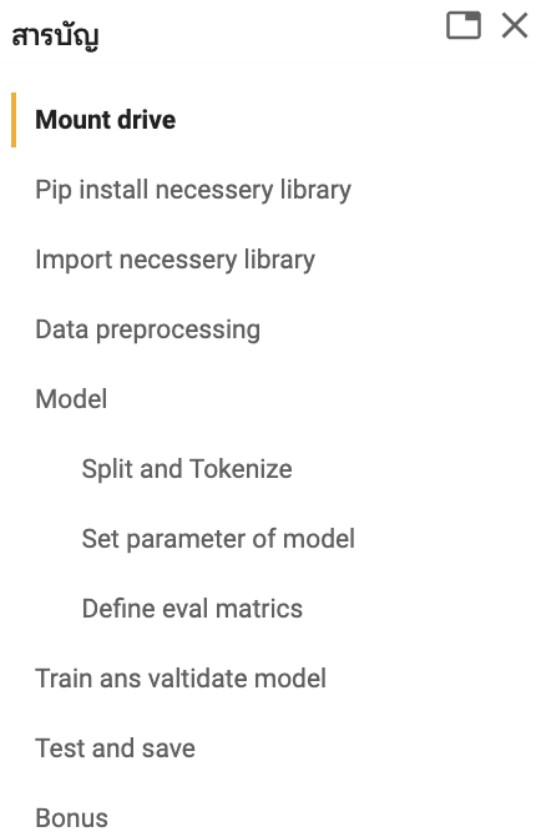
Report Midterm Data 2110531 Data Science and Data Engineering Tools

Chapter 1: Introduction

ปัจจุบันในยุคของเทคโนโลยีข้อมูลส่วนใหญ่มักเก็บอยู่ในรูปแบบออนไลน์เพื่อง่ายต่อการเข้าถึงและจัดเก็บ อย่างไรก็ตามเมื่อข้อมูลมีมากเกินกว่าดึงมาใช้งานได้อย่างสะดวก การจัดกลุ่มข้อมูลให้เป็นระเบียบก็เป็นสิ่งสำคัญ โดยในโจทย์ปัญหานี้ที่เป็นการจัดกลุ่มงานวิจัยในสาขาของวิศวกรรมศาสตร์ รวมทั้งหมด 18 สาขาประกอบด้วย วิศวกรรมโยธา วิศวกรรมสิ่งแวดล้อม วิศวกรรมชีวการแพทย์ วิศวกรรมน้ำมัน วิศวกรรมโลหะวิศวกรรมเครื่องกล วิศวกรรมไฟฟ้า วิศวกรรมคอมพิวเตอร์ วิศวกรรมออพติก วิศวกรรมนาโน วิศวกรรมเคมี วิศวกรรมวัสดุ วิศวกรรมเกษตร วิศวกรรมการศึกษา วิศวกรรมอุตสาหกรรม วิศวกรรมความปลอดภัย"คณิตศาสตร์และสถิติ" ในประเด็นของวิศวกรรม และวิทยาศาสตร์วัสดุ 454 งานวิจัยสำหรับการให้เรียนรู้แบบจำลอง ซึ่งแต่ละงานวิจัยจะประกอบด้วยหัวข้อเรื่องและบทคัดย่อที่ถูกจัดกลุ่มอยู่ในสาขาวิศวกรรมอย่างน้อย 1 สาขาจึงทำให้โจทย์ปัญหานี้เป็นโจทย์ปัญหาแบบ multi-label และ multi-class text classification โดยการจัดกลุ่มงานวิจัยให้ถูกต้องต้องมีการเตรียมข้อมูลที่เหมาะแก่ลักษณะของข้อมูลและนำไปทำนายด้วยแบบจำลองที่เหมาะสมควบคู่ไปกับการปรับพารามิเตอร์ที่เหมาะสมจากการแบ่งส่วนให้การเรียนรู้เพื่อการประเมินก่อนไปตรวจสอบกับข้อมูลส่วนทดสอบ



