# Questions on the existing code:

## **Q1:**

def tokenize(model\_name):

try:

tokenizer = BertTokenizer.from\_pretrained(model\_name, do\_lower\_case=True)

except:

tokenizer = AutoTokenizer.from\_pretrained(model\_name, do\_lower\_case=True)

return tokenizer

**Why should it try BertTokenizer first and not directly use AutoTokenizer?**

**Ans:** Last time few models were run with BertTokenizer and other ones with Autotokenizer. Now, by updating the transformer library, I verify that all the model can be trained AutoTokenizer. So, your new tokenizer function is:

**def tokenize(model\_name):**

**tokenizer = AutoTokenizer.from\_pretrained(model\_name, do\_lower\_case=True)**

**return tokenizer**

### **Q2:**

df = pickle.load(open('dataset/230130\_SmallOberkategorie.pickle', 'rb'))

label\_names = df['labels'].unique()

# label\_names

X = df[['text']]

y = df['labels']

X\_train, X\_val, y\_train, y\_val = split\_df(X, y)

optimizers = [Adam, SGD]

configs = {'epochs': 2, 'batch\_size': 16, 'learning\_rate': 2e-6, 'val\_steps':100}

dataset = {'X\_train': X\_train, 'y\_train': y\_train, 'X\_val': X\_val, 'y\_val': y\_val}

models = ['Bert-base-german-cased',

#'Dbmdz/bert-base-german-uncased',

# 'Deepset/gbert-base',

# 'Xlm-roberta-base',

# 'Uklfr/gottbert-base'

]

training(models, optimizers, configs, dataset)

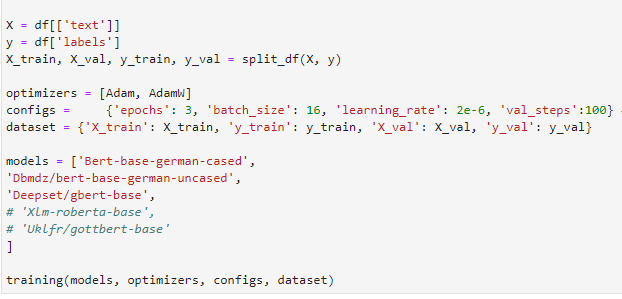
**Why do we import the data again? Once we import it at the beginning and then when training starts. Is there a reason?**

**Ans:** yes, it is just for verification purpose. Sometimes we perform few transformations in our dataset to analyze it. That’s why here we read the fresh copy of the dataset.

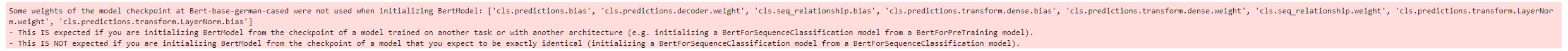
Anyhow you can delete the first lines the dataset but not remove the reading dataset lines in training cell P2. (My suggestion)

## **Q3:**

When starting the training with this cell



This message appears:

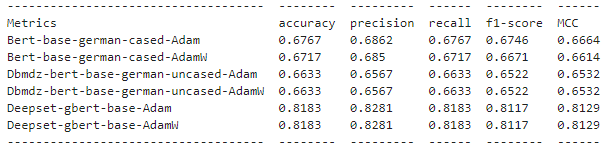


This suggests that something is not right when the model is build, because the model is used for SequenceClassification and the model checkpoint should a BERT model for SequenceClassification checkpoint, no?

**Ans:** I will see this in your extended code and fix this if it will be a problematic issue.

## **Q4:**

Is AdamW correctly imported? There are basically no differences between Adam and AdamW. Does AdamW need further parameters?



**Ans:** No, there is further parameters are required. Basically, Adam and AdamW are quite similar in many cases. That’s why the results of both optimizers are similar. However, I used these optimizers here for just demo purposes. You can just import the any categorical base optimizer and pass in the list of optimizers in P2.

**Note:** On the next page, I am writing a short overview of both optimizers that may be helpful for you.

Adam and AdamW are both optimization algorithms used in training deep neural networks, with AdamW being an extension of Adam.

The primary difference between Adam and AdamW lies in how they handle weight decay regularization. Adam applies weight decay during each update step, which can lead to suboptimal convergence when applied to certain kinds of models. AdamW, on the other hand, decouples weight decay from the optimization step, allowing the learning rate to adapt to the parameters being updated while applying weight decay separately to each parameter.

In other words, AdamW applies weight decay after updating the weights, whereas Adam applies weight decay before the update. This difference can lead to better performance for AdamW in some cases, especially when dealing with large neural networks or when using high learning rates.

Overall, the choice between Adam and AdamW depends on the specific requirements and characteristics of the deep learning model being trained.