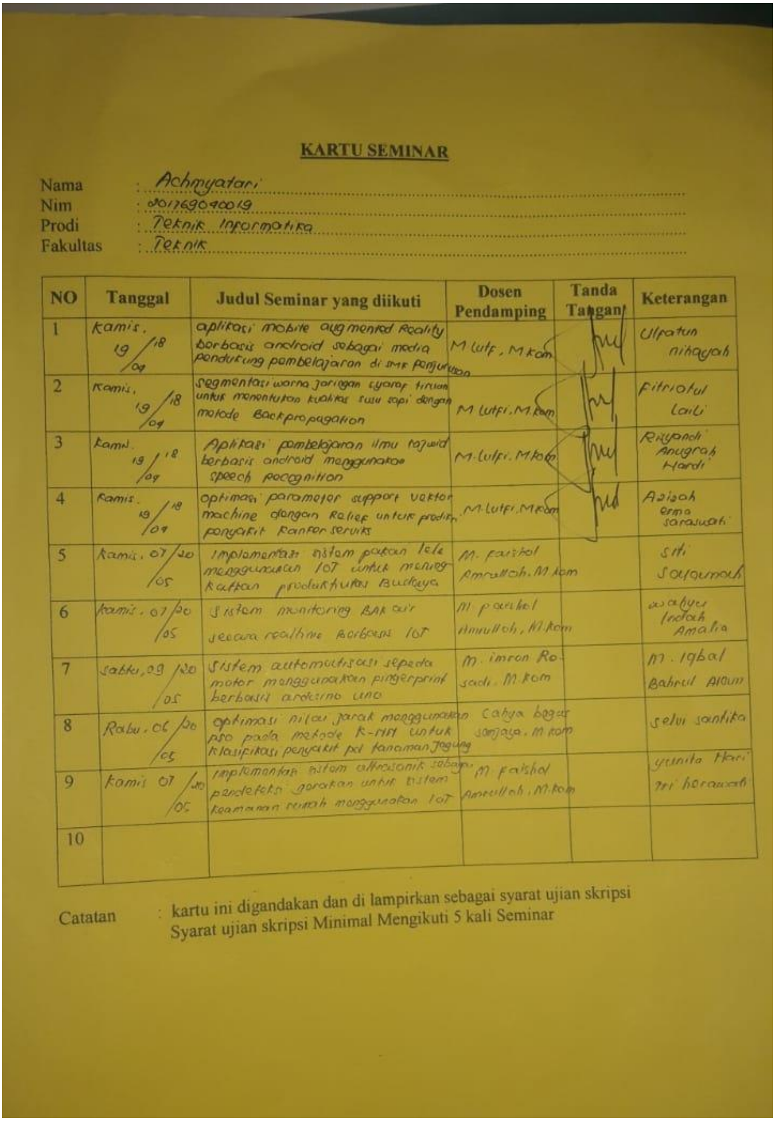
Lampiran 1. Daftar Riwayat Hidup

DATA PRIBADI		
Nama	: Achmyatari	
NIM	: 201769040019	
Tempat,tanggal lahir	:Banyuwangi, 28 Januari 1999	
Jenis Kelamin	: Perempuan	
Agama	: Islam	
Perguruan Tinggi	:Universitas yudharta Pasuruan	
Fakultas	: Teknik	
Program Studi	: Teknik Informatika	
Email	: achmyatari65@gmail.com	
Alamat	:Desa Karangsari, Kecamatan Sempu, Banyuwangi	

RIWAYAT PENDIDIKAN

• SD 1 Tumbak Bayuh	2005-2011
• SMP Alam Banyuwangi Islamic School	2011-2014
• MA Darut Taqwa	2014-2017
• Universitas Yudharta Pasuruan	2017-2021

