DAFTAR PUSTAKA

Chatzimina, M. E., Papadaki, H. A., Pontikoglou, C., & Tsiknakis, M. (2024). A Comparative Sentiment Analysis of Greek Clinical Conversations Using BERT, RoBERTa, GPT-2, and XLNet. *Bioengineering*, *11*(6), Article 6. https://doi.org/10.3390/bioengineering11060521

Gogineni, H., Aranda, J. P., & Garavalia, L. S. (2019). Designing professional program instruction to align with students' cognitive processing. *Currents in Pharmacy Teaching and Learning*, *11*(2), 160–165. https://doi.org/10.1016/j.cptl.2018.11.015

González-Nóvoa, J. A., Campanioni, S., Busto, L., Fariña, J., Rodríguez-Andina, J. J., Vila, D., Íñiguez, A., & Veiga, C. (2023). Improving Intensive Care Unit Early Readmission Prediction Using Optimized and Explainable Machine Learning. *International Journal of Environmental Research and Public Health*, *20*(4), Article 4. https://doi.org/10.3390/ijerph20043455

Han, T., Zhang, Z., Ren, M., Dong, C., Jiang, X., & Zhuang, Q. (2023). Text Emotion Recognition Based on XLNet-BiGRU-Att. *Electronics*, *12*(12), Article 12. https://doi.org/10.3390/electronics12122704

Huang, K., Altosaar, J., & Ranganath, R. (2020). *ClinicalBERT: Modeling Clinical Notes and Predicting Hospital Readmission* (arXiv:1904.05342). arXiv. http://arxiv.org/abs/1904.05342

Johnson, A., Pollard, T., & Mark, R. (2015). *MIMIC-III Clinical Database* (Version 1.4) [Dataset]. PhysioNet. https://doi.org/10.13026/C2XW26

Lorkowski, J., & Pokorski, M. (2022). Medical Records: A Historical Narrative. *Biomedicines*, *10*(10), Article 10. https://doi.org/10.3390/biomedicines10102594

Minaee, S., Kalchbrenner, N., Cambria, E., Nikzad, N., Chenaghlu, M., & Gao, J. (2021). *Deep Learning Based Text Classification: A Comprehensive Review* (arXiv:2004.03705). arXiv. https://doi.org/10.48550/arXiv.2004.03705

Necula, S.-C., Dumitriu, F., & Greavu-Șerban, V. (2024). A Systematic Literature Review on Using Natural Language Processing in Software Requirements Engineering. *Electronics*, *13*(11), Article 11. https://doi.org/10.3390/electronics13112055

Pencatatan SOAP Rekam Medis dan Contoh Penggunaannya—eClinic. (2023, November 17). https://www.eclinic.id/pencatatan-soap-rekam-medis/

Podder, V., Lew, V., & Ghassemzadeh, S. (2024). SOAP Notes. In *StatPearls*. StatPearls Publishing. http://www.ncbi.nlm.nih.gov/books/NBK482263/

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., & Liu, P. J. (2023). *Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer* (arXiv:1910.10683). arXiv. https://doi.org/10.48550/arXiv.1910.10683

Rhazzafe, S., Caraffini, F., Colreavy-Donnelly, S., Dhassi, Y., Kuhn, S., & Nikolov, N. S. (2024). Hybrid Summarization of Medical Records for Predicting Length

of Stay in the Intensive Care Unit. *Applied Sciences*, *14*(13), Article 13. https://doi.org/10.3390/app14135809

Sarzaeim, P., Mahmoud, Q. H., Azim, A., Bauer, G., & Bowles, I. (2023). A Systematic Review of Using Machine Learning and Natural Language Processing in Smart Policing. *Computers*, *12*(12), Article 12. https://doi.org/10.3390/computers12120255

Shi, F., Kai, S., Zheng, J., & Zhong, Y. (2022). XLNet-Based Prediction Model for CVSS Metric Values. *Applied Sciences*, *12*(18), Article 18. https://doi.org/10.3390/app12188983

Tavakolian, A., Rezaee, A., Hajati, F., & Uddin, S. (2023). Hospital Readmission and Length-of-Stay Prediction Using an Optimized Hybrid Deep Model. *Future Internet*, *15*(9), Article 9. https://doi.org/10.3390/fi15090304

Torkman, R., Ghapanchi, A. H., & Ghanbarzadeh, R. (2024). A Framework for Antecedents to Health Information Systems Uptake by Healthcare Professionals: An Exploratory Study of Electronic Medical Records. *Informatics*, *11*(3), Article 3. https://doi.org/10.3390/informatics11030044

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R., & Le, Q. V. (2020). *XLNet: Generalized Autoregressive Pretraining for Language Understanding* (arXiv:1906.08237). arXiv. https://doi.org/10.48550/arXiv.1906.08237

Zhao, D., Chen, X., & Chen, Y. (2024). Named Entity Recognition for Chinese Texts on Marine Coral Reef Ecosystems Based on the BERT-BiGRU-Att-CRF Model. *Applied Sciences*, *14*(13), Article 13. https://doi.org/10.3390/app14135743

