# Project Idea

Make up a text classification model, classifying student's responses into various rubric/requirements, this makes it easier for the professor/graders to make a gist of the submission and see the submission divided into the required goals for the assignment. The dataset used here portrays essays from the IELTS writing section, and the need is to classify the essays into three rubrics being used for the grading of these essays, i.e., Task Response. coherence and correlation and Lexical resources (Vocubulary)

In [1]:
```
# performing imports
# python libraries
import os
import string
import re
from IPython.display import display, HTML
import warnings
import tqdm
from typing import List, Tuple, Dict

# Basic ML libraries and scikit learn libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import sklearn as skl
from sklearn.cluster import KMeans
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MultiLabelBinarizer
from matplotlib.colors import ListedColormap

# LDA libraries
import nltk
import gensim.corpora as corpora
import gensim
import gensim.models
from gensim.models import CoherenceModel
from gensim .models import TfidfModel
import spacy
import pyLDAvis
import pyLDAvis.gensim_models

# Bert and Pytorch libraries
import torch
import accelerate
from nltk.stem import WordNetLemmatizer
from nltk.tokenize import sent_tokenize
from torch.utils.data import DataLoader
from datasets import Dataset, Features, Value, Sequence
from transformers import BertTokenizer, BertForSequenceClassification, Trair
```

In [2]:
```
# Getting current working directory
cwd = os.getcwd()
cwd
```
Out[2]:
```
'D:\\My Work\\Computer Science Courses and Projects\\EMP_Work'
```

## Task-1: Read the Dataset and manually label Task Response, Coherence and Lexical Resource

In [3]:
```
#Read CSV
Essay = pd.read_csv("ielts_writing_dataset.csv")
```
In [4]:
```
Essay.head()
```
Out[4]:

| | Task_Type | Question | Essay | Examiner_Commen | Task_Response | Coher |
|---|---|---|---|---|---|---|
| **0** | 1 | The bar chart below describes some changes abo... | Between 1995 and 2010, a study was conducted r... | NaN | NaN | |
| **1** | 2 | Rich countries often give money to poorer coun... | Poverty represents a worldwide crisis. It is t... | NaN | NaN | |
| **2** | 1 | The bar chart below describes some changes abo... | The left chart shows the population change hap... | NaN | NaN | |
| **3** | 2 | Rich countries often give money to poorer coun... | Human beings are facing many challenges nowada... | NaN | NaN | |
| **4** | 1 | The graph below shows the number of overseas v... | Information about the thousands of visits from... | NaN | NaN | |

In [5]:
```
#Getting rid of some unwanted columns, Dropping Task_Type and Question, as w
Essay = Essay.drop(columns=['Task_Type','Question','Examiner_Commen','Range_
Essay
```

Out[5]:

| | Essay | Task_Response | Coherence_Cohesion | Lexical_Resource |
|---|---|---|---|---|
| **0** | Between 1995 and 2010, a study was conducted r... | NaN | NaN | NaN |
| **1** | Poverty represents a worldwide crisis. It is t... | NaN | NaN | NaN |
| **2** | The left chart shows the population change hap... | NaN | NaN | NaN |
| **3** | Human beings are facing many challenges nowada... | NaN | NaN | NaN |
| **4** | Information about the thousands of visits from... | NaN | NaN | NaN |
| **...** | ... | ... | ... | ... |
| **1430** | Serious crimes need capital punishment so that... | NaN | NaN | NaN |
| **1431** | It is certainly said that learning is an ongoi... | NaN | NaN | NaN |
| **1432** | popular hobbies rather than their individual a... | NaN | NaN | NaN |
| **1433** | Yes, I do feel that universities should have a... | NaN | NaN | NaN |
| **1434** | The modern medicine is very important for livi... | NaN | NaN | NaN |

1435 rows × 4 columns

```
In [6]: # Split the dataset into training and testing
        labeled_df, Test_Essays = train_test_split(Essay, test_size=0.2, random_stat
```

```
In [7]: import pandas as pd
        import re

        # Expanded Keyword Lists
        task_response_keywords = [
            'overall', 'in conclusion', 'to sum up', 'summary', 'to conclude', 'as a
            'overview', 'main trend', 'key trend', 'general pattern', 'it is clear t
            'it can be seen', 'it is evident that', 'the graph shows', 'the chart il
            'according to the data', 'the data suggest', 'depicts', 'illustrates', '
            'describes', 'in general', 'over the period', 'as illustrated', 'in the 
            'the linear figure', 'bar chart', 'line graph', 'pie chart', 'diagram', 
            'to summarize', 'as mentioned above', 'as stated earlier', 'mainly', 'it
            'the results show', 'the data indicates', 'the chart highlights', 'the r
            'the findings suggest', 'the data reveals', 'the trend suggests', 'the f
            'the diagram shows', 'from the data, it is clear that', 'the graph indic
        ]

        coherence_cohesion_keywords = [
            'firstly', 'secondly', 'thirdly', 'then', 'after that', 'before that', '
            'finally', 'in contrast', 'however', 'on the other hand', 'although', 'm
            'while', 'similarly', 'subsequently', 'consequently', 'following this', 
            'in addition', 'additionally', 'furthermore', 'moreover', 'at the start'
            'in detail', 'at the end', 'identical pattern', 'continuous pattern', 't
            'following this', 'alternatively', 'equally', 'more importantly'
        ]

        lexical_resource_keywords = [
            'dramatically', 'sharply', 'steadily', 'significantly', 'substantially',
            'slightly', 'gradually', 'rapidly', 'moderately', 'plummet', 'soar',
            'rocket', 'decline', 'decrease', 'increase', 'peak', 'drop', 'rise',
            'fluctuate', 'stabilize', 'recovered', 'plummeted', 'rose', 'dipped',
            'trends', 'figures', 'statistics', 'growth', 'fall', 'surge', 'upward', 
            'increase in', 'decrease in', 'rise in', 'drop in', 'plunge', 'boom', 's
            'significant rise', 'significant fall', 'sharp increase', 'sharp decline
            'level off', 'stabilize', 'maintain', 'rapid growth', 'slow growth', 'mi
            'stable', 'constant', 'unchanged', 'proportion', 'percentage', 'rate', '
            'fall rate', 'upward trend', 'downward trend', 'figures', 'pattern', 'tr
        ]

        # 2. Regex sentence splitter
        def simple_sentence_split(text):
            sentences = re.split(r'(?<=[.!?]) +', str(text))  # In case of NaN
            return sentences

        # 3. Labeling function
        def label_sentence(sentence):
            sentence_lower = sentence.lower()
            labels = {'Task_Response': 0, 'Coherence_Cohesion': 0, 'Lexical_Resource

            if any(keyword in sentence_lower for keyword in task_response_keywords):
                labels['Task_Response'] = 1
```

```
    if any(keyword in sentence_lower for keyword in coherence_cohesion_keywo
        labels['Coherence_Cohesion'] = 1

    if any(keyword in sentence_lower for keyword in lexical_resource_keyword
        labels['Lexical_Resource'] = 1

    return labels


# 5. Create DataFrame to store all labeled sentences
labeled_data = []

# 6. Process each essay
for idx, essay_text in enumerate(labeled_df["Essay"]):
    sentences = simple_sentence_split(essay_text)
    for sent in sentences:
        if sent.strip() == "":
            continue  # Skip empty sentences
        labels = label_sentence(sent)
        labeled_data.append({
            'Essay_ID': idx,
            'Sentence': sent.strip(),
            'Task_Response': labels['Task_Response'],
            'Coherence_Cohesion': labels['Coherence_Cohesion'],
            'Lexical_Resource': labels['Lexical_Resource'],
        })

# Final labeled dataset
labeled_df = pd.DataFrame(labeled_data)


# 8. Show
print(labeled_df.head())

# 9. Optional: Save to new CSV
# labeled_df.to_csv('labeled_sentences.csv', index=False)
```

```
   Essay_ID                                          Sentence  Task_Respons
e  \
0         0  Usage of drugs is now common in our contempora...
0
1         0  In this essay, i am going to discuss the cause...
0
2         0  A lot of youths today, learnt this lifestyle f...
0
3         0  For example, a teenager whose parents take coc...
0
4         0  The chemical components of this drug are incre...
0

   Coherence_Cohesion  Lexical_Resource
0                   0                 0
1                   0                 1
2                   0                 0
3                   0                 1
4                   0                 0
```

```
In [8]: labeled_df
```

