

Predicting_Heart_Problem_with_BERT_in_Tensorflow_Final RAW

July 30, 2020

##Mounting Google Drive

```
[1]: from google.colab import drive
drive.mount("/GD")
```

Go to this URL in a browser: https://accounts.google.com/o/oauth2/auth?client_id=947318989803-6bn6qk8qdgf4n4g3pfee6491hc0brc4i.apps.googleusercontent.com&redirect_uri=urn%3Aietf%3Awg%3Aoauth%3A2.0%3Aoob&response_type=code&scope=email%20https%3A%2f%2fwww.googleapis.com%2fauth%2fdocs.test%20https%3A%2f%2fwww.googleapis.com%2fauth%2fdrive%20https%3A%2f%2fwww.googleapis.com%2fauth%2fdrive.photos.readonly%20https%3A%2f%2fwww.googleapis.com%2fauth%2fpeopleapi.readonly

Enter your authorization code:

.....

Mounted at /GD

0.1 Importing Necessary Libraries

```
[2]: !pip install tensorflow==1.15.0
import tensorflow as tf
print(tf.__version__)
```

Collecting tensorflow==1.15.0

Downloading https://files.pythonhosted.org/packages/3f/98/5a99af92fb911d7a88a0005ad55005f35b4c1ba8d75fba02df726cd936e6/tensorflow-1.15.0-cp36-cp36m-manylinux2010_x86_64.whl (412.3MB)

| 412.3MB 42kB/s

Collecting tensorboard<1.16.0,>=1.15.0

Downloading <https://files.pythonhosted.org/packages/1e/e9/d3d747a97f7188f48aa5eda486907f3b345cd409f0a0850468ba867db246/tensorboard-1.15.0-py3-none-any.whl> (3.8MB)

| 3.8MB 49.3MB/s

Requirement already satisfied: wheel>=0.26 in

/usr/local/lib/python3.6/dist-packages (from tensorflow==1.15.0) (0.34.2)

Collecting tensorflow-estimator==1.15.1

Downloading https://files.pythonhosted.org/packages/de/62/2ee9cd74c9fa2fa450877847ba560b260f5d0fb70ee0595203082dafcc9d/tensorflow_estimator-1.15.1-py2.py3-none-any.whl

```

y3-none-any.whl (503kB)
|                               | 512kB 48.4MB/s
Collecting gast==0.2.2
  Downloading https://files.pythonhosted.org/packages/4e/35/11749bf99b2d4e3cceb4
d55ca22590b0d7c2c62b9de38ac4a4a7f4687421/gast-0.2.2.tar.gz
Requirement already satisfied: wrapt>=1.11.1 in /usr/local/lib/python3.6/dist-
packages (from tensorflow==1.15.0) (1.12.1)
Requirement already satisfied: grpcio>=1.8.6 in /usr/local/lib/python3.6/dist-
packages (from tensorflow==1.15.0) (1.30.0)
Requirement already satisfied: protobuf>=3.6.1 in /usr/local/lib/python3.6/dist-
packages (from tensorflow==1.15.0) (3.12.2)
Requirement already satisfied: keras-applications>=1.0.8 in
/usr/local/lib/python3.6/dist-packages (from tensorflow==1.15.0) (1.0.8)
Requirement already satisfied: keras-preprocessing>=1.0.5 in
/usr/local/lib/python3.6/dist-packages (from tensorflow==1.15.0) (1.1.2)
Requirement already satisfied: termcolor>=1.1.0 in
/usr/local/lib/python3.6/dist-packages (from tensorflow==1.15.0) (1.1.0)
Requirement already satisfied: opt-einsum>=2.3.2 in
/usr/local/lib/python3.6/dist-packages (from tensorflow==1.15.0) (3.3.0)
Requirement already satisfied: astor>=0.6.0 in /usr/local/lib/python3.6/dist-
packages (from tensorflow==1.15.0) (0.8.1)
Requirement already satisfied: numpy<2.0,>=1.16.0 in
/usr/local/lib/python3.6/dist-packages (from tensorflow==1.15.0) (1.18.5)
Requirement already satisfied: absl-py>=0.7.0 in /usr/local/lib/python3.6/dist-
packages (from tensorflow==1.15.0) (0.9.0)
Requirement already satisfied: six>=1.10.0 in /usr/local/lib/python3.6/dist-
packages (from tensorflow==1.15.0) (1.15.0)
Requirement already satisfied: google-pasta>=0.1.6 in
/usr/local/lib/python3.6/dist-packages (from tensorflow==1.15.0) (0.2.0)
Requirement already satisfied: setuptools>=41.0.0 in
/usr/local/lib/python3.6/dist-packages (from
tensorflow<1.16.0,>=1.15.0->tensorflow==1.15.0) (49.1.0)
Requirement already satisfied: markdown>=2.6.8 in /usr/local/lib/python3.6/dist-
packages (from tensorflowboard<1.16.0,>=1.15.0->tensorflow==1.15.0) (3.2.2)
Requirement already satisfied: werkzeug>=0.11.15 in
/usr/local/lib/python3.6/dist-packages (from
tensorflowboard<1.16.0,>=1.15.0->tensorflow==1.15.0) (1.0.1)
Requirement already satisfied: h5py in /usr/local/lib/python3.6/dist-packages
(from keras-applications>=1.0.8->tensorflow==1.15.0) (2.10.0)
Requirement already satisfied: importlib-metadata; python_version < "3.8" in
/usr/local/lib/python3.6/dist-packages (from
markdown>=2.6.8->tensorflowboard<1.16.0,>=1.15.0->tensorflow==1.15.0) (1.7.0)
Requirement already satisfied: zipp>=0.5 in /usr/local/lib/python3.6/dist-
packages (from importlib-metadata; python_version <
"3.8"->markdown>=2.6.8->tensorflowboard<1.16.0,>=1.15.0->tensorflow==1.15.0) (3.1.0)
Building wheels for collected packages: gast
  Building wheel for gast (setup.py) ... done
  Created wheel for gast: filename=gast-0.2.2-cp36-none-any.whl size=7540

```

```

sha256=90d86e22f37d1bde8e0f110b199a10d479014ae5ee25f200355daf21f7a9c3cc
  Stored in directory: /root/.cache/pip/wheels/5c/2e/7e/a1d4d4fcebe6c381f378ce77
43a3ced3699feb89bcfbdadadd
Successfully built gast
ERROR: tensorflow-probability 0.10.0 has requirement gast>=0.3.2, but
you'll have gast 0.2.2 which is incompatible.
Installing collected packages: tensorboard, tensorflow-estimator, gast,
tensorflow
  Found existing installation: tensorboard 2.2.2
  Uninstalling tensorboard-2.2.2:
    Successfully uninstalled tensorboard-2.2.2
  Found existing installation: tensorflow-estimator 2.2.0
  Uninstalling tensorflow-estimator-2.2.0:
    Successfully uninstalled tensorflow-estimator-2.2.0
  Found existing installation: gast 0.3.3
  Uninstalling gast-0.3.3:
    Successfully uninstalled gast-0.3.3
  Found existing installation: tensorflow 2.2.0
  Uninstalling tensorflow-2.2.0:
    Successfully uninstalled tensorflow-2.2.0
Successfully installed gast-0.2.2 tensorboard-1.15.0 tensorflow-1.15.0
tensorflow-estimator-1.15.1
1.15.0