รูปที่ 1 แสดงการสารบัญของการทำการทดลอง

Chapter 2: Data preparation

1. เนื่องจากข้อมูลที่มีในส่วนของข้อมูลขาเข้าหรือประโยคที่จะนำเข้าแบบจำลองเพื่อการเรียนรู้ใช้ได้ทั้งสอง Column จึงรวม Column Title และ Abstract เป็น Column ชื่อ Text และทิ้งแถวที่ซ้ำ

```
df = data_train.copy()
cols = df.columns.tolist()
df['Text'] = df['Title']+". "+df['Abstract']
df['labels'] = df['Classes']
df = df.drop(columns=['Title', 'Abstract','Classes'])
df.reset_index(drop=True, inplace=True)
df
```

	Text	labels
0	Activated carbon derived from bacterial cellul...	[CHE, MATENG]
1	The algorithm of static hand gesture recogniti...	[CPE]
2	Alternative Redundant Residue Number System Co...	[EE]
3	Comparative study of wax inhibitor performance...	[PE, ME, CHE]
4	Undrained lower bound solutions for end bearin...	[CE, MATSCI]
...	...	...
449	A portable USB-controlled potentiostat for pap...	[CPE, CHE]
450	Literature reviews on applying artificial inte...	[CPE, EDU]
451	A multi-parameterized water quality prediction...	[ENV, EE, CHE]
452	Semantic Segmentation on Medium-Resolution Sat...	[EE, CPE, OPTIC, EDU]
453	Reducing the defects of a-pillar stamping part...	[METAL, EDU, MATSCI]

454 rows × 2 columns

รูปที่ 2 แสดงตัวอย่างข้อมูล Text จากการรวม Title และ Abstract

2. นำข้อมูล Text ที่ได้มากำจัดคำ Stop word ด้วยการเรียกแหล่งเก็บ Stop word ภาษาอังกฤษของ nltk มาใช้พร้อมทั้งเพิ่มเติม Stop word บางคำที่มักเจอในงานวิจัย เช่น คำระบุอย่างจำนวน one, two, three คำเชื่อมอื่นๆอย่าง among, beside เป็นต้น

```
nltk.download('stopwords')
stop_words = set(stopwords.words('english'))
stop_words.update(['zero','one','two','three','four','five','six','seven','eight','nine','ten','may','also','across','among','beside','however','yet','within'])
re_stop_words = re.compile(r"\b(" + "|".join(stop_words) + ")\\W", re.I)

def removeStopWords(sentence):
    global re_stop_words
    return re_stop_words.sub(" ", sentence)
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

รูปที่ 3 แสดงตัวอย่างโค้ดการตัด Stop word

3. สร้างฟังก์ชันเพื่อนำไปใช้กับข้อมูล Text โดยมีการใช้คำสั่ง compile เพื่อกำหนดการจับคู่เพื่อการตัด stop word และใช้คำสั่ง sub เพื่อเรียกรูปแบบการจับคู่ที่ compile ไว้มาใช้ตัดคำ stop word จากนั้นก็นำฟังก์ชันมา map กับข้อมูล Text
4. สร้างฟังก์ชัน text_preprocessing เพื่อจัดการกับข้อมูลให้เรียบร้อยมากขึ้นด้วยการใช้คำสั่ง sub เป็นหลักในการแปลคำย่อ เช่น I'm เป็น I am หรือลบอักขระพิเศษ เพื่อให้ข้อมูลสะอาดและแบบจำลองสามารถวิเคราะห์ได้ดีขึ้น

5. เนื่องจาก labels ยังคงอยู่ในรูปของ list จึงต้องนำข้อมูลมาแปลงเป็นอยู่ในรูป Multi label ด้วยการใช้คำสั่ง fit_transfrom ด้วย MultiLabelBinarizer() และนำผลที่ได้ไปรวมกับ dataframe เก่าโดยตั้งชื่อและละ label ตามคลาสต่างๆ