Out[8]:

| | Essay_ID | Sentence | Task_Response | Coherence_Cohesion | Lexical_R |
|---|---|---|---|---|---|
| **0** | 0 | Usage of drugs is now common in our contempora... | 0 | 0 | |
| **1** | 0 | In this essay, i am going to discuss the cause... | 0 | 0 | |
| **2** | 0 | A lot of youths today, learnt this lifestyle f... | 0 | 0 | |
| **3** | 0 | For example, a teenager whose parents take coc... | 0 | 0 | |
| **4** | 0 | The chemical components of this drug are incre... | 0 | 0 | |
| **...** | ... | ... | ... | ... | |
| **10569** | 1146 | In a contrast, a continuous decline on purchas... | 0 | 0 | |
| **10570** | 1147 | The bar chart illustrates information on the n... | 1 | 1 | |
| **10571** | 1147 | Also, the demand response for accidents was mo... | 0 | 1 | |
| **10572** | 1147 | Although, the number of recorded injuries in t... | 0 | 1 | |
| **10573** | 1147 | million PMT. | 0 | 0 | |

10574 rows × 5 columns

```
In [9]: import pandas as pd

        # Assuming you have the 'labeled_df' DataFrame loaded
        # Filter out rows where all the specified columns have a value of zero
```

```
filtered_df = labeled_df[
    (labeled_df['Task_Response'] != 0) |
    (labeled_df['Coherence_Cohesion'] != 0) |
    (labeled_df['Lexical_Resource'] != 0)
]

labeled_df = filtered_df.reset_index(drop=True)
```

In [10]:
```
labeled_df
```

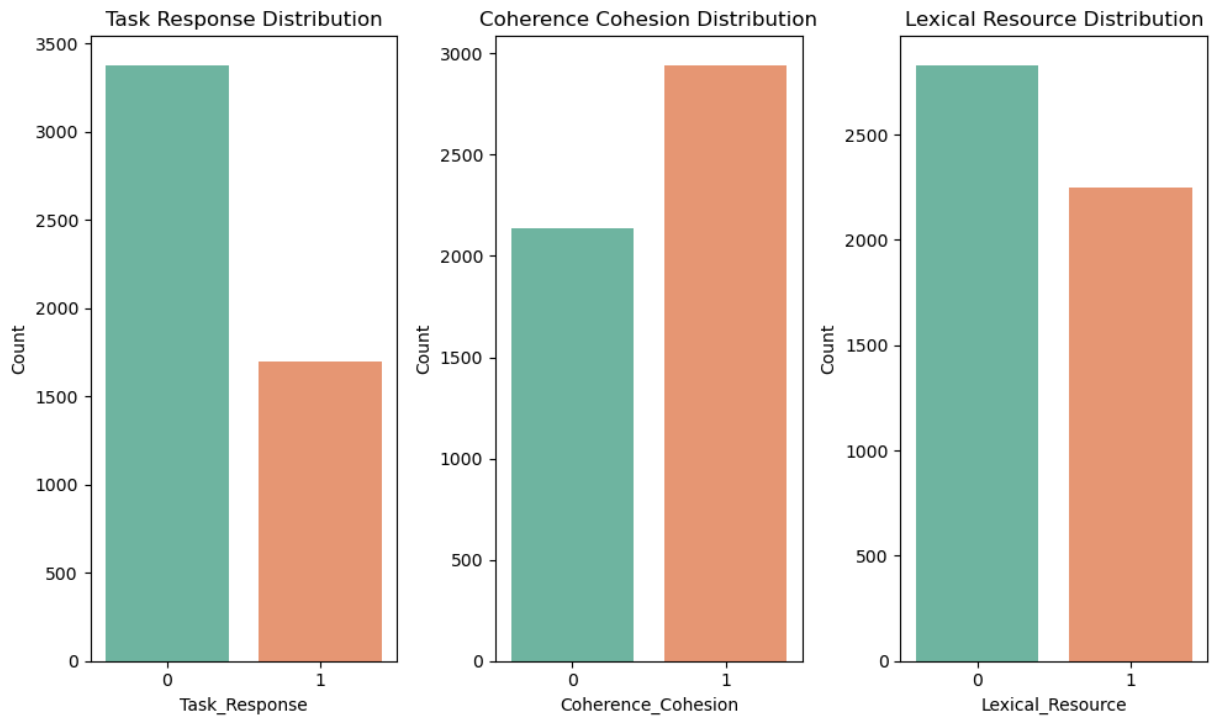
Out[10]:

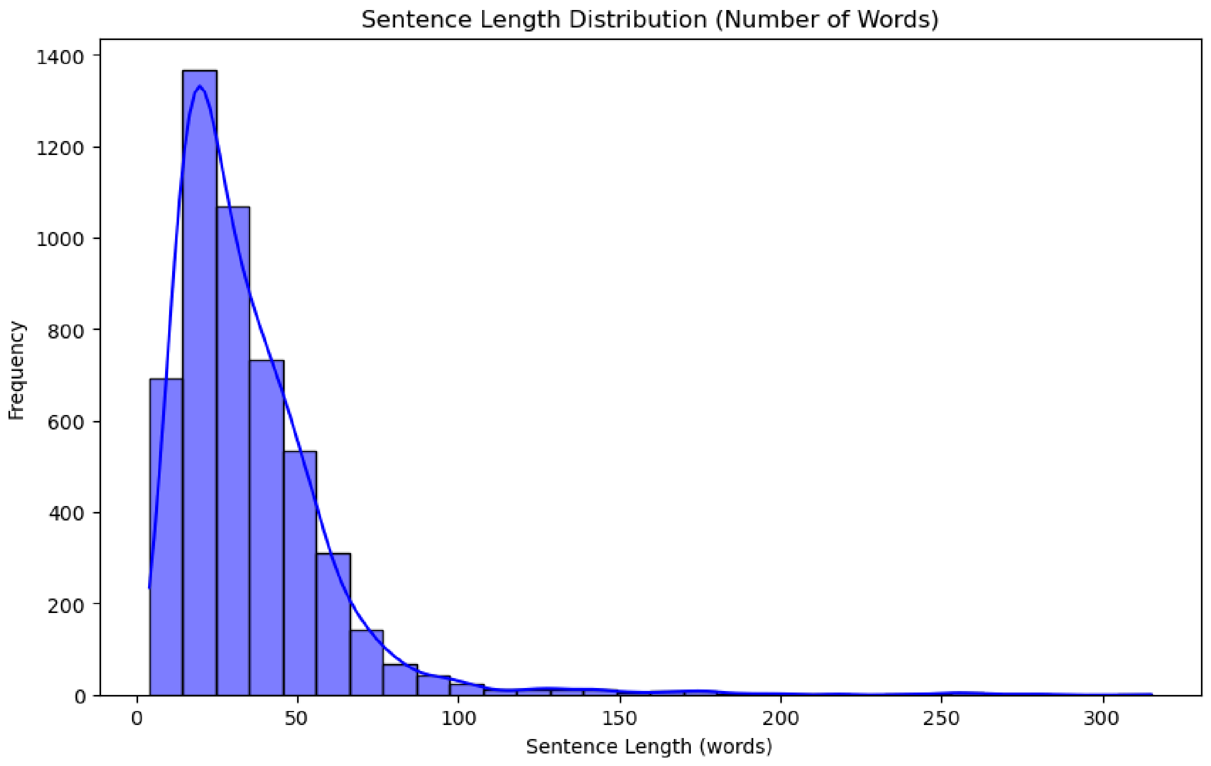
| | Essay_ID | Sentence | Task_Response | Coherence_Cohesion | Lexica |
|---|---|---|---|---|---|
| **0** | 0 | In this essay, i am going to discuss the cause... | 0 | 0 | |
| **1** | 0 | For example, a teenager whose parents take coc... | 0 | 0 | |
| **2** | 0 | A recent study, showed that 75% of young adult... | 1 | 0 | |
| **3** | 0 | Psychotherapy should also be recommended.\nTo ... | 1 | 1 | |
| **4** | 1 | The provided barr chart depicts the proportion... | 1 | 0 | |
| **...** | ... | ... | ... | ... | |
| **5070** | 1146 | Then, a sharp increase in the consumption of h... | 0 | 1 | |
| **5071** | 1146 | In a contrast, a continuous decline on purchas... | 0 | 0 | |
| **5072** | 1147 | The bar chart illustrates information on the n... | 1 | 1 | |
| **5073** | 1147 | Also, the demand response for accidents was mo... | 0 | 1 | |
| **5074** | 1147 | Although, the number of recorded injuries in t... | 0 | 1 | |