```

```

[3]: import pandas as pd
import tensorflow as tf
import tensorflow_hub as hub
from datetime import datetime
from sklearn.model_selection import train_test_split
import os

print("tensorflow version : ", tf.__version__)
print("tensorflow_hub version : ", hub.__version__)

```

```

tensorflow version : 1.15.0
tensorflow_hub version : 0.8.0

```

```

[4]: #Installing BERT module
!pip install bert-tensorflow

```

```

Collecting bert-tensorflow
  Downloading https://files.pythonhosted.org/packages/a6/66/7eb4e8b6ea35b7
cc54c322c816f976167a43019750279a8473d355800a93/bert_tensorflow-1.0.1-py2.py3-non
e-any.whl (67kB)
    | 71kB 3.2MB/s
Requirement already satisfied: six in /usr/local/lib/python3.6/dist-
packages (from bert-tensorflow) (1.15.0)

```

Installing collected packages: bert-tensorflow
Successfully installed bert-tensorflow-1.0.1

```
[7]: #Importing BERT modules
import bert
from bert import run_classifier
from bert import optimization
from bert import tokenization
```

0.2 ##Setting The Output Directory

While fine-tuning the model, we will save the training checkpoints and the model in an output directory so that we can use the trained model for our predictions later.

The following code block sets an output directory :

```
[8]: # Set the output directory for saving model file
OUTPUT_DIR = '/GD/My Drive/Colab Notebooks/LifeHackFinal/'

#@markdown Whether or not to clear/delete the directory and create a new one
DO_DELETE = False #@param {type:"boolean"}

if DO_DELETE:
    try:
        tf.gfile.DeleteRecursively(OUTPUT_DIR)
    except:
        pass

tf.gfile.MakeDirs(OUTPUT_DIR)
print('***** Model output directory: {} *****'.format(OUTPUT_DIR))
```

```
***** Model output directory: /GD/My Drive/Colab Notebooks/LifeHackFinal/ *****
```

0.3 ##Loading The Data

We will now load the data from a Google Drive directory and will also split the training set in to training and validation sets.

```
[10]: train = pd.read_csv("/GD/My Drive/Colab Notebooks/LifeHackFinal/
    ↳raw_sampled_data.csv", encoding = "ISO-8859-1")
train['question'] = train['question'].apply(str)

from sklearn.model_selection import train_test_split

train, val = train_test_split(train, test_size = 0.2, random_state = 100)

[11]: train.to_csv('trainFinal.csv')
val.to_csv('testFinal.csv')
```

```
[12]: #Training set sample
train.head(15)
```

```
[12]:      category  ... question
629      hipertensi  ... Assalamualaikum dok, sy mau bertanya. Ibu saya...
669      hipertensi  ... Dok, 1 bulan yang lalu saya periksa ke puskesm...
25      aritmia  ... Dok saya wanita berusia 24 tahu. Sejak 5 tahun...
88      aritmia  ... Dok dada trasa aga sesak tapi tidak batuk, 3ha...
395      gagal-jantung  ... Halo dok, sy pengidap jantung bocor, selalu ko...
685      hipertensi  ... Permisi, Dok. saya berusia 21th, 3minggu yang ...
1049      stroke  ... Dok, 6 bulan lalu saya terkena stroke (pecah p...
975      serangan-jantung  ... Dok saya kaya terasa sesak dok terus kadang da...
356      gagal-jantung  ... Bapak saya skrg lg d rawat di RS\r\nkepala pus...
146      aritmia  ... Dok saya mau bertanya? Kenapa denyut nadi bisa...
1281      stroke  ... Selamat siang dokter, Maaf saya ingin bertanya,...
45      aritmia  ... Maaf Dok, detakan jantung saya tiba-tiba kuat ...
330      gagal-jantung  ... Pagi dok , saya mau tanya dok mamah saya tanga...
160      aritmia  ... Dok, akhir" ini ketika saya menaiki tangga jan...
306      gagal-jantung  ... dok,, ibu saya kira" umur 37th.. saring kaget ...
```

[15 rows x 6 columns]

```
[13]: print("Training Set Shape :", train.shape)
print("Validation Set Shape :", val.shape)
#print("Test Set Shape :", test.shape)
```

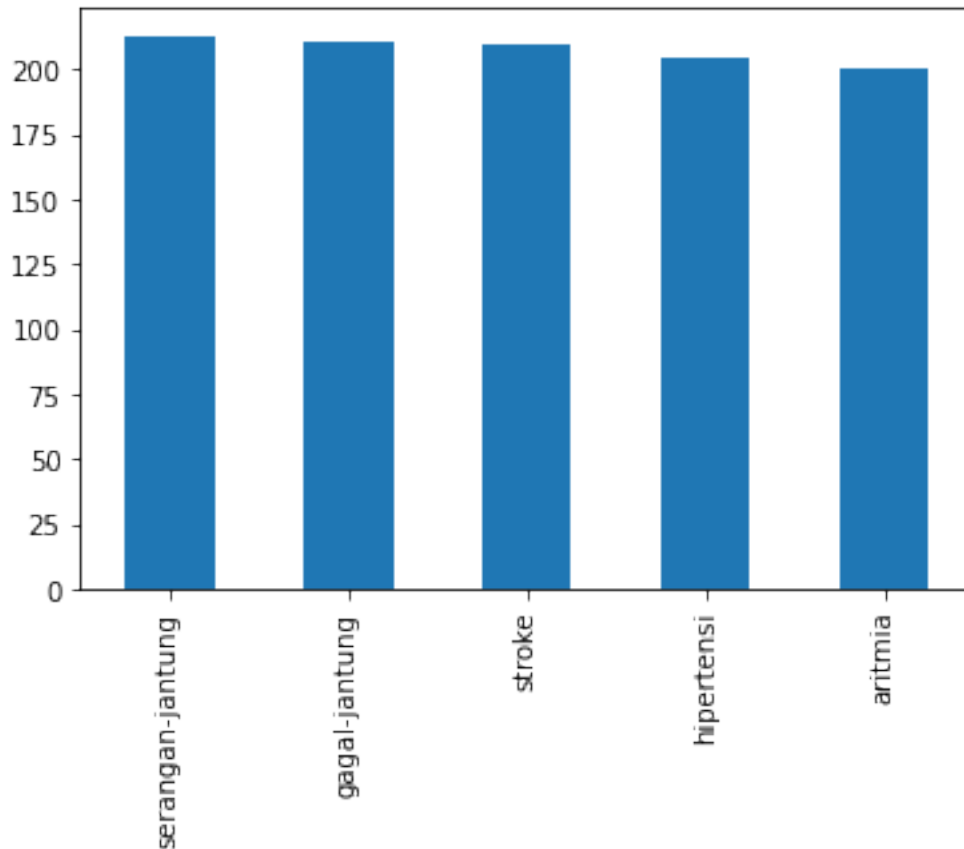
```
Training Set Shape : (1040, 6)
Validation Set Shape : (260, 6)
```

```
[14]: #unique classes
train['category'].unique()
```

```
[14]: array(['hipertensi', 'aritmia', 'gagal-jantung', 'stroke',
        'serangan-jantung'], dtype=object)
```

```
[15]: #Distribution of classes
train['category'].value_counts().plot(kind = 'bar')
```

```
[15]: <matplotlib.axes._subplots.AxesSubplot at 0x7f588b444e10>
```



```
[16]: DATA_COLUMN = 'question'
      LABEL_COLUMN = 'category'
      # The list containing all the classes (train['SECTION'].unique())
      label_list = list(train['category'].unique())
```