Lampiran 2. Kartu Seminar



KARTU SEMINAR

Nama : Achmyatari

Nim : 201769040019

Prodi : Teknik Informatika

Fakultas : Teknik

NO	Tanggal	Judul Seminar yang diikuti	Dosen Pendamping	Tanda Tangan	Keterangan
1	kamis, 19/18 /09	aplikasi mobile augmented Reality berbasis android sebagai media pendukung pembelajaran di SMK penjurusan	M Lutf, M kom		Ulfatun nihayah
2	kamis, 19/18 /09	segmentasi warna jaringan syaraf tiruan untuk menentukan kualitas susu sapi dengan metode Backpropagation	M Lutfi, M.kom		Fitriotul Laili
3	kamis. 19/18 /09	Aplikasi pembelajaran ilmu tajwid berbasis android menggunakan speech recognition	M.Lutfi. M.kom		Rizyandi Anugrah Hardi
4	kamis. 19/18 /09	optimasi parameter support vektor machine dengan Relief untuk prediksi penyakit kanker serviks	M.Lutfi.M.kom		Azizah ermo saraswati
5	kamis. 07/20 /05	implementasi sistem pakan lele menggunakan IoT untuk meningkatkan produktivitas Budidaya	M. Faishol Amrullah. M.kom		Siti Salamah
6	kamis. 07/20 /05	sistem monitoring BAK air secara realtime Berbasis IoT	M. Faishol Amrullah, M.kom		wahyu Indah Amalia
7	sabtu, 09/20 /05	Sistem automatisasi sepeda motor menggunakan fingerprint berbasis arduino uno	M. imron Rosadi. M.kom		M. Iqbal Bahrul Alum
8	Rabu. 06/20 /05	optimasi nilai jarak menggunakan pso pada metode K-nn untuk klasifikasi penyakit pd tanaman Jagung	Cahya bagus sanjaya. M.kom		selvi santika
9	kamis 07/20 /05	implementasi sistem ultrasonik sebagai pendeteksi gerakan untuk sistem keamanan rumah menggunakan IoT	M. Faishol Amrullah. M.kom		yunita Hari tri harawati
10					

Catatan : kartu ini digandakan dan di lampirkan sebagai syarat ujian skripsi
Syarat ujian skripsi Minimal Mengikuti 5 kali Seminar

Lampiran 3. Kartu Bimbingan Skripsi

LEMBAR BIMBINGAN SKRIPSI

Nama	: Achmyatari
NIM	: 201769040019
Jurusan	: Teknik Informatika
Konsentrasi	: Jaringan
Judul	: PERBANDINGAN ARSITEKTUR *LENET* DAN *GOOGLENET* DALAM KLASIFIKASI *DIABETIC RETINOPATHY* PADA CITRA RETINA FUNDUS

Hari	Tanggal	BAB	Materi Bimbingan	Tanda Tangan
Rabu	24 Februari 2021	Penyusunan dan pengajuan judul	Penyusunan Dan Pengajuan Judul	
Sabtu	27 Februari 2021	Revisi Metode	Re-focussing Research Trends Topic	
Minggu	28 Februari 2021	Acc Judul &Pengajuan Bab 1	Progress Bab 1	

Rabu	03 Maret 2021	Revisi Bab 1 & Pengajuan Bab 2	Revisi Bab 1 dan Progress Bab 2	
Sabtu	06 Maret 2021	Progress Bab 2	Progress Bab 2	
Minggu	07 Maret 2021	Revisi Bab 2	Revisi Bab 2	
Rabu	10 Maret 2021	Analisa Dataset	Analisa Data uji	
Sabtu	13 Maret 2021	Analisa Dataset	Analisa Data Uji dan NormalisasiData	
Rabu	17 Maret 2021	Pengajuan Bab 3	Progress Bab 3	
Sabtu	20 Maret 2021	Progress Bab 3	Progress Bab 3	
Rabu	24 Maret 2021	Revisi Bab 3	Revisi Bab 3	
Sabtu	27 Maret 2021	Revisi Bab 3	Revisi Bab 3	
Sabtu	01 Mei 2021	Revisi Proposal Skripsi	Revisi Proposal Skripsi	