5075 rows × 5 columns

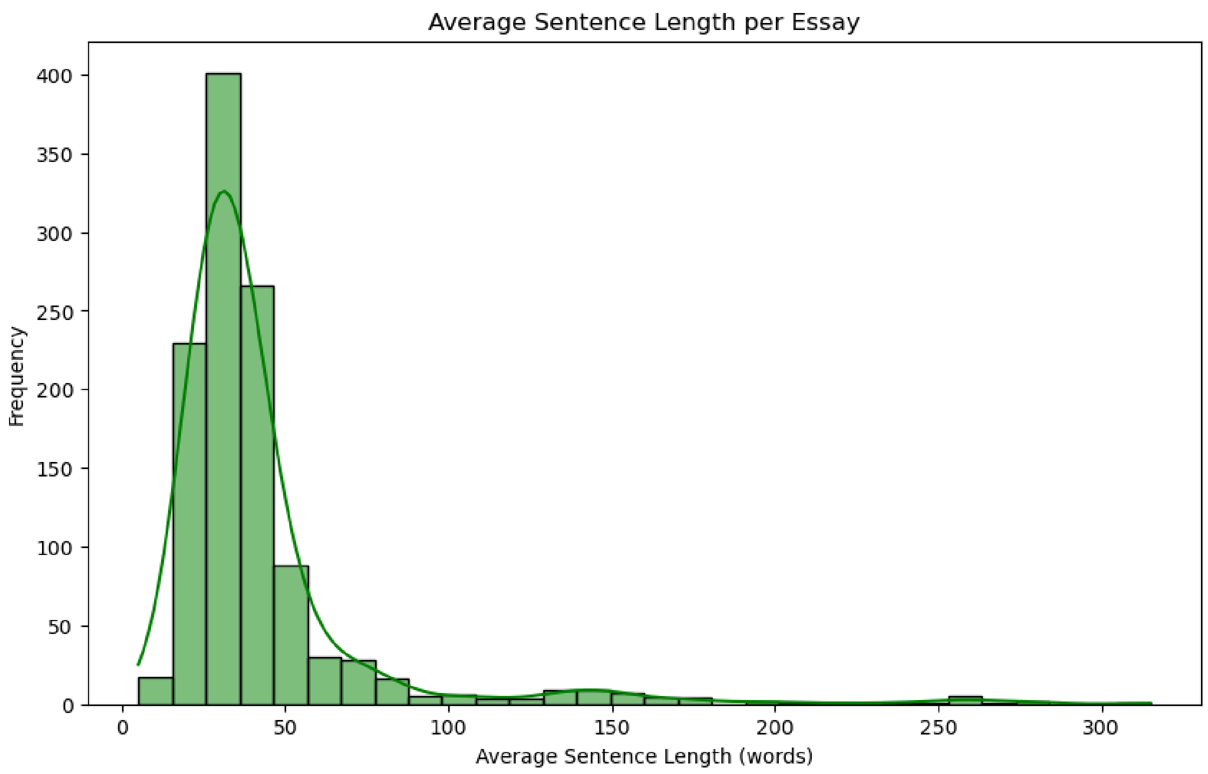
In [11]:
```
print(labeled_df.dtypes)
```
```
Essay_ID                 int64
Sentence                object
Task_Response            int64
Coherence_Cohesion       int64
Lexical_Resource         int64
dtype: object
```
In [12]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Assuming labeled_df is already defined
def analyze_labeled_df(labeled_df):
    # 1. Class Distribution of Task_Response, Coherence_Cohesion, Lexical_Re
    plt.figure(figsize=(10, 6))

    # Plot Task_Response distribution
    plt.subplot(131)
    sns.countplot(data=labeled_df, x='Task_Response', palette='Set2')
    plt.title('Task Response Distribution')
    plt.xlabel('Task_Response')
    plt.ylabel('Count')

    # Plot Coherence_Cohesion distribution
    plt.subplot(132)
    sns.countplot(data=labeled_df, x='Coherence_Cohesion', palette='Set2')
    plt.title('Coherence Cohesion Distribution')
    plt.xlabel('Coherence_Cohesion')
    plt.ylabel('Count')

    # Plot Lexical_Resource distribution
    plt.subplot(133)
    sns.countplot(data=labeled_df, x='Lexical_Resource', palette='Set2')
    plt.title('Lexical Resource Distribution')
    plt.xlabel('Lexical_Resource')
    plt.ylabel('Count')

    plt.tight_layout()
    plt.show()

    # 2. Essay Sentence Length Distribution (word count per sentence)
    labeled_df['sentence_length'] = labeled_df['Sentence'].apply(lambda x: l

    # Plot distribution of sentence lengths
    plt.figure(figsize=(10, 6))
    sns.histplot(labeled_df['sentence_length'], kde=True, color='blue', bins
    plt.title('Sentence Length Distribution (Number of Words)')
    plt.xlabel('Sentence Length (words)')
    plt.ylabel('Frequency')
    plt.show()

    # 3. Missing Values
    missing_values = labeled_df.isnull().sum()
    print(f"Missing Values in each column:\n{missing_values}")
    # 4. Aggregate essay-level statistics:
    # Aggregating by Essay_ID: Calculating average sentence length per essay
    essay_stats = labeled_df.groupby('Essay_ID')['sentence_length'].agg(['me

    # Plot the average sentence length for each essay
    plt.figure(figsize=(10, 6))
    sns.histplot(essay_stats['mean'], kde=True, color='green', bins=30)
    plt.title('Average Sentence Length per Essay')
    plt.xlabel('Average Sentence Length (words)')
    plt.ylabel('Frequency')
    plt.show()

    # 5. Sample Essays
    print("Sample Essays and Their Labels:")
    print(labeled_df[['Essay_ID', 'Sentence', 'Task_Response', 'Coherence_Co

# Perform analysis
analyze_labeled_df(labeled_df)
```

```
C:\Users\ritvi\AppData\Local\Temp\ipykernel_31648\4035380066.py:12: FutureWa
rning:

Passing `palette` without assigning `hue` is deprecated and will be removed
in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the
same effect.

  sns.countplot(data=labeled_df, x='Task_Response', palette='Set2')
C:\Users\ritvi\AppData\Local\Temp\ipykernel_31648\4035380066.py:19: FutureWa
rning:

Passing `palette` without assigning `hue` is deprecated and will be removed
in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the
same effect.

  sns.countplot(data=labeled_df, x='Coherence_Cohesion', palette='Set2')
C:\Users\ritvi\AppData\Local\Temp\ipykernel_31648\4035380066.py:26: FutureWa
rning:

Passing `palette` without assigning `hue` is deprecated and will be removed
in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the
same effect.

  sns.countplot(data=labeled_df, x='Lexical_Resource', palette='Set2')
```





```
Missing Values in each column:
Essay_ID              0
Sentence              0
Task_Response         0
Coherence_Cohesion    0
Lexical_Resource      0
sentence_length       0
dtype: int64
```



```
Sample Essays and Their Labels:
   Essay_ID                                           Sentence  Task_Respons
e  \
0         0  In this essay, i am going to discuss the cause...
0
1         0  For example, a teenager whose parents take coc...
0
2         0  A recent study, showed that 75% of young adult...
1
3         0  Psychotherapy should also be recommended.\nTo ...
1
4         1  The provided barr chart depicts the proportion...
1

   Coherence_Cohesion  Lexical_Resource
0                   0                 1
1                   0                 1
2                   0                 1
3                   1                 1
4                   0                 1
```

## Train first transformer (For better context extraction)

In [13]:
```python
from sklearn.model_selection import train_test_split
from transformers import BertTokenizer, BertForSequenceClassification, Trair
import torch
import pandas as pd


# Split the dataset into training and testing
train_df, test_df = train_test_split(labeled_df, test_size=0.2, random_state
# Load tokenizer and pre-trained BERT model
tokenizer = BertTokenizer.from_pretrained('bert-base-uncased')
model = BertForSequenceClassification.from_pretrained('bert-base-uncased', r