0.4 Data Preprocessing

BERT model accept only a specific type of input and the datasets are usually structured to have the following four features:

- guid : A unique id that represents an observation.
- text_a : The text we need to classify into given categories
- text_b: It is used when we're training a model to understand the relationship between sentences and it does not apply for classification problems.
- label: It consists of the labels or classes or categories that a given text belongs to.

In our dataset we have text_a and label. The following code block will create objects for each of the above mentioned features for all the records in our dataset using the InputExample class provided in the BERT library.

```
[17]: train_InputExamples = train.apply(lambda x: bert.run_classifier.
      ↳ InputExample(guid=None,
      text_a =
      ↳ x[DATA_COLUMN],
      text_b =
      ↳ None,
      label =
      ↳ x[LABEL_COLUMN]), axis = 1)

val_InputExamples = val.apply(lambda x: bert.run_classifier.
      ↳ InputExample(guid=None,
      text_a =
      ↳ x[DATA_COLUMN],
      text_b =
      ↳ None,
      label =
      ↳ x[LABEL_COLUMN]), axis = 1)
```

```
[18]: train_InputExamples
```

```
[18]: 629 <bert.run_classifier.InputExample object at 0x...
      669 <bert.run_classifier.InputExample object at 0x...
      25 <bert.run_classifier.InputExample object at 0x...
      88 <bert.run_classifier.InputExample object at 0x...
      395 <bert.run_classifier.InputExample object at 0x...
      ...
      802 <bert.run_classifier.InputExample object at 0x...
      53 <bert.run_classifier.InputExample object at 0x...
      350 <bert.run_classifier.InputExample object at 0x...
      79 <bert.run_classifier.InputExample object at 0x...
      792 <bert.run_classifier.InputExample object at 0x...
      Length: 1040, dtype: object
```

```
[19]: print("Row 0 - guid of training set : ", train_InputExamples.iloc[0].guid)
      print("\n_____ \nRow 0 - text_a of training set : ", train_InputExamples.
      ↳ iloc[0].text_a)
      print("\n_____ \nRow 0 - text_b of training set : ", train_InputExamples.
      ↳ iloc[0].text_b)
      print("\n_____ \nRow 0 - label of training set : ", train_InputExamples.
      ↳ iloc[0].label)
```

Row 0 - guid of training set : None

 Row 0 - text_a of training set : Assalamualaikum dok, sy mau bertanya. Ibu saya menderita hipertensi. Sudah di bawa ke dokter dan diberi obat penurun hipertensi. Tapi kenapa kok tekanan darahnya tidak kunjung normal? Padahal sudah

rutin minum obat & mengkonsumsi banyak buah2an (timun, semangka, dll). Malah badannya terasa lemas, pusing & penglihatan kabur. Terimakasih. Wassalamualaikum
õ õ

Row 0 - text_b of training set : None

Row 0 - label of training set : hipertensi

We will now get down to business with the pretrained BERT. In this example we will use the bert_uncased_L-12_H-768_A-12/1 model. To check all available versions click [here](#).

We will be using the vocab.txt file in the model to map the words in the dataset to indexes. Also the loaded BERT model is trained on uncased/lowercase data and hence the data we feed to train the model should also be of lowercase.

The following code block loads the pre-trained BERT model and initializes a tokenizer object for tokenizing the texts.

```
[20]: # This is a path to an uncased (all lowercase) version of BERT
BERT_MODEL_HUB = "https://tfhub.dev/google/bert_multi_cased_L-12_H-768_A-12/1"

def create_tokenizer_from_hub_module():
    """Get the vocab file and casing info from the Hub module."""
    with tf.Graph().as_default():
        bert_module = hub.Module(BERT_MODEL_HUB)
        tokenization_info = bert_module(signature="tokenization_info", as_dict=True)
        with tf.Session() as sess:
            vocab_file, do_lower_case = sess.run([tokenization_info["vocab_file"],
                                                  tokenization_info["do_lower_case"]])

    return bert.tokenization.FullTokenizer(
        vocab_file=vocab_file, do_lower_case=do_lower_case)

tokenizer = create_tokenizer_from_hub_module()
```

INFO:tensorflow:Saver not created because there are no variables in the graph to restore

INFO:tensorflow:Saver not created because there are no variables in the graph to restore

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/bert/tokenization.py:125: The name tf.gfile.GFile is deprecated. Please use tf.io.gfile.GFile instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-

packages/bert/tokenization.py:125: The name tf.gfile.GFile is deprecated. Please use tf.io.gfile.GFile instead.

```
[21]: #Here is what the tokenised sample of the first training set observation looks  
      ↳ like  
      print(tokenizer.tokenize(train_InputExamples.iloc[9].text_a))
```

```
['Dok', 'saya', 'mau', 'bertan', '##ya', '?', 'Ken', '##apa', 'den', '##yu',  
 '##t', 'nad', '##i', 'bisa', 'di', 'pak', '##ai', 'untuk', 'menentukan', 'be',  
 '##ban', 'kerja', 'fi', '##sik', ',', 'tingkat', 'kes', '##eh', '##atan', ',',  
 'tingkat', 'ke', '##bu', '##garan', 'dan', 'tingkat', 'stress', '?']
```

We will now format out text in to input features which the BERT model expects. We will also set a sequence length which will be the length of the input features.

```
[23]: max_len = max([len(tokenizer.tokenize(train_InputExamples.iloc[IDX].text_a))  
      ↳ for IDX in range(1040)])  
      print('Max length: ', max_len)
```

Max length: 2161

```
[24]: # We'll set sequences to be at most 128 tokens long.  
      MAX_SEQ_LENGTH = 256  
  
      # Convert our train and validation features to InputFeatures that BERT  
      ↳ understands.  
      train_features = bert.run_classifier.  
      ↳ convert_examples_to_features(train_InputExamples, label_list,  
      ↳ MAX_SEQ_LENGTH, tokenizer)  
  
      val_features = bert.run_classifier.  
      ↳ convert_examples_to_features(val_InputExamples, label_list, MAX_SEQ_LENGTH,  
      ↳ tokenizer)
```

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/bert/run_classifier.py:774: The name tf.logging.info is deprecated. Please use tf.compat.v1.logging.info instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/bert/run_classifier.py:774: The name tf.logging.info is deprecated. Please use tf.compat.v1.logging.info instead.