LAMPIRAN

Lampiran 1. *Dataset* Penelitian: Kelas Positif

Positif:
Contoh 1:
Teks: lv cavity size. normal regional lv systolic function. hyperdynamic lvef >75%. mid-cavitary gradient. no vsd. right ventricle: normal rv chamber size and free wall motion. aorta: mildly dilated ascending aorta. aortic valve: three aortic valve leaflets. moderately thickened aortic valve leaflets. mild as (area 2-9cm2). trace ar. mitral valve: mildly thickened mitral valve leaflets. severe mitral annular calcification. mod functional ms due to mac. mild to moderate (+) mr. tricuspid valve: mildly thickened tricuspid valve leaflets. no ts. mild tr. indeterminate pa systolic pressure. pulmonic valve/pulmonary artery: no ps. pericardium: no pericardial effusion. conclusions the left atrium is normal in size. no atrial septal defect is seen by 2d or color doppler. the estimated right atrial pressure is 0-5 mmhg. there is mild symmetric left ventricular hypertrophy. the left ventricular cavity size is normal. regional left ventricular wall motion is normal. left ventricular systolic function is hyperdynamic (ef>75%). a mid-cavitary gradient is identified. there is no ventricular septal defect. right ventricular chamber size and free wall motion are normal. the ascending aorta is mildly dilated. there are three aortic valve leaflets. the aortic valve leaflets are moderately thickened. there is mild aortic valve stenosis (valve area 2-9cm2). trace aortic regurgitation is seen. the mitral valve leaflets are mildly thickened. there is severe mitral annular calcification. there is moderate functional mitral stenosis (mean gradient 11 mmhg) due to mitral annular calcification. mild to moderate (+) mitral regurgitation is seen. the tricuspid valve leaflets are mildly thickened. the pulmonary artery systolic pressure could not be determined
---
Contoh 2:
Teks: 4 mg tablet, sublingual sig: one (1) tablet sublingual as directed as needed for chest pain. outpatient lab work please check inr on monday and call results to doctor at. acetaminophen 325 mg tablet sig: 1-2 tablets po q6h (every 6 hours) as needed for pain. disp:*6 vils* refills:*0* discharge disposition: home with service facility: vna discharge diagnosis: ventricular tachycardia non st elevation myocardial infarction discharge condition: stable. discharge instructions: you had a dangerous heart rhythm called ventricular tachycardia and was started on sotolol, a medicine to prevent this rhythm. in addition, an internal defibrillator (icd) was placed that will shock you out of this rhythm. you cannot get the icd dressing wet for one week. no showers of baths. you may wash your hair in a sink. you are scheduled in the device clinic in 1 week, they will check the function of the icd and take off the dressing. no lifting more than 5 pounds with your left arm for 6 weeks, no lifting your left arm over your head for 6 weeks. you will be on antibiotics to prevent an infection at the icd site for 3 days. you also had a cardiac catheterization that showed extensive blockages in your coronary artery. your medicines were adjusted to help your heart function. medication changes: sotolol: to prevent ventricular tachycardia restart your coumadin at 2 mg, you will need to check your inr on monday. decrease your aspirin to 81mg, continue taking plavix.. please call doctor if your icd fires, if you have any redness, swelling, tenderness or bleeding at the icd site, if you have any chest pain, fevers, chills or trouble breathing. weigh yourself every morning, md if weight > 3 lbs in 1 day or 6 pounds in 3 days. adhere to 2 gm sodium diet: information was given to you about this at discharge.. followup instructions: cardiology: doctor phone:

Contoh 3:
Teks: date of birth: sex: m service: medicine allergies: pneumovax 23 / allopurinol / hydralazine attending: chief complaint: transferred from osh with abdominal pain major surgical or invasive procedure: ercp with stent; cholecystostomy; hemodialysis history of present illness: the pt is a 74m with hx of cad s/p cabg in, dm, gout, ckd who presented to yesterday with ruq pain and noted to have t=4 with wbc of 13k (17k today), elevated ast/alt/alk phos. u/s w/ gallstones. noted to have elevated troponins although ekg unrevealing. was given cefoxitin and iv hep. is pain free this am. he denies any chest pain or shortness of breath beyond his baseline. he has baseline doe, pnd, orthopnea, ef is 40% in. he was transferred for ercp and placement of biliary stent. on arrival, his vs were stable and he was comfortable. he denied chest pain, sob beyond his basline, fevers, chills. all other review of systems was negative. past medical history: cad s/p cabg, redo cab g s/p aaa repair iddm ckd gout chronic systolic chf h/o gib social history: retired managment consultant. has 5 children. nonsmoker, quit over 20 years ago. no alcohol use. family history: nc physical exam: appearance: nad vitals: t: 9 bp: 115/71 hr: 88 rr: 18 o2: 90% ra eyes: eomi, perrl, conjunctiva clear, noninjected, anicteric, no exudate ent: dry neck: no jvd, no carotid bruits cardiovascular: rrr, nl s1/s2, no m/r/g respiratory: cta bilaterally, comfortable, no wheezing, mild bibasilar crackles gastrointestinal: soft, ruq
---