# Define Dataset class
class EssayDataset(torch.utils.data.Dataset):
    def __init__(self, dataframe, tokenizer, max_length):
        self.dataframe = dataframe
        self.tokenizer = tokenizer
        self.max_length = max_length

    def __len__(self):
        return len(self.dataframe)

    def __getitem__(self, idx):
        sentence = self.dataframe.iloc[idx]['Sentence']

        # Tokenize the sentence and truncate/pad to max_length
        encoding = self.tokenizer.encode_plus(
            sentence,
            add_special_tokens=True,  # Add [CLS] and [SEP]
            max_length=self.max_length,
            padding='max_length',
            truncation=True,
            return_attention_mask=True,  # Attention mask to focus on actual
            return_tensors='pt',  # Return PyTorch tensors
        )

        # Prepare the labels
        task_response = torch.tensor(self.dataframe.iloc[idx]['Task_Response
        coherence_cohesion = torch.tensor(self.dataframe.iloc[idx]['Coherenc
        lexical_resource = torch.tensor(self.dataframe.iloc[idx]['Lexical_Re

        label = torch.stack([task_response, coherence_cohesion, lexical_reso

        return {
            'input_ids': encoding['input_ids'].flatten(),
            'attention_mask': encoding['attention_mask'].flatten(),
            'labels': label
        }

# Create DataLoader for batching
train_dataset = EssayDataset(dataframe=train_df, tokenizer=tokenizer, max_le
train_dataloader = torch.utils.data.DataLoader(train_dataset, batch_size=4,

test_dataset = EssayDataset(dataframe=test_df, tokenizer=tokenizer, max_leng
test_dataloader = torch.utils.data.DataLoader(test_dataset, batch_size=4, sh




# Set up Trainer with BERT model
training_args = TrainingArguments(
    output_dir='./results',
    num_train_epochs=1,
    per_device_train_batch_size=4,
    logging_dir='./logs',
)

trainer = Trainer(
    model=model,                         # The model to train
    args=training_args,                  # Training arguments
    train_dataset=train_dataset,         # Training dataset
    eval_dataset=test_dataset,           # Evaluation dataset
)

# Fine-tune the model
trainer.train()

# Save the fine-tuned BERT model
model.save_pretrained('./fine_tuned_bert')
tokenizer.save_pretrained('./fine_tuned_bert')
```

```
C:\Users\ritvi\miniconda3\envs\EMP_env\lib\site-packages\huggingface_hub\fil
e_download.py:1150: FutureWarning: `resume_download` is deprecated and will
be removed in version 1.0.0. Downloads always resume when possible. If you w
ant to force a new download, use `force_download=True`.
  warnings.warn(
Some weights of the model checkpoint at bert-base-uncased were not used when
initializing BertForSequenceClassification: ['cls.predictions.transform.Laye
rNorm.bias', 'cls.predictions.bias', 'cls.predictions.transform.dense.weigh
t', 'cls.seq_relationship.weight', 'cls.predictions.transform.dense.bias',
'cls.predictions.transform.LayerNorm.weight', 'cls.seq_relationship.bias']
- This IS expected if you are initializing BertForSequenceClassification fro
m the checkpoint of a model trained on another task or with another architec
ture (e.g. initializing a BertForSequenceClassification model from a BertFor
PreTraining model).
- This IS NOT expected if you are initializing BertForSequenceClassification
from the checkpoint of a model that you expect to be exactly identical (init
ializing a BertForSequenceClassification model from a BertForSequenceClassif
ication model).
Some weights of BertForSequenceClassification were not initialized from the
model checkpoint at bert-base-uncased and are newly initialized: ['classifie
r.weight', 'classifier.bias']
You should probably TRAIN this model on a down-stream task to be able to use
it for predictions and inference.
C:\Users\ritvi\miniconda3\envs\EMP_env\lib\site-packages\transformers\optimi
zation.py:306: FutureWarning: This implementation of AdamW is deprecated and
will be removed in a future version. Use the PyTorch implementation torch.op
tim.AdamW instead, or set `no_deprecation_warning=True` to disable this warn
ing
  warnings.warn(
***** Running training *****
  Num examples = 4060
  Num Epochs = 1
  Instantaneous batch size per device = 4
  Total train batch size (w. parallel, distributed & accumulation) = 4
  Gradient Accumulation steps = 1
  Total optimization steps = 1015
  Number of trainable parameters = 109484547
```

[1015/1015 04:23, Epoch 1/1]

| Step | Training Loss |
|---|---|
| 500 | 0.398000 |
| 1000 | 0.189000 |

```
Saving model checkpoint to ./results\checkpoint-500
Configuration saved in ./results\checkpoint-500\config.json
Model weights saved in ./results\checkpoint-500\pytorch_model.bin
Saving model checkpoint to ./results\checkpoint-1000
Configuration saved in ./results\checkpoint-1000\config.json
Model weights saved in ./results\checkpoint-1000\pytorch_model.bin


Training completed. Do not forget to share your model on huggingface.co/mode
ls =)


Configuration saved in ./fine_tuned_bert\config.json
Model weights saved in ./fine_tuned_bert\pytorch_model.bin
tokenizer config file saved in ./fine_tuned_bert\tokenizer_config.json
Special tokens file saved in ./fine_tuned_bert\special_tokens_map.json
```

Out[13]:
```
('./fine_tuned_bert\\tokenizer_config.json',
 './fine_tuned_bert\\special_tokens_map.json',
 './fine_tuned_bert\\vocab.txt',
 './fine_tuned_bert\\added_tokens.json')
```

## Fine tuning the second model

In [14]:
```python
from transformers import BertTokenizer, BertModel, AdamW
import torch
import torch.nn as nn
from tqdm import tqdm
from torch.utils.data import DataLoader, Subset

# Load models and tokenizers
model_1 = BertModel.from_pretrained('./fine_tuned_bert')
tokenizer_1 = BertTokenizer.from_pretrained('./fine_tuned_bert')

model_2 = BertModel.from_pretrained('bert-base-uncased')
tokenizer_2 = BertTokenizer.from_pretrained('bert-base-uncased')

# Simple classifier head
class SimpleClassifier(nn.Module):
    def __init__(self, input_dim, num_classes):
        super(SimpleClassifier, self).__init__()
        self.fc = nn.Linear(input_dim, num_classes)

    def forward(self, embeddings):
        return self.fc(embeddings)

# Instantiate classifier
classifier_model = SimpleClassifier(input_dim=768, num_classes=3)
```

```
# Function to get embeddings from the first transformer
def get_first_transformer_embeddings(sentence):
    if isinstance(sentence, torch.Tensor):
        encoding = {'input_ids': sentence}
    else:
        encoding = tokenizer_1.encode_plus(
            sentence,
            add_special_tokens=True,
            max_length=128,
            padding='max_length',
            truncation=True,
            return_attention_mask=True,
            return_tensors='pt',
        )

    with torch.no_grad():
        outputs = model_1(**encoding)

    embeddings = outputs.last_hidden_state
    return embeddings

# Fine-tune second transformer
def fine_tune_second_transformer(first_transformer_output, attention_mask):
    outputs = model_2(inputs_embeds=first_transformer_output, attention_mask
    last_hidden_state = outputs.last_hidden_state
    return last_hidden_state[:, 0, :]  # [CLS] token

# Optimizer with gradient clipping and lower learning rate
optimizer = AdamW(model_2.parameters(), lr=2e-5)
scheduler = torch.optim.lr_scheduler.StepLR(optimizer, step_size=1, gamma=0.