INFO:tensorflow:Writing example 0 of 1040

INFO:tensorflow:Writing example 0 of 1040

INFO:tensorflow:*** Example ***

INFO:tensorflow:*** Example ***

INFO:tensorflow:guid: None

INFO:tensorflow:guid: None

INFO:tensorflow:tokens: [CLS] Ass ##ala ##mua ##lai ##kum dok , sy mau bertan
##ya . Ibu saya men ##der ##ita hip ##erten ##si . Sud ##ah di ba ##wa ke dokter
dan diberi oba ##t pen ##uru ##n hip ##erten ##si . Ta ##pi ken ##apa ko ##k
tekanan darah ##nya tidak kun ##jung normal ? Pada ##hal sudah ru ##tin min ##um
oba ##t & men ##g ##kon ##sum ##si banyak buah ##2 ##an (tim ##un , sem ##ang
##ka , dl ##l) . Mala ##h badan ##nya ter ##asa lema ##s , pus ##ing & pen
##gli ##hat ##an ka ##bur . Ter ##ima ##kasi ##h . Was ##sala ##mua ##lai ##kum
ð ##ð [SEP]

INFO:tensorflow:tokens: [CLS] Ass ##ala ##mua ##lai ##kum dok , sy mau bertan
##ya . Ibu saya men ##der ##ita hip ##erten ##si . Sud ##ah di ba ##wa ke dokter
dan diberi oba ##t pen ##uru ##n hip ##erten ##si . Ta ##pi ken ##apa ko ##k
tekanan darah ##nya tidak kun ##jung normal ? Pada ##hal sudah ru ##tin min ##um
oba ##t & men ##g ##kon ##sum ##si banyak buah ##2 ##an (tim ##un , sem ##ang
##ka , dl ##l) . Mala ##h badan ##nya ter ##asa lema ##s , pus ##ing & pen
##gli ##hat ##an ka ##bur . Ter ##ima ##kasi ##h . Was ##sala ##mua ##lai ##kum
ð ##ð [SEP]

INFO:tensorflow:input_ids: 101 77014 13322 78314 31181 36811 17674 117 12261
43024 49111 10679 119 68096 64981 10588 11304 11622 25377 26645 10449 119 13352
12257 10120 15688 11037 11163 96140 10215 50479 35355 10123 66558 25279 10115
25377 26645 10449 119 14248 12675 67680 46757 11252 10174 93131 43947 10676
11868 13158 30425 16626 136 12270 18453 25147 13483 15364 13484 10465 35355
10123 111 10588 10240 17423 31417 10449 15175 21988 10729 10206 113 19604 11107
117 11531 11889 10371 117 63940 10161 114 119 58335 10237 51463 10676 12718
23031 93661 10107 117 46960 10230 111 66558 20986 19180 10206 10730 34660 119
65272 12443 37997 10237 119 22034 104655 78314 31181 36811 270 12332 102 0 0 0 0
0
0
0
0 0

INFO:tensorflow:input_ids: 101 77014 13322 78314 31181 36811 17674 117 12261
43024 49111 10679 119 68096 64981 10588 11304 11622 25377 26645 10449 119 13352
12257 10120 15688 11037 11163 96140 10215 50479 35355 10123 66558 25279 10115
25377 26645 10449 119 14248 12675 67680 46757 11252 10174 93131 43947 10676
11868 13158 30425 16626 136 12270 18453 25147 13483 15364 13484 10465 35355
10123 111 10588 10240 17423 31417 10449 15175 21988 10729 10206 113 19604 11107
117 11531 11889 10371 117 63940 10161 114 119 58335 10237 51463 10676 12718
23031 93661 10107 117 46960 10230 111 66558 20986 19180 10206 10730 34660 119
65272 12443 37997 10237 119 22034 104655 78314 31181 36811 270 12332 102 0 0 0 0
0
0
0
0 0

50902 28586 12438 12177 10237 21041 11847 108702 10592 119 89003 32105 21343
64951 46972 10251 11754 11868 10549 17674 136 102 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0
0
0 0 0 0 0 0 0 0 0 0

INFO:tensorflow:input_ids: 101 85566 64981 27763 56440 10233 81565 119 55748 126
10989 12718 10710 16757 64981 28586 42060 10349 10710 63923 23091 10265 11868
16626 119 49056 12055 10679 10349 10710 63923 23091 16626 13908 34384 16113
13549 10911 21114 18234 10347 107329 15334 10349 10896 31288 10347 17229 19049
20879 10659 98336 10706 193 10240 67680 65301 27920 43861 10116 25514 10174
10215 10974 10710 119 65052 25147 91542 10991 42020 64981 12718 23031 44586
10706 10215 10132 57797 11999 32444 18029 10126 92101 10141 14206 119 13026
12476 10174 10132 57797 15907 19554 18029 13377 51193 119 16113 12718 76010
10376 64981 11999 46960 10230 117 10178 11159 12718 23031 11163 110448 117 56944
10161 57177 101833 53538 119 144 28167 10330 193 10240 64981 40796 10116 10592
50902 28586 12438 12177 10237 21041 11847 108702 10592 119 89003 32105 21343
64951 46972 10251 11754 11868 10549 17674 136 102 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0
0
0 0 0 0 0 0 0 0 0 0

INFO:tensorflow:input_mask: 1
1
1
1
1 1 1 1 1 0
0
0 0

INFO:tensorflow:input_mask: 1
1
1
1
1 1 1 1 1 0
0
0 0

INFO:tensorflow:segment_ids: 0
0
0
0
0
0
0 0

INFO:tensorflow:segment_ids: 0
0
0
0 0

##mang sy men ##gida ##p hip ##erten ##si ? A ##paka ##h ada kor ##elas ##i d
##gn ke ##bo ##cora ##n jan ##tung dan tekanan darah tinggi ? [SEP]

INFO:tensorflow:input_ids: 101 67679 17674 117 12261 66558 32556 10410 63923
23091 20506 49167 117 56894 55605 24590 123 12385 46233 111 10794 10174 93131
43947 117 111 31102 10676 16626 113 12718 76010 10376 25377 44073 10449 114 117
46233 10676 23057 178 10240 92287 10237 70733 11471 17502 119 14248 12675 67680
46757 46687 12261 56894 66558 10230 24590 10126 100025 10291 76306 75902 12438
27300 136 138 103208 10237 10592 176 10676 10198 63952 10116 11754 10911 45306
12261 10588 32556 10410 25377 26645 10449 136 138 103208 10237 15290 33705 50567
10116 172 19962 11163 11790 75347 10115 63923 23091 10215 93131 43947 23057 136
102 0
0
0
0 0