Lampiran 2. *Dataset* Penelitian: Kelas Negatif

Negatif:
Contoh 1:
Teks: metoprolol tartrate 25 mg nifedipine sr 90 mg daily sevelamer carbonate 1600 mg tid w/ meals simvastatin 80 mg daily iron 325 mg daily discharge medications: metoprolol tartrate 25 mg tablet sig: one (1) tablet po bid (2 times a day). furosemide 40 mg tablet sig: two (2) tablet po bid (2 times a day). nifedipine 90 mg tablet sustained release sig: one (1) tablet sustained release po daily (daily). insulin glargine 100 unit/ml solution sig: eight (8) units subcutaneous once a day. calcitriol 25 mcg capsule sig: one (1) capsule po once a day. sevelamer hcl 800 mg tablet sig: two (2) tablet po three times a day. simvastatin 80 mg tablet sig: one (1) tablet po once a day. discharge disposition: home discharge diagnosis: primary diagnosis: angioedema secondary diagnosis: diabetes mellitus end-stage renal disease discharge condition: mental status: clear and coherent. level of consciousness: alert and interactive. activity status: ambulatory - independent. discharge instructions: you were admitted to the hospital with a reaction called angioedema. we think that this was due to your lisinopril dosing. it may have been related to your iron as well. you should not take lisinipril ever or this may cause a life-threatening reaction. the following changes were made to your medications: stop lisinopril stop glyburide you should follow-up with your pcp 2 weeks. weigh yourself every morning, md if weight goes up more than 3 lbs. you should monitor your blood pressure at home and call your primary care doctor values over 160 systolic. you should check your blood sugars regularly and call your primary care doctor for values over 200 or less than followup instructions: scheduled appointments: provider: phone: date/time: 3:30 provider:, md phone: date/time: 4:00 provider:, md phone: date/time: 8:30 please see your primary care doctor within 2

Contoh 2:
Teks: fasciitis. placement of a suprapubic tube. medications on discharge: sarna lotion applied topically four times per day as needed (for rash). sinemet 25/100-mg tablets one tablet by mouth four times per day. acetaminophen 325-mg tablets one to two tablets by mouth q.4-6h. as needed. metoprolol 25-mg tablets one-half tablet by mouth twice per day. pantoprazole 40-mg tablets one tablet by mouth q.24h. miconazole powder one application topically as needed (for yeast at skin folds). polyvinyl alcohol 4 percent-drops 1 to 2 drops in the eyes as needed. percocet 5/325-mg tablets one to two tablets by mouth q.4- 6h. as needed (for pain). discharge instructions and followup: the patient was instructed to follow up with doctor in plastic surgery in one to two weeks. the patient was to call telephone number for an appointment. the patient was also instructed to follow up with doctor in urology in two weeks; please call telephone number for an appointment. the patient will need wet-to-dry dressing changes twice per day for her left thigh wound and a dry dressing on the right arm wound., dictated by: medquist36 d: 10:24:45 t: 11:31:39 job#:

Contoh 3:
Teks: ventricular wire was placed with good capture. he was taken on for a permanent pacemaker placement. see procedure note for full details. he was being 100% paced with good capture and weaned off neosynephrine after placement. of note, the patient has a history of chronic left pleural effusion. thoracic surgery was consulted and recommended a formal decortication. he needs to follow up with doctor in weeks to determine the timing for the decortication. he continued to be followed by the renal team and dialyzed 3 times per week via left upper extremity fistula. transplant surgery was consulted for a concern in the exam of his lue fistula. reportedly prior to surgery the graft had a strong thrill and postoperatively found to have only a weak pulse. access was found to be patent. last hemodialysis treatment was. chest tubes and pacing wires were removed per cardiac surgery protocol. he was transferred to the step down unit on post operative day 8 in stable condition. physical therapy continued to work with him for increased strength and endurance. a bedside swallowing was performed due to coughing while taking thin liquids. it was suggested a diet of thin liquids and soft consistency solids with 1:1 supervision during meals secondary to mental status. of note, patient did have episodes of sundowning, for which he received haldol with good results. once on the step down unit, mr. well. he was working with physical therapy, tolerating a full po diet and his incisions were healing well. it was felt that he was safe for transfer to rehab at this time. he was discharged to rehab following hemodialysis on pod medications on admission: asa 81mg daily, insulin sliding scale, imdur 60mg daily, lopressor 25mg daily, prilosec 20mg daily, renvela 8mg tid, travatan opthalmic solution, vitamin b complex/vitamin c/folic acid 1 capsule daily, crestor 20mg daily, repaglinide 2mg daily, filgrastim 300 mcg sc saturdays allergies:sulfa, niacin,
---

Lampiran 3. *Source Code Program: Preprocessing*

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from tqdm import tqdm, trange
import pickle