Rabu	26 Mei 2021	Revisi Proposal Skripsi	Revisi Proposal Skripsi	
Sabtu	29 Mei 2021	Implementasi Algoritma	Analisis Pra Implementasi Algoritma	
Rabu	02 Juni 2021	Implementasi Algoritma + Analisa Preprocessing	Implementasi Algoritma + Analisa Preprocessing	
Minggu	06 Juni 2021	Penyusunan Source Code	Penyusunan Sourcecode Uji Dataset	
Sabtu	12 Juni 2021	Penyusunan Source Code	Penyusunan Sourcecode Uji Dataset	
Rabu	16 Juni 2021	Pengujian I Source Code	Pengujian I Sourcecode Uji Dataset	
Minggu	20 Juni 2021	Pengujian II Source Code	Pengujian II Sourcecode Uji Dataset	
Rabu	07 Juli	Cek Hasil Uji	Cek Hasil Uji	

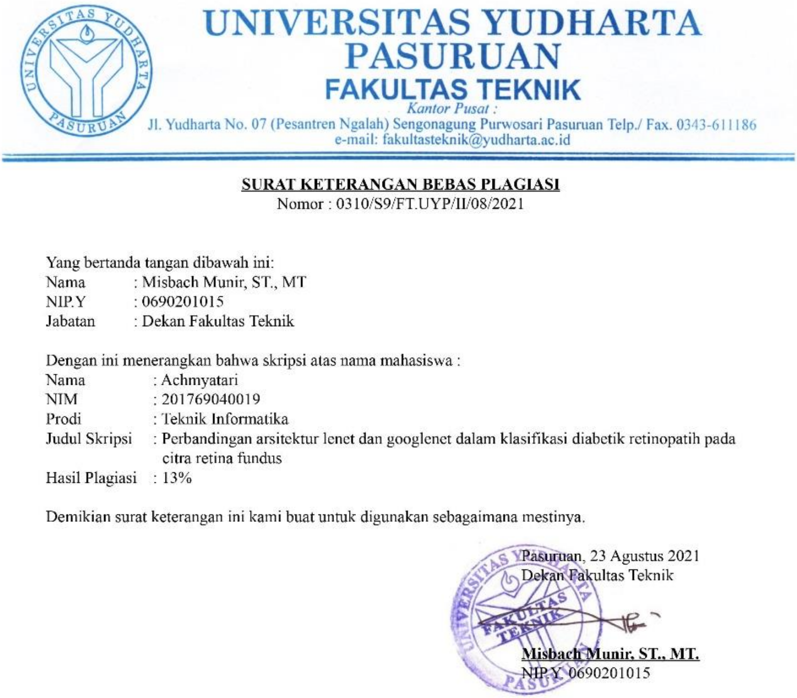
	2021	Algoritma	Algoritma	
Minggu	11 Juli 2021	Penyusunan Proposal & Cek Uji Algoritma	Penyusunan Proposal & Cek Uji Algoritma	
Sabtu	17 Juli 2021	Revisi Laporan Skripsi	Revisi Laporan Skripsi	
Selasa	20 Juli 2021	Revisi Laporan Skripsi	Revisi Laporan Skripsi	

Pasuruan, 24 Agustus 2021

Pembimbing,

Moch. Lutfi, S.Kom., M.Kom.,
NIP.Y 0691603004

Lampiran 4. Lembar Bebas Plagiasi



UNIVERSITAS YUDHARTA
PASURUAN
FAKULTAS TEKNIK
Kantor Pusat :
Jl. Yudharta No. 07 (Pesantren Ngalah) Sengonagung Purwosari Pasuruan Telp./ Fax. 0343-611186
e-mail: fakultasteknik@yudharta.ac.id

SURAT KETERANGAN BEBAS PLAGIASI
Nomor : 0310/S9/FT.UYP/II/08/2021

Yang bertanda tangan dibawah ini:

Nama	: Misbach Munir, ST., MT
NIP.Y	: 0690201015
Jabatan	: Dekan Fakultas Teknik

Dengan ini menerangkan bahwa skripsi atas nama mahasiswa :

Nama	: Achmyatari
NIM	: 201769040019
Prodi	: Teknik Informatika
Judul Skripsi	: Perbandingan arsitektur lenet dan googlenet dalam klasifikasi diabetik retinopatih pada citra retina fundus
Hasil Plagiasi	: 13%

Demikian surat keterangan ini kami buat untuk digunakan sebagaimana mestinya.