INFO:tensorflow:input_ids: 101 67679 17674 117 12261 66558 32556 10410 63923
23091 20506 49167 117 56894 55605 24590 123 12385 46233 111 10794 10174 93131
43947 117 111 31102 10676 16626 113 12718 76010 10376 25377 44073 10449 114 117
46233 10676 23057 178 10240 92287 10237 70733 11471 17502 119 14248 12675 67680
46757 46687 12261 56894 66558 10230 24590 10126 100025 10291 76306 75902 12438
27300 136 138 103208 10237 10592 176 10676 10198 63952 10116 11754 10911 45306
12261 10588 32556 10410 25377 26645 10449 136 138 103208 10237 15290 33705 50567
10116 172 19962 11163 11790 75347 10115 63923 23091 10215 93131 43947 23057 136
102 0
0
0
0 0

INFO:tensorflow:input_mask: 1
1
1
1 1 0
0
0
0 0

INFO:tensorflow:input_mask: 1
1
1
1 1 0
0
0
0 0

INFO:tensorflow:segment_ids: 0
0
0
0
0 0

masuk angin , bu ##ang air dan lain - lain ? Sia ##ng itu me ##mang dia men ##gel ##uhkan Kepala P ##using , Bu ##ang air , mun ##tah , dan dada ses ##ak . Kami kira dia masuk angin , jadi kami kas ##ih tola ##k angin dan lain - lain . Kemudian dia men ##gel ##uhkan , kala ##u tangan ##nya itu dan ja ##ri - ja ##rin ##ya sul ##it dig ##era ##kkan (Dia be ##rp ##iki ##r ken ##a stroke) . Karena masih mua ##l - mua ##l , saya akhirnya pergi untuk amb ##il salon ##pas , dan teman saya men ##jaga ab ##ang saya . La ##lu saya pulang , Ab ##ang saya sudah tidak ada . Kata ##nya sebelum meninggal itu , Dia ke ##jang - ke ##jang luar biasa . Sud ##ah dibawa di rumah sakit , dilakukan CP ##R , namun tidak ter ##sel ##amat ##kan . Beliau me ##mang memiliki ri ##way ##at Darah Ren ##dah . Dari semua ini , apa ##kah ada pen ##jela ##san yang saling berkaitan antara ge ##jala - ge ##jala yang disebut ##kan dia ##tas dok [SEP]

```
INFO:tensorflow:input_ids: 101 67679 85566 117 15595 11889 64981 18049 44725
31585 124 39769 10265 31288 119 52559 12871 51382 10676 76496 85566 10877 10265
71959 33670 10784 38393 10161 11806 23091 119 10685 10237 17674 117 64981 43024
89875 10113 119 89003 118 32105 52559 50274 10486 11806 23091 10265 12729 40796
10116 32500 28977 15290 41585 10676 10659 64951 13908 34675 105676 117 11499
11889 12566 10215 13514 118 13514 136 88468 10376 11910 10911 45306 10671 10588
16039 68637 54087 153 95179 117 11916 11889 12566 117 101833 53538 117 10215
42020 10974 10710 119 87966 32105 10671 34675 105676 117 17760 64985 14399 13187
90470 10174 105676 10215 13514 118 13514 119 52362 10671 10588 16039 68637 117
84844 10138 48371 10676 11910 10215 10201 10401 118 10201 13778 10679 12037
10486 80592 12015 33928 113 18552 10347 33394 20897 10129 67680 10113 57071 114
119 56360 20535 56944 10161 118 56944 10161 117 64981 30448 59159 10782 10559
11030 61658 20084 117 10215 71476 64981 10588 55539 11357 11889 64981 119 10159
11435 64981 107874 117 15595 11889 64981 25147 11868 15290 119 76496 10676 23667
31585 11910 117 18552 11163 37445 118 11163 37445 27120 34384 119 13352 12257
95118 10120 22740 57236 117 28920 40070 11273 117 22736 11868 12718 12912 49158
10706 119 26189 10911 45306 13363 29956 14132 10526 110311 52712 30942 119 35711
23367 10592 117 32500 28977 15290 66558 37142 14434 10265 109002 85783 15345
46503 30216 118 46503 30216 10265 21250 10706 10671 11390 17674 102
```

```
INFO:tensorflow:input_ids: 101 67679 85566 117 15595 11889 64981 18049 44725
31585 124 39769 10265 31288 119 52559 12871 51382 10676 76496 85566 10877 10265
71959 33670 10784 38393 10161 11806 23091 119 10685 10237 17674 117 64981 43024
89875 10113 119 89003 118 32105 52559 50274 10486 11806 23091 10265 12729 40796
10116 32500 28977 15290 41585 10676 10659 64951 13908 34675 105676 117 11499
11889 12566 10215 13514 118 13514 136 88468 10376 11910 10911 45306 10671 10588
16039 68637 54087 153 95179 117 11916 11889 12566 117 101833 53538 117 10215
42020 10974 10710 119 87966 32105 10671 34675 105676 117 17760 64985 14399 13187
90470 10174 105676 10215 13514 118 13514 119 52362 10671 10588 16039 68637 117
84844 10138 48371 10676 11910 10215 10201 10401 118 10201 13778 10679 12037
10486 80592 12015 33928 113 18552 10347 33394 20897 10129 67680 10113 57071 114
119 56360 20535 56944 10161 118 56944 10161 117 64981 30448 59159 10782 10559
11030 61658 20084 117 10215 71476 64981 10588 55539 11357 11889 64981 119 10159
11435 64981 107874 117 15595 11889 64981 25147 11868 15290 119 76496 10676 23667
31585 11910 117 18552 11163 37445 118 11163 37445 27120 34384 119 13352 12257
95118 10120 22740 57236 117 28920 40070 11273 117 22736 11868 12718 12912 49158
```

10706 119 26189 10911 45306 13363 29956 14132 10526 110311 52712 30942 119 35711
 23367 10592 117 32500 28977 15290 66558 37142 14434 10265 109002 85783 15345
 46503 30216 118 46503 30216 10265 21250 10706 10671 11390 17674 102

INFO:tensorflow:input_mask: 1
 1
 1
 1
 1
 1
 1
 1

INFO:tensorflow:input_mask: 1
 1
 1
 1
 1
 1
 1
 1

INFO:tensorflow:segment_ids: 0
 0
 0
 0
 0
 0
 0
 0

INFO:tensorflow:segment_ids: 0
 0
 0
 0
 0
 0
 0
 0

INFO:tensorflow:label: gagal-jantung (id = 2)