df_adm = pd.read_csv('M:/Dataset/mimic/mimic-iii-clinical-database-1.4/ADMISSIONS.csv')
df_adm.head()
df_adm.ADMITTIME = pd.to_datetime(df_adm.ADMITTIME, format='%Y-%m-%d %H:%M:%S', errors='coerce')
df_adm.DISCHTIME = pd.to_datetime(df_adm.DISCHTIME, format='%Y-%m-%d %H:%M:%S', errors='coerce')
df_adm.DEATHTIME = pd.to_datetime(df_adm.DEATHTIME, format='%Y-%m-%d %H:%M:%S', errors='coerce')
# urutan berdasarkan subject_ID dan tanggal masuk
df_adm = df_adm.sort_values(['SUBJECT_ID','ADMITTIME'])
df_adm = df_adm.reset_index(drop = True)
df_adm.loc[df_adm.SUBJECT_ID == 124,['SUBJECT_ID','ADMITTIME','ADMISSION_TYPE']]
df_adm = df_adm.sort_values(['SUBJECT_ID', 'ADMITTIME'])
df_adm = df_adm.reset_index(drop=True)
df_adm['NEXT_ADMITTIME'] = df_adm.groupby('SUBJECT_ID').ADMITTIME.shift(-1)
df_adm['NEXT_ADMISSION_TYPE'] = df_adm.groupby('SUBJECT_ID').ADMISSION_TYPE.shift(-1)
df_adm.loc[df_adm.SUBJECT_ID == 124,['SUBJECT_ID','ADMITTIME','ADMISSION_TYPE','NEXT_ADMITTIME','NEXT_ADMISSION_TYPE']]
rows = df_adm.NEXT_ADMISSION_TYPE == 'ELECTIVE'
df_adm.loc[rows,'NEXT_ADMITTIME'] = pd.NaT
df_adm.loc[rows,'NEXT_ADMISSION_TYPE'] = np.NaN
df_adm.loc[df_adm.SUBJECT_ID == 124,['SUBJECT_ID','ADMITTIME','ADMISSION_TYPE','NEXT_ADMITTIME','NEXT_ADMISSION_TYPE']]
# urutan berdasarkan subject_ID dan tanggal masuk
df_adm = df_adm.sort_values(['SUBJECT_ID','ADMITTIME'])

df_adm[['NEXT_ADMITTIME','NEXT_ADMISSION_TYPE']] = df_adm.groupby(['SUBJECT_ID'])
[['NEXT_ADMITTIME','NEXT_ADMISSION_TYPE']].fillna(method = 'bfill')
df_adm.loc[df_adm.SUBJECT_ID == 124,['SUBJECT_ID','ADMITTIME','ADMISSION_TYPE','NEXT_ADMITTIME','NEXT_ADMISSION_TYPE']]
# hitung jumlah hari antara keluar dan masuk berikutnya
df_adm['DAYS_NEXT_ADMIT']=  (df_adm.NEXT_ADMITTIME - df_adm.DISCHTIME).dt.total_seconds()/(24*60*60)
```

```
print('Number with a readmission:', (~df_adm.DAYS_NEXT_ADMIT.isnull()).sum())
print('Total Number:', len(df_adm))

df_adm = df_adm.loc[df_adm.ADMISSION_TYPE != 'NEWBORN']
df_adm = df_adm.loc[df_adm.DEATHTIME.isnull()]

df_adm['OUTPUT_LABEL'] = (df_adm.DAYS_NEXT_ADMIT < 30).astype('int')
df_adm['DURATION'] = (df_adm['DISCHTIME']-df_adm['ADMITTIME']).dt.total_seconds()/(24*60*60)

df_notes = pd.read_csv('M:\\Dataset\\mimic\\mimic-iii-clinical-database-1.4\\NOTEEVENTS.csv')
# Sort by subject_ID, HAD_ID then CHARTDATE
df_notes = df_notes.sort_values(by=['SUBJECT_ID','HADM_ID','CHARTDATE'])
# Merge notes table to admissions table
df_adm_notes = pd.merge(df_adm[['SUBJECT_ID','HADM_ID','ADMITTIME','DISCHTIME','DAYS_NEXT_ADMIT','NEXT_ADMITTIME',
                                'ADMISSION_TYPE','DEATHTIME','OUTPUT_LABEL','DURATION']],
                        df_notes[['SUBJECT_ID','HADM_ID','CHARTDATE','TEXT','CATEGORY']],
                        on = ['SUBJECT_ID','HADM_ID'],
                        how = 'left')

# Grab date only, not the time
df_adm_notes.ADMITTIME_C = df_adm_notes.ADMITTIME.apply(lambda x: str(x).split(' ')[0])

df_adm_notes['ADMITTIME_C'] = pd.to_datetime(df_adm_notes.ADMITTIME_C, format = '%Y-%m-%d', errors = 'coerce')
df_adm_notes['CHARTDATE'] = pd.to_datetime(df_adm_notes.CHARTDATE, format = '%Y-%m-%d', errors = 'coerce')

# Gather Discharge Summaries Only
df_discharge = df_adm_notes[df_adm_notes['CATEGORY'] == 'Discharge summary']
df_discharge = (df_discharge.groupby(['SUBJECT_ID','HADM_ID']).nth(-1)).reset_index()
df_discharge=df_discharge[df_discharge['TEXT'].notnull()]

def less_n_days_data(df_adm_notes, n):
    df_less_n = df_adm_notes[
        ((df_adm_notes['CHARTDATE'] - df_adm_notes['ADMITTIME_C']).dt.total_seconds() / (24 * 60 * 60)) < n]
    df_less_n = df_less_n[df_less_n['TEXT'].notnull()]
```