# Assuming `train_dataloader` is defined
# To get 10% of the dataset, we can use the following approach:
# Get a subset of 10% of the indices from the train dataset
dataset_size = len(train_dataloader.dataset)
subset_size = int(dataset_size * 0.1)  # 10% of the dataset size
subset_indices = torch.randperm(dataset_size).tolist()[:subset_size]

# Create a new DataLoader using the subset
train_subset = Subset(train_dataloader.dataset, subset_indices)
train_subset_dataloader = DataLoader(train_subset, batch_size=train_dataloac

# Training loop with mixed precision
scaler = torch.cuda.amp.GradScaler()  # For mixed precision training

for epoch in range(1):
    model_2.train()
    classifier_model.train()
    for batch in tqdm(train_subset_dataloader, desc="Training Progress", uni
        input_ids = batch['input_ids']
        attention_mask = batch['attention_mask']

        # Get embeddings from the first transformer (fine-tuned BERT)
        first_transformer_output = get_first_transformer_embeddings(input_ic
        # Fine-tune the second transformer
        second_transformer_output = fine_tune_second_transformer(first_trans

        # Pass the output to the classifier
        logits = classifier_model(second_transformer_output)

        # Compute loss
        loss = nn.MSELoss()(logits, batch['labels'])

        # Backpropagation with mixed precision
        optimizer.zero_grad()
        scaler.scale(loss).backward()
        scaler.step(optimizer)
        scaler.update()

    scheduler.step()
```

```
loading configuration file ./fine_tuned_bert\config.json
Model config BertConfig {
  "_name_or_path": "bert-base-uncased",
  "architectures": [
    "BertForSequenceClassification"
  ],
  "attention_probs_dropout_prob": 0.1,
  "classifier_dropout": null,
  "gradient_checkpointing": false,
  "hidden_act": "gelu",
  "hidden_dropout_prob": 0.1,
  "hidden_size": 768,
  "id2label": {
    "0": "LABEL_0",
    "1": "LABEL_1",
    "2": "LABEL_2"
  },
  "initializer_range": 0.02,
  "intermediate_size": 3072,
  "label2id": {
    "LABEL_0": 0,
    "LABEL_1": 1,
    "LABEL_2": 2
  },
  "layer_norm_eps": 1e-12,
  "max_position_embeddings": 512,
  "model_type": "bert",
  "num_attention_heads": 12,
  "num_hidden_layers": 12,
  "pad_token_id": 0,
  "position_embedding_type": "absolute",
  "problem_type": "multi_label_classification",
  "torch_dtype": "float32",
  "transformers_version": "4.26.1",
  "type_vocab_size": 2,
  "use_cache": true,
  "vocab_size": 30522
}

loading weights file ./fine_tuned_bert\pytorch_model.bin
C:\Users\ritvi\miniconda3\envs\EMP_env\lib\site-packages\transformers\modeli
ng_utils.py:415: FutureWarning: You are using `torch.load` with `weights_onl
y=False` (the current default value), which uses the default pickle module i
mplicitly. It is possible to construct malicious pickle data which will exec
ute arbitrary code during unpickling (See https://github.com/pytorch/pytorc
h/blob/main/SECURITY.md#untrusted-models for more details). In a future rele
ase, the default value for `weights_only` will be flipped to `True`. This li
mits the functions that could be executed during unpickling. Arbitrary objec
ts will no longer be allowed to be loaded via this mode unless they are expl
icitly allowlisted by the user via `torch.serialization.add_safe_globals`. W
e recommend you start setting `weights_only=True` for any use case where you
don't have full control of the loaded file. Please open an issue on GitHub f
or any issues related to this experimental feature.
  return torch.load(checkpoint_file, map_location="cpu")
Some weights of the model checkpoint at ./fine_tuned_bert were not used when
initializing BertModel: ['classifier.weight', 'classifier.bias']
```

```
- This IS expected if you are initializing BertModel from the checkpoint of
a model trained on another task or with another architecture (e.g. initializ
ing a BertForSequenceClassification model from a BertForPreTraining model).
- This IS NOT expected if you are initializing BertModel from the checkpoint
of a model that you expect to be exactly identical (initializing a BertForSe
quenceClassification model from a BertForSequenceClassification model).
All the weights of BertModel were initialized from the model checkpoint at
./fine_tuned_bert.
If your task is similar to the task the model of the checkpoint was trained
on, you can already use BertModel for predictions without further training.
loading file vocab.txt
loading file added_tokens.json
loading file special_tokens_map.json
loading file tokenizer_config.json
C:\Users\ritvi\miniconda3\envs\EMP_env\lib\site-packages\huggingface_hub\fil
e_download.py:1150: FutureWarning: `resume_download` is deprecated and will
be removed in version 1.0.0. Downloads always resume when possible. If you w
ant to force a new download, use `force_download=True`.
  warnings.warn(
loading configuration file config.json from cache at C:\Users\ritvi/.cache\h
uggingface\hub\models--bert-base-uncased\snapshots\86b5e0934494bd15c9632b12f
734a8a67f723594\config.json
Model config BertConfig {
  "architectures": [
    "BertForMaskedLM"
  ],
  "attention_probs_dropout_prob": 0.1,
  "classifier_dropout": null,
  "gradient_checkpointing": false,
  "hidden_act": "gelu",
  "hidden_dropout_prob": 0.1,
  "hidden_size": 768,
  "initializer_range": 0.02,
  "intermediate_size": 3072,
  "layer_norm_eps": 1e-12,
  "max_position_embeddings": 512,
  "model_type": "bert",
  "num_attention_heads": 12,
  "num_hidden_layers": 12,
  "pad_token_id": 0,
  "position_embedding_type": "absolute",
  "transformers_version": "4.26.1",
  "type_vocab_size": 2,
  "use_cache": true,
  "vocab_size": 30522
}

loading weights file model.safetensors from cache at C:\Users\ritvi/.cache\h
uggingface\hub\models--bert-base-uncased\snapshots\86b5e0934494bd15c9632b12f
734a8a67f723594\model.safetensors
Some weights of the model checkpoint at bert-base-uncased were not used when
initializing BertModel: ['cls.predictions.transform.LayerNorm.bias', 'cls.pr
edictions.bias', 'cls.predictions.transform.dense.weight', 'cls.seq_relation
ship.weight', 'cls.predictions.transform.dense.bias', 'cls.predictions.trans
form.LayerNorm.weight', 'cls.seq_relationship.bias']
- This IS expected if you are initializing BertModel from the checkpoint of
```

```
a model trained on another task or with another architecture (e.g. initializ
ing a BertForSequenceClassification model from a BertForPreTraining model).
- This IS NOT expected if you are initializing BertModel from the checkpoint
of a model that you expect to be exactly identical (initializing a BertForSe
quenceClassification model from a BertForSequenceClassification model).
All the weights of BertModel were initialized from the model checkpoint at b
ert-base-uncased.
If your task is similar to the task the model of the checkpoint was trained
on, you can already use BertModel for predictions without further training.
loading file vocab.txt from cache at C:\Users\ritvi/.cache\huggingface\hub\m
odels--bert-base-uncased\snapshots\86b5e0934494bd15c9632b12f734a8a67f723594
\vocab.txt
loading file added_tokens.json from cache at None
loading file special_tokens_map.json from cache at None
loading file tokenizer_config.json from cache at C:\Users\ritvi/.cache\huggi
ngface\hub\models--bert-base-uncased\snapshots\86b5e0934494bd15c9632b12f734a
8a67f723594\tokenizer_config.json
loading configuration file config.json from cache at C:\Users\ritvi/.cache\h
uggingface\hub\models--bert-base-uncased\snapshots\86b5e0934494bd15c9632b12f
734a8a67f723594\config.json
Model config BertConfig {
  "_name_or_path": "bert-base-uncased",
  "architectures": [
    "BertForMaskedLM"
  ],
  "attention_probs_dropout_prob": 0.1,
  "classifier_dropout": null,
  "gradient_checkpointing": false,
  "hidden_act": "gelu",
  "hidden_dropout_prob": 0.1,
  "hidden_size": 768,
  "initializer_range": 0.02,
  "intermediate_size": 3072,
  "layer_norm_eps": 1e-12,
  "max_position_embeddings": 512,
  "model_type": "bert",
  "num_attention_heads": 12,
  "num_hidden_layers": 12,
  "pad_token_id": 0,
  "position_embedding_type": "absolute",
  "transformers_version": "4.26.1",
  "type_vocab_size": 2,
  "use_cache": true,
  "vocab_size": 30522
}

C:\Users\ritvi\miniconda3\envs\EMP_env\lib\site-packages\transformers\optimi
zation.py:306: FutureWarning: This implementation of AdamW is deprecated and
will be removed in a future version. Use the PyTorch implementation torch.op
tim.AdamW instead, or set `no_deprecation_warning=True` to disable this warn
ing
  warnings.warn(
C:\Users\ritvi\AppData\Local\Temp\ipykernel_31648\570885447.py:69: FutureWar
ning: `torch.cuda.amp.GradScaler(args...)` is deprecated. Please use `torch.
amp.GradScaler('cuda', args...)` instead.
  scaler = torch.cuda.amp.GradScaler()  # For mixed precision training
```