Pasuruan, 23 Agustus 2021
Dekan Fakultas Teknik

UNIVERSITAS YUDHARTA
FAKULTAS TEKNIK
PASURUAN

Misbach Munir, ST., MT.
NIP.Y. 0690201015

Lampiran 5. Source Code Keseluruhan

```
!pip install tensorFlow-gpu
!pip install keras
import tensorflow as tf
tf.__version__
from google.colab import drive
drive.mount('/content/drive')
mypath= '/content/drive/MyDrive/fundus'
import os
import pandas as pd
file_name = []
tag = []
full_path = []
for path, subdirs, files in os.walk(mypath):
    for name in files:

      full_path.append(os.path.join(path, name))
        tag.append(path.split('/')[-1])
        file_name.append(name)

# memasukan variabel yang sudah dikumpulkan pada looping di atas menjadi sebuah dataframe agar rapih
df = pd.DataFrame({"path":full_path,'file_name':file_name,"tag":tag})
df.groupby(['tag']).size()

#cek sample datanya
df.head()
```

```
#variabel yang digunakan pada pemisahan data ini
X= df['path']
y= df['tag']

# split dataset awal menjadi data train dan test
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.20, random_state=100)
df_tr = pd.DataFrame({'path':X_train
              ,'tag':y_train
             ,'set':'train'})

df_te = pd.DataFrame({'path':X_test
              ,'tag':y_test
             ,'set':'test'})
print('train size', len(df_tr))
print('test size', len(df_te))

# melihat proporsi pada masing masing set apakah sudah ok atau masih ada yang ingin diubah
df_all = df_tr.append([df_te]).reset_index(drop=1)\

print('===================================================== \n')
print(df_all.groupby(['set','tag']).size(),'\n')
print('=====================================================
================ \n')

#cek sample datanya
df_all.sample(3)
from tensorflow.compat.v1 import ConfigProto
from tensorflow.compat.v1 import InteractiveSession

config = ConfigProto()
config.gpu_options.per_process_gpu_memory_fraction = 0.5
config.gpu_options.allow_growth = True
session = InteractiveSession(config=config)
from keras.models import Sequential
from keras.layers import Dense, Flatten, Conv2D, MaxPooling2D
# Helper libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import cv2
import glob
from sklearn.model_selection import train_test_split
from sklearn import preprocessing

#Build The CNN
#model CNN dalam sistem
model = Sequential() #Create the architecture
```

```
model.add(Conv2D(64, (5, 5), activation='relu',
input_shape=(128,128,3)))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(64, (5, 5), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Conv2D(32, (5, 5), activation='relu'))
model.add(Flatten())
model.add(Dense(512, activation='relu'))
model.add(Dense(128, activation='relu'))
model.add(Dense(3, activation='softmax'))

# import the libraries as shown below
import tensorflow as tf
from tensorflow.keras.layers import Input, Lambda, Dense, Flatten
from tensorflow.keras.models import Model
from tensorflow.keras.applications.inception_v3 import InceptionV3
#from keras.applications.vgg16 import VGG16
from tensorflow.keras.applications.inception_v3 import preprocess_input
from tensorflow.keras.preprocessing import image
from tensorflow.keras.preprocessing.image import ImageDataGenerator,load_img
from tensorflow.keras.models import Sequential
import numpy as np
```

```
from glob import glob
#import matplotlib.pyplot as plt

import tensorflow as tf
from tensorflow.keras.optimizers import RMSprop
from sklearn.model_selection import train_test_split
from tensorflow.keras.preprocessing.image import ImageDataGenerator
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
%matplotlib inline
import pandas as pd
from tensorflow.keras.preprocessing.image import img_to_array, load_img
import random

import tensorflow as tf
import tensorflow.keras.layers as Layers
import tensorflow.keras.activations as Activations
import tensorflow.keras.models as Models