```
    # concatenate first
    df_concat = pd.DataFrame(df_less_n.groupby('HADM_ID')['TEXT'].apply(lambda x: "%s" % ' '.join(x))).reset_index()
    df_concat['OUTPUT_LABEL'] = df_concat['HADM_ID'].apply(
        lambda x: df_less_n[df_less_n['HADM_ID'] == x].OUTPUT_LABEL.values[0])

    return df_concat

import re

def preprocess1(x):
    y = re.sub('\\[(.*?)\\]', '', x)  # remove de-identified brackets
    y = re.sub('[0-9]+\.', '', y)  # remove 1.2. since the segmenter segments based on this
    y = re.sub('dr\.', 'doctor', y)
    y = re.sub('m\.d\.', 'md', y)
    y = re.sub('admission date:', '', y)
    y = re.sub('discharge date:', '', y)
    y = re.sub('--|__|==', '', y)
    return y

def preprocessing(df_less_n):
    df_less_n['TEXT'] = df_less_n['TEXT'].fillna(' ')
    df_less_n['TEXT'] = df_less_n['TEXT'].str.replace('\n', ' ')
    df_less_n['TEXT'] = df_less_n['TEXT'].str.replace('\r', ' ')
    df_less_n['TEXT'] = df_less_n['TEXT'].apply(str.strip)
    df_less_n['TEXT'] = df_less_n['TEXT'].str.lower()

    df_less_n['TEXT'] = df_less_n['TEXT'].apply(lambda x: preprocess1(x))

    # to get 318 words chunks for readmission tasks
    df_len = len(df_less_n)
    want = pd.DataFrame({'ID': [], 'TEXT': [], 'Label': []})
    for i in tqdm(range(df_len)):
        x = df_less_n.TEXT.iloc[i].split()
        n = int(len(x) / 318)
        for j in range(n):
            want = want.append({'TEXT': ' '.join(x[j * 318:(j + 1) * 318]), 'Label': df_less_n.OUTPUT_LABEL.iloc[i],
                                'ID': df_less_n.HADM_ID.iloc[i]}, ignore_index=True)
        if len(x) % 318 > 10:
            want = want.append({'TEXT': ' '.join(x[-(len(x) % 318):]), 'Label': df_less_n.OUTPUT_LABEL.iloc[i],
                                'ID': df_less_n.HADM_ID.iloc[i]}, ignore_index=True)

    return want

df_discharge = preprocessing(df_discharge)
df_discharge.to_pickle("M:/Informatika/End Game/Skripsi/Program/pickle/df_discharge.pkl")
df_discharge = pd.read_pickle('M:\\Informatika\\End Game\\Proposal Skripsi\\4\\18-4-2024\\pickle\\df_discharge.pkl')
df_discharge.shape

readmit_ID = df_adm[df_adm.OUTPUT_LABEL == 1].HADM_ID
not_readmit_ID = df_adm[df_adm.OUTPUT_LABEL == 0].HADM_ID
# subsampling to get the balanced pos/neg numbers of patients for each dataset
not_readmit_ID_use = not_readmit_ID.sample(n=len(readmit_ID), random_state=1)
id_val_test_t = readmit_ID.sample(frac=0.2, random_state=1)
id_val_test_f = not_readmit_ID_use.sample(frac=0.2, random_state=1)

id_train_t = readmit_ID.drop(id_val_test_t.index)
id_train_f = not_readmit_ID_use.drop(id_val_test_f.index)

id_val_t = id_val_test_t.sample(frac=0.5, random_state=1)
id_test_t = id_val_test_t.drop(id_val_t.index)

id_val_f = id_val_test_f.sample(frac=0.5, random_state=1)
id_test_f = id_val_test_f.drop(id_val_f.index)

# test if there is overlap between train and test, should return "array([], dtype=int64)"
(pd.Index(id_test_t).intersection(pd.Index(id_train_t))).values

id_test = pd.concat([id_test_t, id_test_f])
test_id_label = pd.DataFrame(data=list(zip(id_test, [1] * len(id_test_t) + [0] * len(id_test_f))),
                             columns=['id', 'label'])
```

```
id_val = pd.concat([id_val_t, id_val_f])
val_id_label = pd.DataFrame(data=list(zip(id_val, [1] * len(id_val_t) + [0] * len(id_val_f))), columns=['id', 'label'])

id_train = pd.concat([id_train_t, id_train_f])
train_id_label = pd.DataFrame(data=list(zip(id_train, [1] * len(id_train_t) + [0] * len(id_train_f))),
                              columns=['id', 'label'])

discharge_train = df_discharge[df_discharge.ID.isin(train_id_label.id)]
discharge_val = df_discharge[df_discharge.ID.isin(val_id_label.id)]
discharge_test = df_discharge[df_discharge.ID.isin(test_id_label.id)]

df = pd.concat([not_readmit_ID_use, not_readmit_ID])
df = df.drop_duplicates(keep=False)
# check to see if there are overlaps
(pd.Index(df).intersection(pd.Index(not_readmit_ID_use))).values

# for this set of split with random_state=1, we find we need 400 more negative training samples
not_readmit_ID_more = df.sample(n=400, random_state=1)
discharge_train_snippets = pd.concat([df_discharge[df_discharge.ID.isin(not_readmit_ID_more)], discharge_train])

# shuffle
discharge_train_snippets = discharge_train_snippets.sample(frac=1, random_state=1).reset_index(drop=True)

# check if balanced
discharge_train_snippets.Label.value_counts()

discharge_train_snippets.to_csv('M:\\Informatika\\End Game\\Proposal Skripsi\\4\\18-4-2024\\data\\discharge\\train.csv')
discharge_val.to_csv('M:\\Informatika\\End Game\\Proposal Skripsi\\4\\18-4-2024\\data\\discharge\\val.csv')
discharge_test.to_csv('M:\\Informatika\\End Game\\Proposal Skripsi\\4\\18-4-2024\\data\\discharge\\test.csv')
```

Lampiran 4. *Source Code Program: Modeling* XLNet-BiGRU-ATT