```
Training Progress: 100%|██████████████████████████████████████████████████████
██████████████████████████████████████████████████████████████████████████████
██████████████████████| 102/102 [04:41<00:00,  2.76s/batch]
```

In [15]:
```python
import numpy as np
import torch
from tqdm import tqdm

# Function to extract features (embeddings) from both models
def extract_embeddings(data_loader, model_1, model_2, tokenizer_1, tokenizer
    all_embeddings = []
    all_labels = []

    with torch.no_grad():
        for batch in tqdm(data_loader, desc="Extracting Features", unit="bat
            input_ids = batch['input_ids']
            attention_mask = batch['attention_mask']
            labels = batch['labels']

            # Get embeddings from model_1 (fine-tuned BERT)
            first_transformer_output = get_first_transformer_embeddings(inpu

            # Extract [CLS] token from first model (batch_size, embedding_si
            first_model_embeddings = first_transformer_output[:, 0, :].cpu()

            # Fine-tune second transformer (model_2 - BERT)
            second_transformer_output = fine_tune_second_transformer(first_t
            second_model_embeddings = second_transformer_output.cpu().numpy(

            # Combine both embeddings
            combined_embeddings = np.concatenate((first_model_embeddings, se

            all_embeddings.append(combined_embeddings)
            all_labels.append(labels.cpu().numpy())

    all_embeddings = np.concatenate(all_embeddings, axis=0)
    all_labels = np.concatenate(all_labels, axis=0)

    return all_embeddings, all_labels
```

In [24]:
```python
import torch
import numpy as np
from sklearn.metrics import classification_report, accuracy_score
from tqdm import tqdm

# ------------------------ #
# Extract embeddings from Model 1
# ------------------------ #
def extract_embeddings_from_model_1(data_loader, model_1, device='cuda'):
    model_1.eval()
    all_embeddings = []
    all_labels = []

    with torch.no_grad():
        for batch in tqdm(data_loader, desc="Extracting Embeddings from Mode
            input_ids = batch['input_ids'].to(device)
            attention_mask = batch['attention_mask'].to(device)
            labels = batch['labels'].to(device)

            # Forward pass
            outputs = model_1(input_ids=input_ids, attention_mask=attention_
            cls_embeddings = outputs.last_hidden_state[:, 0, :]  # CLS toker

            all_embeddings.append(cls_embeddings.cpu())
            all_labels.append(labels.cpu())

    all_embeddings = torch.cat(all_embeddings).numpy()
    all_labels = torch.cat(all_labels).numpy()

    return all_embeddings, all_labels

# ------------------------ #
# Classify using Model 2 + Classifier Head
# ------------------------ #
def classify_using_model_2_with_embeddings(embeddings, model_2, classifier_m
    model_2.eval()
    classifier_model.eval()

    embeddings_tensor = torch.tensor(embeddings, dtype=torch.float32).to(dev
    embeddings_tensor = embeddings_tensor.unsqueeze(1)  # Add dummy seq_leng

    attention_mask = torch.ones(embeddings_tensor.size()[:-1], dtype=torch.l

    with torch.no_grad():
        outputs = model_2(inputs_embeds=embeddings_tensor, attention_mask=at
        pooled_output = outputs.last_hidden_state[:, 0, :]  # CLS token
        logits = classifier_model(pooled_output)
        predictions = torch.argmax(logits, dim=-1)

    return predictions.cpu().numpy()

# ------------------------ #
# Main Evaluation Pipeline
# ------------------------ #
def evaluate(test_dataloader, model_1, model_2, classifier_model, device='cu
    model_1.eval()
    model_2.eval()
    classifier_model.eval()


    # 3. Metrics
    accuracy = accuracy_score(true_labels, predictions)
    report = classification_report(true_labels, predictions)

    print(f'Accuracy: {accuracy * 100:.2f}%')
    print(report)
```

In [32]:
```
# 1. Extract embeddings
test_embeddings, true_labels = extract_embeddings_from_model_1(test_dataloac

# 👉 Convert true_labels if needed
if len(true_labels.shape) > 1 and true_labels.shape[1] > 1:
    true_labels = np.argmax(true_labels, axis=1)

# 2. Classify
predictions = classify_using_model_2_with_embeddings(test_embeddings, model_
```

```
Extracting Embeddings from Model 1: 100%|███████████████████████████████████
████████████████████████████████████████████████████████████████████████████
█████████████████████| 254/254 [00:21<00:00, 11.65batch/s]
```

In [33]:
```
# Example usage
device = 'cuda' if torch.cuda.is_available() else 'cpu'
model_1.to(device)
model_2.to(device)
classifier_model.to(device)

evaluate(test_dataloader, model_1, model_2, classifier_model, device)
```

```
Extracting Embeddings from Model 1: 100%|███████████████████████████████████
████████████████████████████████████████████████████████████████████████████
█████████████████████| 254/254 [00:21<00:00, 11.87batch/s]
Accuracy: 79.21%
              precision    recall  f1-score   support

           0       0.95      0.44      0.61       344
           1       0.83      0.98      0.90       499
           2       0.62      0.93      0.74       172

    accuracy                           0.79      1015
   macro avg       0.80      0.79      0.75      1015
weighted avg       0.83      0.79      0.77      1015

[1 1 1 ... 1 0 1]
```

In [34]:
```
import torch
from IPython.display import display, HTML
import re

# Define colors for each label
label_colors = {
    0: '#ADD8E6',  # Light Blue, task response
    1: '#90EE90',  # Light Green, cohesion
    2: '#FFB6C1',  # Light Pink, lexical resources
}

# Function to split essays into sentences (simple version)
def split_into_sentences(text):
    sentences = re.split(r'(?<=[.!?]) +', text)
    return [s for s in sentences if s]

# Function to predict label for one sentence
def predict_label(sentence, model_1, model_2, classifier_model, tokenizer_1,
    model_1.eval()
    model_2.eval()
    classifier_model.eval()

    encoding = tokenizer_1.encode_plus(
        sentence,
        add_special_tokens=True,
        max_length=128,
        padding='max_length',
        truncation=True,
        return_attention_mask=True,
        return_tensors='pt',
    ).to(device)

    with torch.no_grad():
        outputs1 = model_1(**encoding)
        first_transformer_output = outputs1.last_hidden_state  # (1, seq_ler

        # Pass embeddings into model_2
        outputs2 = model_2(inputs_embeds=first_transformer_output, attentior
        second_transformer_output = outputs2.last_hidden_state[:, 0, :]  # C

        logits = classifier_model(second_transformer_output)
        predicted_label = torch.argmax(logits, dim=-1).item()

    return predicted_label

# Function to highlight sentences according to their label
def highlight_essay(essay_text, model_1, model_2, classifier_model, tokenize
    sentences = split_into_sentences(essay_text)
    highlighted_sentences = []

    for sentence in sentences:
        label = predict_label(sentence, model_1, model_2, classifier_model,
        color = label_colors.get(label, '#FFFFFF')  # Default to white if la
        highlighted_sentence = f'<span style="background-color:{color}; padc
        highlighted_sentences.append(highlighted_sentence)