INFO:tensorflow:label: gagal-jantung (id = 2)

```
[25]: #Example on first observation in the training set
print("Sentence : ", train_InputExamples.iloc[0].text_a)
print("-"*30)
print("Tokens : ", tokenizer.tokenize(train_InputExamples.iloc[0].text_a))
print("-"*30)
print("Input IDs : ", train_features[0].input_ids)
print("-"*30)
print("Input Masks : ", train_features[0].input_mask)
print("-"*30)
print("Segment IDs : ", train_features[0].segment_ids)
```



```

log_probs = tf.nn.log_softmax(logits, axis=-1)

# Convert labels into one-hot encoding
one_hot_labels = tf.one_hot(labels, depth=num_labels, dtype=tf.float32)

predicted_labels = tf.squeeze(tf.argmax(log_probs, axis=-1, output_type=tf.
→int32))
# If we're predicting, we want predicted labels and the probabilities.
if is_predicting:
    return (predicted_labels, log_probs)

# If we're train/eval, compute loss between predicted and actual label
per_example_loss = -tf.reduce_sum(one_hot_labels * log_probs, axis=-1)
loss = tf.reduce_mean(per_example_loss)
return (loss, predicted_labels, log_probs)

```

```

[27]: #A function that adapts our model to work for training, evaluation, and
→prediction.

# model_fn_builder actually creates our model function
# using the passed parameters for num_labels, learning_rate, etc.
def model_fn_builder(num_labels, learning_rate, num_train_steps,
                    num_warmup_steps):
    """Returns `model_fn` closure for TPUEstimator."""
    def model_fn(features, labels, mode, params): # pylint:
→disable=unused-argument
        """The `model_fn` for TPUEstimator."""

        input_ids = features["input_ids"]
        input_mask = features["input_mask"]
        segment_ids = features["segment_ids"]
        label_ids = features["label_ids"]

        is_predicting = (mode == tf.estimator.ModeKeys.PREDICT)

        # TRAIN and EVAL
        if not is_predicting:

            (loss, predicted_labels, log_probs) = create_model(
                is_predicting, input_ids, input_mask, segment_ids, label_ids,
→num_labels)

            train_op = bert.optimization.create_optimizer(
                loss, learning_rate, num_train_steps, num_warmup_steps, use_tpu=False)

            # Calculate evaluation metrics.
            def metric_fn(label_ids, predicted_labels):

```

```

accuracy = tf.metrics.accuracy(label_ids, predicted_labels)
true_pos = tf.metrics.true_positives(
    label_ids,
    predicted_labels)
true_neg = tf.metrics.true_negatives(
    label_ids,
    predicted_labels)
false_pos = tf.metrics.false_positives(
    label_ids,
    predicted_labels)
false_neg = tf.metrics.false_negatives(
    label_ids,
    predicted_labels)

return {
    "eval_accuracy": accuracy,
    "true_positives": true_pos,
    "true_negatives": true_neg,
    "false_positives": false_pos,
    "false_negatives": false_neg
}

eval_metrics = metric_fn(label_ids, predicted_labels)

if mode == tf.estimator.ModeKeys.TRAIN:
    return tf.estimator.EstimatorSpec(mode=mode,
        loss=loss,
        train_op=train_op)
else:
    return tf.estimator.EstimatorSpec(mode=mode,
        loss=loss,
        eval_metric_ops=eval_metrics)
else:
    (predicted_labels, log_probs) = create_model(
        is_predicting, input_ids, input_mask, segment_ids, label_ids,
        num_labels)

    predictions = {
        'probabilities': log_probs,
        'labels': predicted_labels
    }
    return tf.estimator.EstimatorSpec(mode, predictions=predictions)

# Return the actual model function in the closure
return model_fn

```

```
[28]: # Compute train and warmup steps from batch size
# These hyperparameters are copied from this colab notebook (https://colab.
↳sandbox.google.com/github/tensorflow/tpu/blob/master/tools/colab/
↳bert_finetuning_with_cloud_tpus.ipynb)
BATCH_SIZE = 16
LEARNING_RATE = 2e-5
NUM_TRAIN_EPOCHS = 10
# Warmup is a period of time where the learning rate is small and gradually
↳increases--usually helps training.
WARMUP_PROPORTION = 0.1
# Model configs
SAVE_CHECKPOINTS_STEPS = 300
SAVE_SUMMARY_STEPS = 100

# Compute train and warmup steps from batch size
num_train_steps = int(len(train_features) / BATCH_SIZE * NUM_TRAIN_EPOCHS)
num_warmup_steps = int(num_train_steps * WARMUP_PROPORTION)

# Specify output directory and number of checkpoint steps to save
run_config = tf.estimator.RunConfig(
    model_dir=OUTPUT_DIR,
    save_summary_steps=SAVE_SUMMARY_STEPS,
    save_checkpoints_steps=SAVE_CHECKPOINTS_STEPS)

# Specify output directory and number of checkpoint steps to save
run_config = tf.estimator.RunConfig(
    model_dir=OUTPUT_DIR,
    save_summary_steps=SAVE_SUMMARY_STEPS,
    save_checkpoints_steps=SAVE_CHECKPOINTS_STEPS)
```

```
[29]: #Initializing the model and the estimator
```

```
model_fn = model_fn_builder(
    num_labels=len(label_list),
    learning_rate=LEARNING_RATE,
    num_train_steps=num_train_steps,
    num_warmup_steps=num_warmup_steps)

estimator = tf.estimator.Estimator(
    model_fn=model_fn,
    config=run_config,
    params={"batch_size": BATCH_SIZE})
```

```
INFO:tensorflow:Using config: {'_model_dir': '/GD/My Drive/Colab
Notebooks/LifeHackFinal/', '_tf_random_seed': None, '_save_summary_steps': 100,
'_save_checkpoints_steps': 300, '_save_checkpoints_secs': None,
'_session_config': allow_soft_placement: true
graph_options {
```

```

    rewrite_options {
      meta_optimizer_iterations: ONE
    }
  }
, '_keep_checkpoint_max': 5, '_keep_checkpoint_every_n_hours': 10000,
'_log_step_count_steps': 100, '_train_distribute': None, '_device_fn': None,
'_protocol': None, '_eval_distribute': None, '_experimental_distribute': None,
'_experimental_max_worker_delay_secs': None, '_session_creation_timeout_secs':
7200, '_service': None, '_cluster_spec':
<tensorflow.python.training.server_lib.ClusterSpec object at 0x7f582c2ae240>,
'_task_type': 'worker', '_task_id': 0, '_global_id_in_cluster': 0, '_master':
'', '_evaluation_master': '', '_is_chief': True, '_num_ps_replicas': 0,
'_num_worker_replicas': 1}

INFO:tensorflow:Using config: {'_model_dir': '/GD/My Drive/Colab
Notebooks/LifeHackFinal/', '_tf_random_seed': None, '_save_summary_steps': 100,
'_save_checkpoints_steps': 300, '_save_checkpoints_secs': None,
'_session_config': allow_soft_placement: true
graph_options {
  rewrite_options {
    meta_optimizer_iterations: ONE
  }
}
, '_keep_checkpoint_max': 5, '_keep_checkpoint_every_n_hours': 10000,
'_log_step_count_steps': 100, '_train_distribute': None, '_device_fn': None,
'_protocol': None, '_eval_distribute': None, '_experimental_distribute': None,
'_experimental_max_worker_delay_secs': None, '_session_creation_timeout_secs':
7200, '_service': None, '_cluster_spec':
<tensorflow.python.training.server_lib.ClusterSpec object at 0x7f582c2ae240>,
'_task_type': 'worker', '_task_id': 0, '_global_id_in_cluster': 0, '_master':
'', '_evaluation_master': '', '_is_chief': True, '_num_ps_replicas': 0,
'_num_worker_replicas': 1}