```
import sys
from transformers import XLNetTokenizer
import torch
import torch.nn as nn
from transformers import XLNetForSequenceClassification, XLNetTokenizer, AdamW, get_linear_schedule_with_warmup
from torch.utils.data import TensorDataset, DataLoader, RandomSampler, DistributedSampler
from tqdm import tqdm
import matplotlib.pyplot as plt
import math

tokenizer = XLNetTokenizer.from_pretrained('xlnet-base-cased')
data_dir = 'M:\\Informatika\\End Game\\Proposal Skripsi\\4\\18-4-2024\\data\\discharge'
train_examples = None
num_train_steps = None
if do_train:
    train_examples = processor.get_train_examples(data_dir)
    num_train_steps = int(
        len(train_examples) / train_batch_size / gradient_accumulation_steps * num_train_epochs)

class XLNetBiGRUAttention(nn.Module):
    def __init__(self, xlnet_model_path, num_labels=1):
        super().__init__()
        # Load pre-trained XLNetForSequenceClassification model
        self.xlnet = XLNetForSequenceClassification.from_pretrained(xlnet_model_path, num_labels=num_labels)
        # All XLNet parameters will be updatable
        for param in self.xlnet.parameters():
            param.requires_grad = True
        hidden_size = self.xlnet.config.d_model
        # BiGRU layer
        self.bigru = nn.GRU(hidden_size, hidden_size // 2,
                            bidirectional=True, batch_first=True)
        # Attention mechanism
        self.attention = nn.Linear(hidden_size, 1)
        # Text vector representation (fully connected layer)
        self.fc = nn.Linear(hidden_size, hidden_size)
        # Dropout layer
        self.dropout = nn.Dropout(0.1)
        # Replace the original classifier
        self.classifier = nn.Linear(hidden_size, num_labels)
        # Loss function for binary classification
        self.loss_fct = nn.BCEWithLogitsLoss()
    def forward(self, input_ids, attention_mask, labels=None):
        # XLNet forward pass
        outputs = self.xlnet(input_ids=input_ids,
                             attention_mask=attention_mask,
                             labels=labels,
                             output_hidden_states=True)
        sequence_output = outputs.hidden_states[-1]
        # BiGRU
        gru_output, _ = self.bigru(sequence_output)
        # Attention mechanism
        attention_scores = self.attention(gru_output)
        attention_weights = torch.softmax(attention_scores, dim=1)
        context_vector = torch.sum(attention_weights * gru_output, dim=1)
        # Text vector representation with dropout
        text_vector = self.dropout(torch.relu(self.fc(context_vector)))
        # Output layer
        logits = self.classifier(text_vector)
        # Use our own loss function if labels are provided
        if labels is not None:
            loss = self.loss_fct(logits.view(-1), labels.view(-1))
        else:
            loss = None
        return loss, logits, outputs.hidden_states
# Model initialization
xlnet_model_path = 'M:\Informatika\End Game\Proposal Skripsi\Model\mimiciii_xlnet_5e_128b\mimiciii_xlnet_5e_128b'
model = XLNetBiGRUAttention(xlnet_model_path)
# Send data to the chosen device
model.to(device)
```

Lampiran 5. *Source Code Program: Training Model* XLNet-BiGRU-ATT

```
import torch
import numpy as np
import pandas as pd
from tqdm import tqdm, trange
from torch.utils.data import TensorDataset, DataLoader, RandomSampler, SequentialSampler, DistributedSampler
from transformers import get_linear_schedule_with_warmup
import os
import logging

# Set up logging
logging.basicConfig(level=logging.INFO, format='%(asctime)s - %(levelname)s - %(message)s')
logger = logging.getLogger(__name__)

if do_train:
    no_decay = ['bias', 'LayerNorm.bias', 'LayerNorm.weight']
    optimizer_grouped_parameters = [
        {'params': [p for n, p in model.named_parameters() if not any(nd in n for nd in no_decay)], 'weight_decay': 0.01},
        {'params': [p for n, p in model.named_parameters() if any(nd in n for nd in no_decay)], 'weight_decay': 0.0}
    ]

    optimizer = torch.optim.AdamW(optimizer_grouped_parameters, lr=learning_rate)
    warmup_steps = int(warmup_proportion * num_train_steps)
    scheduler = get_linear_schedule_with_warmup(optimizer,
                                                num_warmup_steps=warmup_steps,
                                                num_training_steps=num_train_steps)