import tensorflow.keras.optimizers as Optimizers
import tensorflow.keras.metrics as Metrics
import tensorflow.keras.utils as Utils
import pandas as pd
import tensorflow.keras.backend as K
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import i
mage
from tensorflow.keras import regularizers
from tensorflow.keras.applications.inception
_v3 import InceptionV3
from tensorflow.keras.callbacks import Model
Checkpoint, CSVLogger
from tensorflow.keras.optimizers import SGD
from tensorflow.keras.regularizers import l2

from tensorflow import keras
from tensorflow.keras import models
from tensorflow.keras.applications.inception
_v3 import preprocess_input

#from tensorflow.keras.models import Models,
 Sequential
from tensorflow.keras.layers import GlobalAv
eragePooling2D
from tensorflow.keras.layers import Dense, I
nput, Conv2D, Flatten, Dropout, MaxPooling2D
from tensorflow.keras.preprocessing.image im
port ImageDataGenerator
from tensorflow.keras.utils import get_file

import numpy as np
import matplotlib.pyplot as plt
import matplotlib.gridspec as gridspec

import PIL.Image as Image
from sklearn.utils import shuffle
```

```python
from sklearn.metrics import confusion_matrix as CM
from IPython.display import  SVG


import os
# re-size all the images to this
IMAGE_SIZE = [150, 150]

valid_path = '/content/drive/MyDrive/fundus/test'
valid_path = '/content/drive/MyDrive/fundus/train'

# Here we will be using imagenet weights
inception = InceptionV3(input_shape=IMAGE_SIZE + [3], weights='imagenet', include_top=False)
class InceptionV3():
    @staticmethod
    def build(numChannels, imgRows, imgCols, numClasses,  pooling= "max", activation= "relu"):
        # initialize the model
        model = Sequential()
        inputShape = (imgRows, imgCols, numChannels)

        # add first set of layers: Conv -> Activation -> Pool
        model.add(Conv2D(filters= 6, kernel_
size= 5, input_shape= inputShape))
        model.add(Activation(activation))

        if pooling == "max":
           model.add(MaxPooling2D(pool_size=
 (3, 3), strides= (2, 2)))
        else:
            model.add(AveragePooling2D(pool_
size= (7, 7), strides= (2, 2)))

        # add second set of layers: Conv -
> Activation -> Pool
        model.add(Conv2D(filters= 16, kernel
_size= 5,))
        model.add(Activation(activation))

        if pooling == "avg":
           model.add(AveragePooling2D(pool_s
ize=(7, 7), strides=(2, 2)))
        else:
           model.add(MaxPooling2D(pool_size=
(3, 3), strides=(2, 2)))

        # Flatten -> FC 120 -> Dropout -
> Activation
        model.add(Flatten())
        model.add(Dense(64))
        model.add(Dropout(0.6))
        model.add(Activation(activation))
        # FC 84 -> Dropout -> Activation
        model.add(Dense(32))
        model.add(Dropout(0.6))
        model.add(Activation(activation))

        # FC 4-> Softmax
        model.add(Dense(numClasses))
        model.add(Activation("softmax"))

        return model
# don't train existing weights
for layer in inception.layers:
    layer.trainable = False
# useful for getting number of output classes
folders = glob('/content/drive/MyDrive/fundus/train/*')
folders = glob('/content/drive/MyDrive/fundus/test/*')
# our layers - you can add more if you want
x = Flatten()(inception.output)
prediction = Dense(len(folders), activation='softmax')(x)

# create a model object
model = Model(inputs=inception.input, outputs=prediction)

# view the structure of the model
model.summary()
```

```
# tell the model what cost and optimization method to use
#training data
model.compile(
  loss='categorical_crossentropy',
  optimizer='adam',
  metrics=['accuracy']
)
# Use the Image Data Generator to import the images from the dataset
from tensorflow.keras.preprocessing.image import ImageDataGenerator

train_datagen = ImageDataGenerator(rescale = 1./255,