```
labels = df['labels']
mlb = MultiLabelBinarizer()
one_hot_labels = mlb.fit_transform(labels)

encoded_labels_df = pd.DataFrame(one_hot_labels, columns=mlb.classes_)
encoded_labels_df = encoded_labels_df.astype(int)
encoded_labels_df.head()
```

	AGRI	BME	CE	CHE	CPE	EDU	EE	ENV	IE	MATENG	MATH	MATSCI	ME	METAL	NANO	OPTIC	PE	SAFETY
0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0
1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	1	0
4	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0

รูปที่ 4 แสดงตัวอย่างผลการแสดงการแปลงข้อมูลเป็น one hot label

6. นำข้อมูลชื่อคลาสต่างๆมาแปลงเป็น dict แบบตัวเลขเป็นชื่อคลาสและคลาสเป็นชื่อตัวเลขเพื่อเหมาะแก่การนำไปแปลผลหลังจากได้ผลลัพธ์จากแบบจำลอง

```
df2 = df.copy()
df2.drop(inplace=True,columns=labels)
df2
```

	Text	one_hot_labels
0	activated carbon derived bacterial cellulose u...	[0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, ...
1	algorithm static hand gesture recognition usin...	[0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
2	alternative redundant residue number system co...	[0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, ...
3	comparative study wax inhibitor performance po...	[0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, ...
4	undrained lower bound solutions end bearing ca...	[0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, ...
...	...	...
449	portable usb controlled potentiostat paper bas...	[0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
450	literature reviews applying artificial intelli...	[0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
451	multi parameterized water quality prediction m...	[0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, ...
452	semantic segmentation medium resolution satell...	[0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, ...
453	reducing defects pillar stamping part automoti...	[0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, ...

454 rows × 2 columns

รูปที่ 5 แสดงตัวอย่างข้อมูลสุดท้ายหลังการทำ Preprocessing

Chapter 3: Model

1. เริ่มต้นจากการแบ่งข้อมูลเพื่อการเรียนรู้ (train) และปรับพารามิเตอร์ (validation) โดยการแบ่ง train : validation เป็น 8 : 2
2. จากนั้นนำข้อมูลที่ได้มารวมเป็น dict ด้วยคำสั่ง dataset เพื่อง่ายต่อการนำมาปรับเปลี่ยน หรือการ map เข้ากับฟังก์ชันแปลงเป็น token ตามที่รูปแบบที่ต้องการก่อนนำไปเข้าแบบจำลอง

```
df_train, df_val = train_test_split(df_n, random_state=32, test_size=0.2, shuffle=True)

train_dataset = Dataset.from_pandas(df_train,preserve_index=False)
val_dataset = Dataset.from_pandas(df_val,preserve_index=False)

dataset_dict = DatasetDict({"train": train_dataset, "val": val_dataset})
print(dataset_dict)
```

```
DatasetDict({
    train: Dataset({
        features: ['Text', 'AGRI', 'BME', 'CE', 'CHE', 'CPE', 'EDU', 'EE', 'ENV', 'IE', 'MATENG', 'MATH', 'MATSCI', 'ME', 'METAL', 'NANO', 'OPTIC', 'PE', 'SAFETY'],
        num_rows: 363
    })
    val: Dataset({
        features: ['Text', 'AGRI', 'BME', 'CE', 'CHE', 'CPE', 'EDU', 'EE', 'ENV', 'IE', 'MATENG', 'MATH', 'MATSCI', 'ME', 'METAL', 'NANO', 'OPTIC', 'PE', 'SAFETY'],
        num_rows: 91
    })
})
```

รูปที่ 6 แสดงผลการรวมเป็น dict

3. สร้างฟังก์ชัน preprocess_data ซึ่งเป็นกระบวนการแปลงจากคำเป็น token โดยคำสั่ง tokenizer และกำหนด labels ของแต่ละ text แบบรวมเป็น column เดียวเพื่อการนำไปใช้
4. นำฟังก์ชัน preprocess_data มา map เข้ากับข้อมูล text เก็บในตัวแปล encoded_dataset จากนั้นก็ทิ้ง colume อื่นๆ ที่ไม่ได้ใช้งานเหลือเพียงจากการแปลงเป็น token (input_ids, token_type_ids, attention_mask) และ labels