    # Prepare training data
    train_features = convert_examples_to_features(
        train_examples, label_list, max_seq_length, tokenizer)

    logger.info("***** Running training *****")
    logger.info("  Num examples = %d", len(train_examples))
    logger.info("  Batch size = %d", train_batch_size)
    logger.info("  Num steps = %d", num_train_steps)
    all_input_ids = torch.tensor([f.input_ids for f in train_features], dtype=torch.long).to(device)
    all_attention_mask = torch.tensor([f.attention_mask for f in train_features], dtype=torch.long).to(device)
    all_label_ids = torch.tensor([f.label_id for f in train_features], dtype=torch.float).to(device)

    train_data = TensorDataset(all_input_ids, all_attention_mask, all_label_ids)

    if local_rank == -1:
        train_sampler = RandomSampler(train_data)
    else:
        train_sampler = DistributedSampler(train_data)

    train_dataloader = DataLoader(train_data, sampler=train_sampler, batch_size=train_batch_size)

    model.train()
    global_step = 0

    for epoch in range(int(num_train_epochs)):
        logger.info(f"***** Epoch {epoch+1}/{num_train_epochs} *****")
        progress_bar = tqdm(train_dataloader, desc="Training", disable=local_rank not in [-1, 0], miniters=1, ncols=100)
        tr_loss = 0
        nb_tr_examples, nb_tr_steps = 0, 0

        for step, batch in enumerate(progress_bar):
            batch = tuple(t.to(device) for t in batch)
            input_ids, attention_mask, label_ids = batch

            loss, logits, hidden_states = model(input_ids=input_ids,
                                                attention_mask=attention_mask,
                                                labels=label_ids)

            if n_gpu > 1:
                loss = loss.mean()

            if gradient_accumulation_steps > 1:
                loss = loss / gradient_accumulation_steps
```

```
            loss.backward()
            tr_loss += loss.item()
            nb_tr_examples += input_ids.size(0)
            nb_tr_steps += 1

            if (step + 1) % gradient_accumulation_steps == 0:
                optimizer.step()
                scheduler.step()
                model.zero_grad()
                global_step += 1

            if (step + 1) % 200 == 0:
                progress_bar.set_postfix({'loss': f'{loss.item():.3f}'})

        avg_train_loss = tr_loss / len(train_dataloader)
        logger.info(f'Epoch {epoch+1}/{num_train_epochs} - Average Loss: {avg_train_loss:.4f}')

        # Save model after each epoch
        output_model_file = f'/content/drive/MyDrive/TUGAS AKHIR/Output6/xlnet_bigru_attention_model_{readmission_mode}_epoch{epo
        torch.save(model.state_dict(), output_model_file)
        logger.info(f"Model for epoch {epoch+1} saved to {output_model_file}")

    logger.info("Training completed")

    # Save the final model (which is the same as the last epoch's model)
    final_output_model_file = f'/content/drive/MyDrive/TUGAS AKHIR/Output6/xlnet_bigru_attention_model_{readmission_mode}_final.b
    torch.save(model.state_dict(), final_output_model_file)
    logger.info(f"Final model saved to {final_output_model_file}")
```

Lampiran 6. *Source Code Program*: Evaluasi Model XLNet-BiGRU-ATT

```
import torch
import numpy as np
import pandas as pd
from tqdm import tqdm
from torch.utils.data import TensorDataset, DataLoader, SequentialSampler, DistributedSampler
from sklearn.metrics import roc_curve, auc, precision_recall_curve
import os

if do_eval:
    eval_examples = processor.get_test_examples(data_dir)
    eval_features = convert_examples_to_features(
        eval_examples, label_list, max_seq_length, tokenizer)
    logger.info("***** Running evaluation *****")
    logger.info("  Num examples = %d", len(eval_examples))
    logger.info("  Batch size = %d", eval_batch_size)

    all_input_ids = torch.tensor([f.input_ids for f in eval_features], dtype=torch.long).to(device)
    all_attention_mask = torch.tensor([f.attention_mask for f in eval_features], dtype=torch.long).to(device)
    all_label_ids = torch.tensor([f.label_id for f in eval_features], dtype=torch.float).to(device)

    eval_data = TensorDataset(all_input_ids, all_attention_mask, all_label_ids)

    if local_rank == -1:
        eval_sampler = SequentialSampler(eval_data)
    else:
        eval_sampler = DistributedSampler(eval_data)

    eval_dataloader = DataLoader(eval_data, sampler=eval_sampler, batch_size=eval_batch_size)

    model.eval()
    eval_loss, eval_accuracy = 0, 0
    nb_eval_steps, nb_eval_examples = 0, 0
    true_labels = []
    pred_labels = []
    logits_history = []

    for batch in tqdm(eval_dataloader, desc="Evaluating"):
        batch = tuple(t.to(device) for t in batch)
        input_ids, attention_mask, label_ids = batch

        with torch.no_grad():
            loss, logits, hidden_states = model(input_ids=input_ids,
                                                attention_mask=attention_mask,
                                                labels=label_ids)

        logits = torch.sigmoid(logits).squeeze().detach().cpu().numpy()
        label_ids = label_ids.to('cpu').numpy()

        eval_loss += loss.mean().item()
        outputs = np.asarray([1 if i else 0 for i in (logits >= 0.5)])
        tmp_eval_accuracy = np.sum(outputs == label_ids)

        true_labels.extend(label_ids.flatten().tolist())
        pred_labels.extend(outputs.flatten().tolist())
        logits_history.extend(logits.flatten().tolist())

        eval_accuracy += tmp_eval_accuracy
        nb_eval_examples += input_ids.size(0)
        nb_eval_steps += 1

    eval_loss = eval_loss / nb_eval_steps
    eval_accuracy = eval_accuracy / nb_eval_examples

    df = pd.DataFrame({'logits': logits_history, 'pred_label': pred_labels, 'label': true_labels})
    string = f'logits_xlnet_bigru_attention_chunks.csv'
    df.to_csv(os.path.join(output_dir, string))

    df_test = pd.read_csv(os.path.join(data_dir, "test.csv"))
    fpr, tpr, df_out = vote_score(df_test, logits_history, readmission_mode, output_dir)
```