    shear_range = 0.2,
    zoom_range = 0.2,
    horizontal_flip = True)

test_datagen = ImageDataGenerator(rescale = 1./255,
    shear_range = 0.2,
    zoom_range = 0.2,
    horizontal_flip = True)

# Make sure you provide the same target size as initialied for the image size
training_set = train_datagen.flow_from_directory('/content/drive/MyDrive/fundus/train',

     target_size = (150, 150),
     batch_size = 8,

     class_mode = 'categorical')
test_set = test_datagen.flow_from_directory(
'/content/drive/MyDrive/fundus/test',

target_size = (150, 150),

batch_size = 8,

class_mode = 'categorical')
# fit the model
# Run the cell. It will take some time to ex
ecute
r = model.fit_generator(
  training_set,
  validation_data=test_set,
  epochs=10,
  steps_per_epoch=len(training_set),
  validation_steps=len(test_set)
)
import matplotlib.pyplot as plt

# plot the loss
plt.plot(r.history['loss'], label='train los
s')
plt.plot(r.history['val_loss'], label='val l
oss')
plt.legend()
plt.show()
```

```python
plt.savefig('LossVal_loss')

# plot the accuracy
plt.plot(r.history['accuracy'], label='train acc')
plt.plot(r.history['val_accuracy'], label='val acc')
plt.legend()
plt.show()
plt.savefig('AccVal_acc')

# save it as a h5 file
from tensorflow.keras.models import load_model

model.save('model_inception.h5')
y_pred = model.predict(test_set)
import numpy as np
y_pred = np.argmax(y_pred, axis=1)
test_set.labels
from sklearn import metrics
from keras import metrics

model.compile(loss='mean_squared_error', optimizer='sgd',
              metrics=[metrics.mae,
                       metrics.categorical_accuracy])

from sklearn import metrics
from sklearn.metrics import confusion_matrix
```

```python
conf = metrics.confusion_matrix(test_set.labels, y_pred)
conf
import itertools
classes = [0, 1, 2]

# plot confusion matrix
plt.imshow(conf, interpolation='nearest', cmap=plt.cm.Blues)
plt.title("Confusion Matrix")
plt.colorbar()
tick_marks = np.arange(len(classes))
plt.xticks(tick_marks, classes)
plt.yticks(tick_marks, classes)

fmt = 'd'
thresh = conf.max() / 2.
for i, j in itertools.product(range(conf.shape[0]), range(conf.shape[1])):
    plt.text(j, i, format(conf[i, j], fmt),
             horizontalalignment="center",
             color="black" if conf[i, j] > thresh else "black")

plt.tight_layout()
plt.ylabel('True label')
plt.xlabel('Predicted label')
class LeNet():
    @staticmethod
    def build(numChannels, imgRows, imgCols, numClasses,  pooling= "max", activation= "relu"):
        # initialize the model
        model = Sequential()
        inputShape = (imgRows, imgCols, numChannels)

        # add first set of layers: Conv -> Activation -> Pool
        model.add(Conv2D(filters= 6, kernel_size= 5, input_shape= inputShape))
        model.add(Activation(activation))

        if pooling == "max":
           model.add(MaxPooling2D(pool_size= (2, 2), strides= (2, 2)))
        else:
            model.add(AveragePooling2D(pool_size= (2, 2), strides= (2, 2)))

        # add second set of layers: Conv -> Activation -> Pool
        #model.add(Conv2D(filters= 16, kernel_size= 5,))
        #model.add(Activation(activation))

        #if pooling == "avg":
        #   model.add(AveragePooling2D(pool_size=(2, 2), strides=(2, 2)))
       # else:
        #   model.add(MaxPooling2D(pool_size
=(2, 2), strides=(2, 2)))

        # Flatten -> FC 120 -> Dropout -
> Activation
        model.add(Flatten())
        model.add(Dense(64))
        model.add(Dropout(0.5))
        model.add(Activation(activation))

        # FC 84 -> Dropout -> Activation
        model.add(Dense(32))
        model.add(Dropout(0.5))
        model.add(Activation(activation))

        # FC 4-> Softmax
        model.add(Dense(numClasses))
        model.add(Activation("softmax"))

        return model
```