```

we will now create an input builder function that takes our training feature set (`train_features`) and produces a generator. This is a pretty standard design pattern for working with Tensorflow Estimators.

```

[30]: # Create an input function for training. drop_remainder = True for using TPUs.
train_input_fn = bert.run_classifier.input_fn_builder(
    features=train_features,
    seq_length=MAX_SEQ_LENGTH,
    is_training=True,
    drop_remainder=False)

# Create an input function for validating. drop_remainder = True for using TPUs.
val_input_fn = run_classifier.input_fn_builder(
    features=val_features,
    seq_length=MAX_SEQ_LENGTH,
    is_training=False,

```



```
drop_remainder=False)
```

```
## Training & Evaluating
```

```
[31]: #Training the model
print(f'Beginning Training!')
current_time = datetime.now()
estimator.train(input_fn=train_input_fn, max_steps=num_train_steps)
print("Training took time ", datetime.now() - current_time)
```

Beginning Training!

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/tensorflow_core/python/training/training_util.py:236:

Variable.initialized_value (from tensorflow.python.ops.variables) is deprecated and will be removed in a future version.

Instructions for updating:

Use Variable.read_value. Variables in 2.X are initialized automatically both in eager and graph (inside tf.defun) contexts.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/tensorflow_core/python/training/training_util.py:236:

Variable.initialized_value (from tensorflow.python.ops.variables) is deprecated and will be removed in a future version.

Instructions for updating:

Use Variable.read_value. Variables in 2.X are initialized automatically both in eager and graph (inside tf.defun) contexts.

INFO:tensorflow:Calling model_fn.

INFO:tensorflow:Calling model_fn.

INFO:tensorflow:Saver not created because there are no variables in the graph to restore

INFO:tensorflow:Saver not created because there are no variables in the graph to restore

WARNING:tensorflow:From <ipython-input-26-bdfb628bf45b>:33: calling dropout (from tensorflow.python.ops.nn_ops) with keep_prob is deprecated and will be removed in a future version.

Instructions for updating:

Please use `rate` instead of `keep_prob`. Rate should be set to `rate = 1 - keep_prob`.

WARNING:tensorflow:From <ipython-input-26-bdfb628bf45b>:33: calling dropout (from tensorflow.python.ops.nn_ops) with keep_prob is deprecated and will be removed in a future version.

Instructions for updating:

Please use `rate` instead of `keep_prob`. Rate should be set to `rate = 1 - keep_prob`.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/bert/optimization.py:27: The name tf.train.get_or_create_global_step is deprecated. Please use tf.compat.v1.train.get_or_create_global_step instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/bert/optimization.py:27: The name tf.train.get_or_create_global_step is deprecated. Please use tf.compat.v1.train.get_or_create_global_step instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/bert/optimization.py:32: The name tf.train.polynomial_decay is deprecated. Please use tf.compat.v1.train.polynomial_decay instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/bert/optimization.py:32: The name tf.train.polynomial_decay is deprecated. Please use tf.compat.v1.train.polynomial_decay instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/bert/optimization.py:70: The name tf.trainable_variables is deprecated. Please use tf.compat.v1.trainable_variables instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/bert/optimization.py:70: The name tf.trainable_variables is deprecated. Please use tf.compat.v1.trainable_variables instead.

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/tensorflow_core/python/ops/math_grad.py:1375: where (from tensorflow.python.ops.array_ops) is deprecated and will be removed in a future version.

Instructions for updating:

Use tf.where in 2.0, which has the same broadcast rule as np.where

WARNING:tensorflow:From /usr/local/lib/python3.6/dist-packages/tensorflow_core/python/ops/math_grad.py:1375: where (from tensorflow.python.ops.array_ops) is deprecated and will be removed in a future version.

Instructions for updating:

Use tf.where in 2.0, which has the same broadcast rule as np.where

/usr/local/lib/python3.6/dist-packages/tensorflow_core/python/framework/indexed_slices.py:424: UserWarning: Converting sparse IndexedSlices to a dense Tensor of unknown shape. This may consume a large amount of memory.

"Converting sparse IndexedSlices to a dense Tensor of unknown shape. "

INFO:tensorflow:Done calling model_fn.

INFO:tensorflow:Done calling model_fn.

INFO:tensorflow:Create CheckpointSaverHook.
INFO:tensorflow:Create CheckpointSaverHook.
INFO:tensorflow:Graph was finalized.
INFO:tensorflow:Graph was finalized.
INFO:tensorflow:Running local_init_op.
INFO:tensorflow:Running local_init_op.
INFO:tensorflow:Done running local_init_op.
INFO:tensorflow:Done running local_init_op.
INFO:tensorflow:Saving checkpoints for 0 into /GD/My Drive/Colab Notebooks/LifeHackFinal/model.ckpt.
INFO:tensorflow:Saving checkpoints for 0 into /GD/My Drive/Colab Notebooks/LifeHackFinal/model.ckpt.
INFO:tensorflow:loss = 1.5817094, step = 0
INFO:tensorflow:loss = 1.5817094, step = 0
INFO:tensorflow:global_step/sec: 0.929525
INFO:tensorflow:global_step/sec: 0.929525
INFO:tensorflow:loss = 1.4598477, step = 100 (107.585 sec)
INFO:tensorflow:loss = 1.4598477, step = 100 (107.585 sec)
INFO:tensorflow:global_step/sec: 1.05716
INFO:tensorflow:global_step/sec: 1.05716
INFO:tensorflow:loss = 1.0209707, step = 200 (94.592 sec)
INFO:tensorflow:loss = 1.0209707, step = 200 (94.592 sec)
INFO:tensorflow:Saving checkpoints for 300 into /GD/My Drive/Colab Notebooks/LifeHackFinal/model.ckpt.
INFO:tensorflow:Saving checkpoints for 300 into /GD/My Drive/Colab Notebooks/LifeHackFinal/model.ckpt.
INFO:tensorflow:global_step/sec: 0.810489
INFO:tensorflow:global_step/sec: 0.810489
INFO:tensorflow:loss = 0.940634, step = 300 (123.382 sec)
INFO:tensorflow:loss = 0.940634, step = 300 (123.382 sec)
INFO:tensorflow:global_step/sec: 1.05215
INFO:tensorflow:global_step/sec: 1.05215
INFO:tensorflow:loss = 0.32010147, step = 400 (95.042 sec)

```

INFO:tensorflow:loss = 0.32010147, step = 400 (95.042 sec)
INFO:tensorflow:global_step/sec: 1.05385
INFO:tensorflow:global_step/sec: 1.05385
INFO:tensorflow:loss = 0.15604638, step = 500 (94.891 sec)
INFO:tensorflow:loss = 0.15604638, step = 500 (94.891 sec)
INFO:tensorflow:Saving checkpoints for 600 into /GD/My Drive/Colab
Notebooks/LifeHackFinal/model.ckpt.
INFO:tensorflow:Saving checkpoints for 600 into /GD/My Drive/Colab
Notebooks/LifeHackFinal/model.ckpt.
INFO:tensorflow:global_step/sec: 0.810798
INFO:tensorflow:global_step/sec: 0.810798
INFO:tensorflow:loss = 0.25885272, step = 600 (123.334 sec)
INFO:tensorflow:loss = 0.25885272, step = 600 (123.334 sec)
INFO:tensorflow:Saving checkpoints for 650 into /GD/My Drive/Colab
Notebooks/LifeHackFinal/model.ckpt.
INFO:tensorflow:Saving checkpoints for 650 into /GD/My Drive/Colab
Notebooks/LifeHackFinal/model.ckpt.
INFO:tensorflow:Loss for final step: 0.050609734.
INFO:tensorflow:Loss for final step: 0.050609734.
Training took time 0:12:59.069056