```
string = f'logits_xlnet_bigru_attention_readmissions.csv'
df_out.to_csv(os.path.join(output_dir, string))

rp80 = vote_pr_curve(df_test, logits_history, readmission_mode, output_dir)

result = {
    'eval_loss': eval_loss,
    'eval_accuracy': eval_accuracy,
    'global_step': global_step,
    'training loss': tr_loss / nb_tr_steps,
    'RP80': rp80
}

output_eval_file = os.path.join(output_dir, "eval_results.txt")
with open(output_eval_file, "w") as writer:
    logger.info("***** Eval results *****")
    for key in sorted(result.keys()):
        logger.info("  %s = %s", key, str(result[key]))
        writer.write("%s = %s\n" % (key, str(result[key])))
```

Lampiran 7. Permohonan akses data pada *PhysioNet*

- Profile
- Password
- Emails
- Username
- Cloud
- ORCID
- Credentialing
- Training
- Certification
- Agreements

Data Use Agreements

You have signed the following Data Use Agreements:

Date of Agreement	Agreement	Project	View Agreement
March 6, 2024	PhysioNet Credentialed Health Data Use Agreement 1.5.0	MIMIC-IV (v2.2)	View
March 6, 2024	PhysioNet Credentialed Health Data Use Agreement 1.5.0	Chest X-ray Dataset with Lung Segmentation (v1.0.0)	View
March 6, 2024	PhysioNet Credentialed Health Data Use Agreement 1.5.0	Learning to Ask Like a Physician: a Discharge Summary Clinical Questions (DiSCQ) Dataset (v1.0)	View
March 6, 2024	PhysioNet Credentialed Health Data Use Agreement 1.5.0	Annotated Question-Answer Pairs for Clinical Notes in the MIMIC-III Database (v1.0.0)	View
March 6, 2024	Korea Credentialed Health Data Agreement-1-0-0	INSPIRE, a publicly available research dataset for perioperative medicine (v1.2)	View
Oct. 18, 2023	PhysioNet Credentialed Health Data Use Agreement 1.5.0	MIMIC-III Clinical Database (v1.4)	View

Code of Conduct

You have signed the following Code of Conduct:

Date of Agreement	Name	Version
Sept. 27, 2023	Code of Conduct	1.0

Lampiran 8. *PhysioNet Credentialed Health: Signed Data Use Agreement*

PhysioNet Credentialed Health Data Use Agreement 1.5.0

Data Use Agreement for the MIMIC-III Clinical Database (v1.4)

If I am granted access to the database:

1. I will not attempt to identify any individual or institution referenced in PhysioNet restricted data.
2. I will exercise all reasonable and prudent care to avoid disclosure of the identity of any individual or institution referenced in PhysioNet restricted data in any publication or other communication.
3. I will not share access to PhysioNet restricted data with anyone else.
4. I will exercise all reasonable and prudent care to maintain the physical and electronic security of PhysioNet restricted data.
5. If I find information within PhysioNet restricted data that I believe might permit identification of any individual or institution, I will report the location of this information promptly by email to PHI-report@physionet.org, citing the location of the specific information in question.
6. I have requested access to PhysioNet restricted data for the sole purpose of lawful use in scientific research, and I will use my privilege of access, if it is granted, for this purpose and no other.
7. I have completed a training program in human research subject protections and HIPAA regulations, and I am submitting proof of having done so.
8. I will indicate the general purpose for which I intend to use the database in my application.
9. If I openly disseminate my results, I will also contribute the code used to produce those results to a repository that is open to the research community.
10. This agreement may be terminated by either party at any time, but my obligations with respect to PhysioNet data shall continue after termination.

SIGNED: **Annisa Darwis**

DATED: **Oct. 18, 2023**

LEMBAR PERBAIKAN SKRIPSI

"PREDIKSI KEBERLANJUTAN RAWAT INAP PASIEN RUMAH SAKIT MENGGUNAKAN XLNET-BiGRU-ATT PADA CATATAN KHUSUS KLINIS PASIEN"

OLEH:

ANNISA DARWIS
D121 20 1032

Skripsi ini telah dipertahankan pada Ujian Akhir Sarjana tanggal 04 November 2024. Telah dilakukan perbaikan penulisan dan isi skripsi berdasarkan usulan dari penguji dan pembimbing skripsi.

Persetujuan perbaikan oleh tim penguji:

	Nama	Tanda Tangan
Ketua/Sekretaris	Ir. Anugrayani Bustami, S.T., M.T	
Anggota	Dr. Ir. Ingrid Nurtanio, M.T.	
Anggota	Ir. Tyanita Puti Marindah Wardhani.,S.T.M.Inf.	

Persetujuan Perbaikan oleh pembimbing:

Pembimbing	Nama	Tanda Tangan
I	Ir. Anugrayani Bustami, S.T., M.T	