    highlighted_essay = ' '.join(highlighted_sentences)
    return highlighted_essay

# 🚀 Now go through 10 essays and highlight them
def display_highlighted_essays(Test_Essays, model_1, model_2, classifier_moc
    for idx, essay in enumerate(Test_Essays["Essay"][:10]):  # Take first 1C
        highlighted = highlight_essay(essay, model_1, model_2, classifier_mc
        display(HTML(f"<h3>Essay {idx+1}</h3><p>{highlighted}</p>"))
```

In [35]:
```
display_highlighted_essays(Test_Essays, model_1, model_2, classifier_model,
```

## Essay 1

The graphic illustrates the nutritional consistency of two different types of dinners, medium baked potato and macaroni. Both dinners are more constituents of carbohydrates, coloured in green,with respectively the 35% in the baked potato and 52% in the macaroni. In the first dinner, the protein, in purple, occupie the second place in the pie with the 25%, followed by the glucose, in yellow, that represents just the 15% of the pie . In the macaroni, the glucose and the protein occupies exactly the same amount of the pie with the 11% while the second place is occupied by the saturated fat, in dark blue, with the 21%, that, in contrast, it's only the 10% in the medium baked potato. The last 15% of the pie of the first dinner is completed by other nutrients, in light blue, while, in the second dinner, the other nutrients occupie just the 5% of the pie graph.

## Essay 2

Life is full of competition. Soon after the release from competitive university exams, the hard fighting for higher positions in a company will be waiting for you. While the competitiveness has been bringing us a convenient society, no small number of employments have been tired of never-ending competitions. No doubt about a positive effect on societies, but when we start to argue about the effect on individuals, it becomes more controversial. This essay will argue the drawbacks of competitiveness for individuals, especially from the perspective of Japanese modern society and typical Japanese office workers. I believe that cooperativeness and uniqueness are sometimes more important attitude than competitiveness. Firstly, competitiveness causes less diversity. If there is a competition, there must be the same evaluation metrics to differentiate winners and losers. The evaluation procedure needs to limit the metrics into specific aspects of the human being. Therefore, it is extremely difficult to prove that the metrics are proper for all of the possible cases and effective for the HR evaluation. For example, Japanese companies are likely to judge quantitatively based on the numbers related to performance. However, how can we evaluate the cooperativity or the non-quantitative performance like in creative jobs? Most Japanese companies adopted the HR evaluation methods based on average working hours per day and number of reports monthly. That’s why the Japanese tend to overwork to fulfill this requirement. In addition, as a result of placing more importance in quantity of reports, rather than quality, Japanese employees are prone to choose easy and short-term tasks or outsourcing instead of building by their own methods. This tendency causes fewer complicated and long-term tasks. Secondly, competitiveness causes less cooperativeness. Fewer people volunteer to play a disadvantageous role for smooth teamwork. Moreover, some people play an evil role by blaming or cheating the competitors in order to be superior to them. As a consequence, overall efficiency will decline and some people will be overestimated. Even if there are no problems temporarily, the long-lasting large companies will eventually become corrupt, and those who have the most political power, rather than real power, will remain at the top. Consequently, competitiveness is the basis of capitalism which brings us a brighter and more convenient society, however, biased evaluation and less cooperative working environment possibly drive people into nervous breakdowns. Karoshi, overwork death derived from too much stress and less sleeping, is one of the most serious social problems in Japan. We may need to rethink about the competitiveness.

## Essay 3

Sugar-based drinks has become a popular beverage amongst the people especially in today's generation, the millenials. This essay will outline possible reasons on why people are consuming such drinks and also some ways for them to consume less. To start, sugar is the main component of this beverage. Study shows that consuming this can boost a person's energy levels. Hence athelete ae addicted to this refreshments such as energy drinks. Furthermore, this can increase mental alertness especially to working individuals. For instance, not all workers has the capability to maintain their energy at work, sometimes they tend to get sleepy leading them to buy sugary drinks like milk teas. Not only that, advertisement can also be a factor. For example, young people gets curious whenever new sugared drinks is advertised especially if this is shown by their favorite icon. Lastly, consumer's lack of knowledge and misinformation regarding their product as evident by the coca cola lawsuit. Although it may be difficut for people to stay away from this. There are ways that can help them drink less. One solution is for government to impose a price hike. Meaning high price can discourage them to buy this drinks. Second, just like alcohol, buying sugar-based drinks at malls or grocery store should have time limit. For instance, whoever will buy and sell beyond the allotted time should pay a fine. Also, Educating the public about the health issue of drinking too much sugar-based drinks should be prioritize. Finally, government should encourage company such as pepsiCo, and the Coca-cola company to be transparent of their nutritional facts. In conclusion, sugar-based drinks has become popular due to advertisements and society's thinking that it can replenish their energy levels. In contrast, goverment should take the first step educate and encourage individuals to mimimize drinking this.

## Essay 4

In the Third World, children are usually sent to factories for laborious work. Many people believe that it is merely exploitation, while others think it is a good opportunity for them to life experience. In any case, children have their right to live and study in peaceful conditions. Therefore, using them as workforce is considered an unacceptable action.First of all, children are not workers. They have just learned about the vast world and do not have any experience or concept of working. Since these innocent children are naive and , they can be easily cheated and exploited. There are many examples of this in poor nations. Because using children is cheap and to control, many enterprises hire them and don't pay them much. Although the government in these countries has tried its best, this kind of taking advantage of children cannot be eliminated.Moreover, children do not need such thing as 'valuable work experience' that is supposedly 'important for learning and taking responsibility'. The brief responsibility of children is learning. They are not old enough to understand what working experiences are. Nevertheless, they can help parents do chores or housework. This will be a much better way for them to become more responsible for . In addition, childhood is one of the most remarkable memories and must not be taken away by forcing them to work.In conclusion, since all children are the great concern of parent and society, they should be allowed to enjoy life and rather than to work. Hence, one must ponder what view is actually appropriate for the sake of the children.

## Essay 5

The line graph illustrates the proportion of male and female students graduated from Canadian universities from 1992 to 2007. Overall, there were 40. 000 more female graduates than male graduates in 1992 and this trend remained constant for the given time period. In the starting year, in 1992 the rate of the female gender representatives was slight below 100.000, after which line grew up to approximately 105.000 in 1995 and again dropped down to 100.000 between 1997 and 1998. This graph steadily increased in 1998 and peaked at over 148. 000 in 2007. The male line graph showed a fluctuation. It's started from the rate 70.000 in 1992 and slowly went up to approximately 78. 000 in 1995 and then went down to above 72. 000 in 1998. In 1998 line dropped, and then showed a slight increase and again a decrease to the rate of about 73.000. In 2001 line indicator grew dramatically and reached at the point of 95. 000 in 2007.