```

```

[32]: #Evaluating the model with Validation set
      eval_results = estimator.evaluate(input_fn=val_input_fn, steps=None)

```

```

INFO:tensorflow:Calling model_fn.
INFO:tensorflow:Calling model_fn.
INFO:tensorflow:Saver not created because there are no variables in the graph to
restore
INFO:tensorflow:Saver not created because there are no variables in the graph to
restore
/usr/local/lib/python3.6/dist-
packages/tensorflow_core/python/framework/indexed_slices.py:424: UserWarning:
Converting sparse IndexedSlices to a dense Tensor of unknown shape. This may
consume a large amount of memory.
    "Converting sparse IndexedSlices to a dense Tensor of unknown shape. "
INFO:tensorflow:Done calling model_fn.
INFO:tensorflow:Done calling model_fn.

```

```

INFO:tensorflow:Starting evaluation at 2020-07-29T18:07:24Z
INFO:tensorflow:Starting evaluation at 2020-07-29T18:07:24Z
INFO:tensorflow:Graph was finalized.
INFO:tensorflow:Graph was finalized.
INFO:tensorflow:Restoring parameters from /GD/My Drive/Colab
Notebooks/LifeHackFinal/model.ckpt-650
INFO:tensorflow:Restoring parameters from /GD/My Drive/Colab
Notebooks/LifeHackFinal/model.ckpt-650
INFO:tensorflow:Running local_init_op.
INFO:tensorflow:Running local_init_op.
INFO:tensorflow:Done running local_init_op.
INFO:tensorflow:Done running local_init_op.
INFO:tensorflow:Finished evaluation at 2020-07-29-18:07:48
INFO:tensorflow:Finished evaluation at 2020-07-29-18:07:48
INFO:tensorflow:Saving dict for global step 650: eval_accuracy = 0.7653846,
false_negatives = 10.0, false_positives = 11.0, global_step = 650, loss =
0.82353604, true_negatives = 44.0, true_positives = 195.0
INFO:tensorflow:Saving dict for global step 650: eval_accuracy = 0.7653846,
false_negatives = 10.0, false_positives = 11.0, global_step = 650, loss =
0.82353604, true_negatives = 44.0, true_positives = 195.0
INFO:tensorflow:Saving 'checkpoint_path' summary for global step 650: /GD/My
Drive/Colab Notebooks/LifeHackFinal/model.ckpt-650
INFO:tensorflow:Saving 'checkpoint_path' summary for global step 650: /GD/My
Drive/Colab Notebooks/LifeHackFinal/model.ckpt-650

```

```
[33]: eval_results
```

```
[33]: {'eval_accuracy': 0.7653846,
      'false_negatives': 10.0,
      'false_positives': 11.0,
      'global_step': 650,
      'loss': 0.82353604,
      'true_negatives': 44.0,
      'true_positives': 195.0}
```

```
[34]: predictions = estimator.predict(val_input_fn)
```

```
[35]: preds_result = []
      for prediction in predictions:
          preds_result.append((prediction['probabilities'], prediction['labels']))
```

```

INFO:tensorflow:Calling model_fn.
INFO:tensorflow:Calling model_fn.
INFO:tensorflow:Saver not created because there are no variables in the graph to
restore
INFO:tensorflow:Saver not created because there are no variables in the graph to
restore
INFO:tensorflow:Done calling model_fn.
INFO:tensorflow:Done calling model_fn.
INFO:tensorflow:Graph was finalized.
INFO:tensorflow:Graph was finalized.
INFO:tensorflow:Restoring parameters from /GD/My Drive/Colab
Notebooks/LifeHackFinal/model.ckpt-650
INFO:tensorflow:Restoring parameters from /GD/My Drive/Colab
Notebooks/LifeHackFinal/model.ckpt-650
INFO:tensorflow:Running local_init_op.
INFO:tensorflow:Running local_init_op.
INFO:tensorflow:Done running local_init_op.
INFO:tensorflow:Done running local_init_op.

```

```
[36]: y_pred = list(map(lambda x: x[1], preds_result))
```

```
[37]: mapping = dict()

for i in range(len(label_list)):
    mapping[label_list[i]] = i

y_actual = list(map(lambda x: mapping[x], val['category'].tolist()))
```

```
[38]: from sklearn.metrics import confusion_matrix

confusion_matrix(y_actual, y_pred)
```

```
[38]: array([[44,  1,  5,  4,  1],
            [ 0, 45,  6,  1,  7],
            [ 2,  0, 39,  3,  5],
            [ 5,  0,  4, 41,  0],
            [ 2,  4, 10,  0, 31]])
```

```
[39]: val_pred = val.copy()
val_pred['pred'] = list(map(lambda x: label_list[x], y_pred))
val_pred.to_csv('prediction_final_raw.csv')
```

```
[42]: from sklearn.metrics import classification_report
print(classification_report(val_pred['category'], val_pred['pred']))
```

	precision	recall	f1-score	support
aritmia	0.90	0.76	0.83	59
gagal-jantung	0.61	0.80	0.69	49
hipertensi	0.83	0.80	0.81	55
serangan-jantung	0.70	0.66	0.68	47
stroke	0.84	0.82	0.83	50
accuracy			0.77	260
macro avg	0.78	0.77	0.77	260
weighted avg	0.78	0.77	0.77	260

```
[40]: val_pred.head()
```

```
[40]:
```

	category	...	pred
1001	serangan-jantung	...	aritmia
1266	stroke	...	stroke
503	gagal-jantung	...	serangan-jantung
756	hipertensi	...	hipertensi
459	gagal-jantung	...	stroke

```
[5 rows x 7 columns]
```

#Reference: Most of the code has been taken from the following resource:

- https://colab.research.google.com/github/google-research/bert/blob/master/predicting_movie_reviews_w