## Essay 6

Despite the fact that physical punishment of pupils at schools is decreasing, it is considered by many that it improves discipline. This essay completely disagrees with this view because such measures are applied to those who are unable to counterattack and it maybe the easiest way but also the most inhumane. Firstly, physical assault is usually applied by teachers against children who are not able to stop the violence. In other words, it is not highly probable that an adolescent will try to hurt a teenager who may stop the hitting or even push it back. Thus, kids are at higher risk because the only thing they are able to do in such cases is just try to run away. Moreover, children should understand that they earn what they deserve meaning that the level of repercussions should correspond to the seriousness of so called crime. For example, Sweden schools have publicly available the set of rules and non-physical measures that will be taken against the violators, that confirms that other methods are also effective to keep appropriate kids' behaviour. Secondly, it is much more easier for many to maintain a discipline using the fear of corporal punishment. This means that it seems unnecessary and time-waste for grown-ups to try to apply other methods while a hectic kid may be calmed down just in few seconds using the fear of pain. Pupils may not be involved in a lesson and interrupt a teacher because they are not interested in the subject that may be caused by the fact that the teacher was not able to select appropriate technics to convey knowledge. For example, a research conducted by a US institution revealed that 90% of corporal punishments were applied to pupils who did not l listened to their teachers during a lesson. In conclusion, physical violence should be discontinued in schools because mostly weak kids are in danger and they are subject for such punishments because adolescents chose the simplest way of violence and fear to better the behaviour.

## Essay 7

Nowadays having a job is a common type of lifestyle for all individuals. As a matter of fact, jobs are very important because is the only way that people have to earn money. However, in many countries, the distribution of salary is not equal and a small number of people earn higher salaries than others. Some individuals think that this is good for the country because goverment encourage only categories that are more in touch with money, such as employees who work in banks, offices and who has a business. In other words, the economy of the country can increase in a significant way since goverments incentivate these categories by giving them extremely high salaries. On the other hand, there are those who argue that goverments should not allow salaries above a certain level because there are a lot of employees, such as workers and teachers that have a very small salary. In fact, they are the ones who works for the wealth and the future of the society and it is unfair that they are discourage. Moreover, I also believe that these kind of jobs are more stressful and challenging than others and they work as good as categories that have higher salaries. In conclusion, I hold the view that everybody should have an equal distribution of salary and it is unfair that goverments incentivate only a certain category of workers. All individuals should be paid according to the commitment they give and we should not think only about the economy of the country.

## Essay 8

In many countries the proportion of older people is steadily increasing. What problems will this cause for individuals and society? Suggest some measures that could be taken to reduce the impact of ageing populations.It is true that nowadays people in industrialised nations can expect to live longer than ever before. Although there will undoubtedly be some negative consequences to this trend, societies can take steps to mitigate these potential problems.As people live longer and the populations of developed countries grow older, several related problems can be anticipated. The main issue is that there will obviously be more people of retirement age who will be eligible to receive a pension. The proportion of younger, working adults will be smaller, and governments will therefore receive less money in taxes in relation to the size of the population. In other words, an ageing population will mean a greater tax burden working adults. Further pressures will include a rise in the demand for healthcare, and the adults will increasingly have to look after their elderly relatives.There are several actions that governments could take to solve the problems described above. Firstly, a simple solution would be to increase the retirement age for working adults, perhaps from 65 to 70. Nowadays, people of this age tend to be healthy enough to continue a productive working life. A second measure would be for governments to encourage immigration in order to increase the number of working adults who pay taxes. Finally, money from national budgets will need to be taken from other areas and spent vital healthcare, accommodation and transport facilities for the rising numbers of senior citizens.In conclusion, various measures can be taken to tackle the problems that are certain to arise as the populations of countries grow older.This essay covers the task requirements, however some problems of aging population get only a brief mention. The linking of sentences in the second body paragraph is somewhat ‘mechanical’ and could have been improved. Information sequencing and organisation in paragraphs are done well. The range of vocabulary and the fluency of its use, as well as lack of errors are impressive. Overall, this is a good example of how to get Band 8 without writing a very long essay.

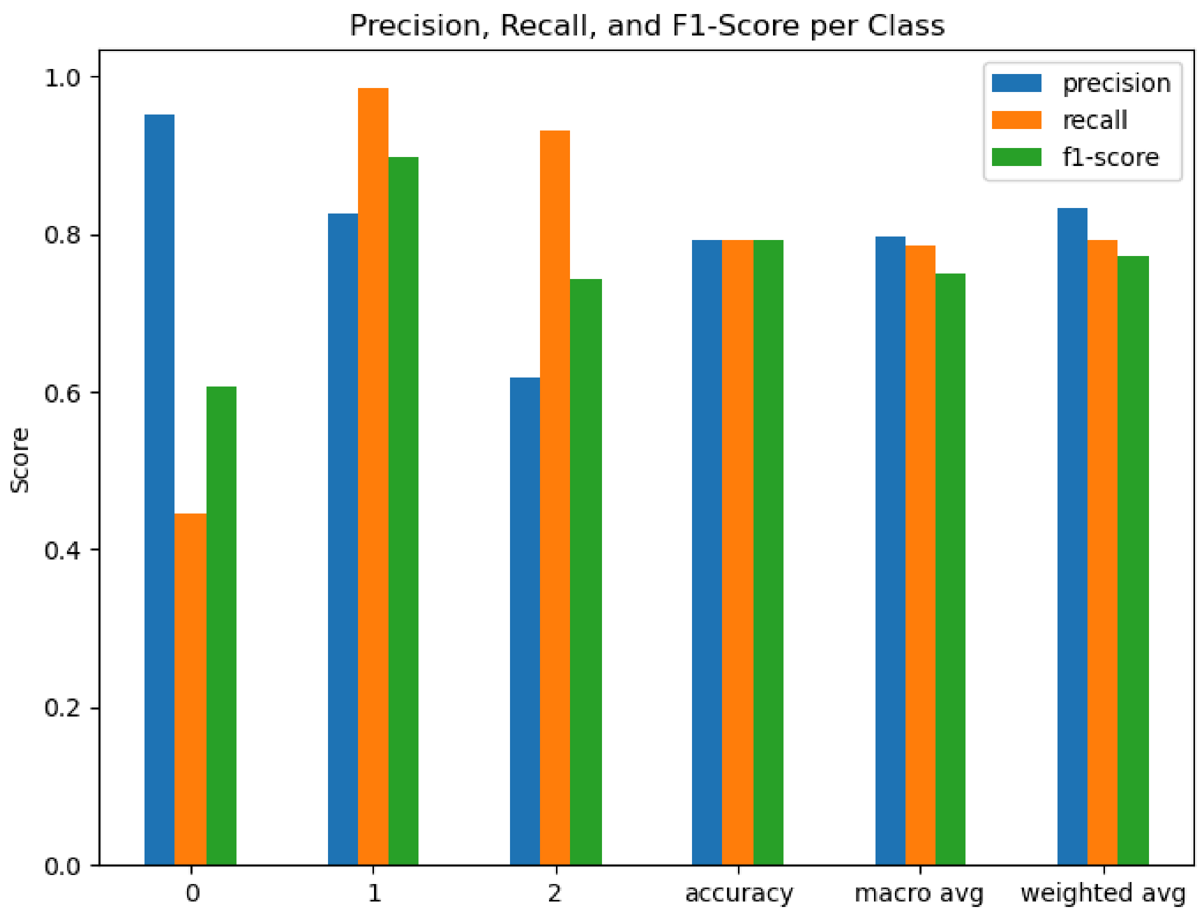
## Essay 9

Nowadays, more and more criminal activities and violent events are showed on television and in newspapers. Some people consider that we should restrict those kinds of information because of the adverse outcomes. However, in my opinion, those events shown in the media should not be restricted for the following reasons. Firstly, reporting of criminal activities and other violent news can reduce crimes. If those criminal and violent information can be showed to the public, people can clearly understand the whole process and detailed scenes of crimes, provoking them more easily think critically serious consquences. Abusing durgs, for example, people will be afarid of the severe effects on both social and personal areas when they vivily see real outcomes it may bring, thus prevent them from committing criminal acts. Hence, there will be fewer criminal activities or violent events occurring on our society. That is why we should not restrict criminal activities being shown in the media. In addition, restricting the criminal activities may lead to a high rate of crimes. If people cannot easily obtain those information, they may consider that no one care about it and even take part in some criminal activities without thinking. Because they cannot image detailed impacts on social and personal aspects when they think wether can do it or not. Finally, to a large extent, they may commit cirmes and cause some severe consquences, which may bring nagative effects on society. In conclusion, I agree that we should show criminal activities and violent events in the media , which can largely reduce crimes. Moreover, we should also think much more about that restricting this kind of information may lead to a high rate of crimes.

## Essay 10

The bar chart illustrates the figure for uses of a community website during the first and the second year after it is created. Overall, it can be seen that the number of visits to the website in each month in the second year is more than the figure in the first year noticeably, except for August. Additionally, the website gains popularity differently, depending on the periods of the year. Looking at the detail, at the beginning of each year the record commences at below 5,000 uses. This figure then plateaus to 10,000 uses and around 17,500 uses in December of the first and the second year respectively. After that, from December to February the website popularity declines to almost zero for the first year which is the lowest record and 10,000 uses for the second year. In the following month, even though the number in the first year increases slightly, the number in the second year remains unchanged. Nevertheless, between March and June the figure starts to rise for a second time to approximately 12,500 uses in the first year and more than 20,000 uses in the second year. Then, the number of both years stays at this level until July. On August, the popularity in the second year drops to below 15,000 uses while the number in the first year continues increasing, exceeding the number in the second year.

In [37]:
```
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.metrics import confusion_matrix, classification_report
import pandas as pd
import numpy as np

# Assuming `true_labels` and `predictions` are already available from your e
# 1. Confusion Matrix

def plot_confusion_matrix(true_labels, predictions, labels):
    cm = confusion_matrix(true_labels, predictions, labels=labels)
    plt.figure(figsize=(8, 6))
    sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=labels, y
    plt.title('Confusion Matrix')
    plt.ylabel('True Label')
    plt.xlabel('Predicted Label')
    plt.show()

# 2. Precision, Recall, F1 Score Plot
def plot_metrics(metrics_report):
    metrics_df = pd.DataFrame(metrics_report).T
    metrics_df = metrics_df[['precision', 'recall', 'f1-score']]
    metrics_df.plot(kind='bar', figsize=(8, 6))
    plt.title('Precision, Recall, and F1-Score per Class')
    plt.ylabel('Score')
    plt.xticks(rotation=0)
    plt.show()

# Assuming `classification_report` (from sklearn) is used to get precision,
metrics_report = classification_report(true_labels, predictions, output_dict
plot_metrics(metrics_report)
```



```
In [ ]:
```