```
example = encoded_dataset['train'][0]
print(example.keys())
```

```
dict_keys(['input_ids', 'token_type_ids', 'attention_mask', 'labels'])
```

```
print(tokenizer.decode(example['input_ids']))
print(example['labels'])
```

```
[CLS] identification zinc ion battery via equivalent circuit model. © ieee. notable zinc ion batteries ( zibs ) show coming trend replace li ion batteries ( libs ) future.
[0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 1.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0]
```

รูปที่ 7 แสดงตัวอย่างผลการ tokenize

5. แปลงให้ข้อมูลเป็นรูปแบบ torch เพื่อการนำไปเข้าแบบจำลองด้วยฟังก์ชัน set_format
6. เรียกใช้แบบจำลองที่มีค่าน้ำหนักการให้การเรียนรู้มาแล้วด้วยคำสั่ง AutoModelForSequenceClassification.from_pretrain และตั้งค่า parameter ให้ถูกต้องตามเหมาะสม

```
model = AutoModelForSequenceClassification.from_pretrained("bert-base-uncased",
                                                           problem_type="multi_label_classification",
                                                           num_labels=18,
                                                           id2label=id2label,
                                                           label2id=label2id)
```

```
Some weights of BertForSequenceClassification were not initialized from the model checkpoint at bert-base-unc
You should probably TRAIN this model on a down-stream task to be able to use it for predictions and inference
```

```
#best param
batch_size = 8
learning_rate = 1e-4
num_train_epochs = 35
weight_decay = 1e-4
```

รูปที่ 8 แสดงตัวอย่างผลเรียกใช้แบบจำลองและการตั้งค่าพารามิเตอร์

7. เนื่องจากจะใช้คำสั่ง trainer จึงต้องมีการตั้งค่า Argument ต่างๆ ตามต้องการด้วยคำสั่ง TrainningAgruments เช่น learning rate epoch การแสดงผลบน wanb

```
training_args = TrainingArguments(
    f"bert-finetuned-sem_eval-english",
    evaluation_strategy = "epoch",
    save_strategy = "epoch",
    learning_rate=learning_rate,
    per_device_train_batch_size=batch_size,
    per_device_eval_batch_size=batch_size,
    num_train_epochs=num_train_epochs,
    weight_decay=weight_decay,
    load_best_model_at_end=True,
    metric_for_best_model="f1",
    report_to="wandb"
)
```

รูปที่ 9 แสดงตัวอย่างการตั้งค่า Argument

8. จากนั้นมีการกำหนดการแสดงวิธีการวัดผลของแบบจำลองด้วยฟังก์ชัน multi_label_metrics ที่เป็นการให้สามารถกำหนดข้อมูลจริงและข้อมูลทำนายเพื่อการนำไปวิเคราะห์ค่า f1-score, roc-auc, accuracy และ compute_matrics ปรับให้เหมาะแก่การนำไปใช้ใน trainer โดยมีการเรียกใช้ multi_label_metrics ในการคำนวณผล

```
def multi_label_metrics(predictions, labels):
    sigmoid = torch.nn.Sigmoid()
    probs = sigmoid(torch.Tensor(predictions))
    y_pred = np.zeros(probs.shape)
    y_pred[np.where(probs >= 0.5)] = 1
    y_true = labels
    f1_macro_average = f1_score(y_true=y_true, y_pred=y_pred, average='macro')
    roc_auc = roc_auc_score(y_true, y_pred, average = 'macro')
    accuracy = accuracy_score(y_true, y_pred)
    metrics = {'f1': f1_macro_average,
               'roc_auc': roc_auc,
               'accuracy': accuracy}
    return metrics
```

รูปที่ 10 แสดงฟังก์ชันที่จะใช้ในการประเมินค่าแบบจำลอง

9. ตั้งค่าแบบจำลองโดยคำสั่ง trainer เช่น แบบจำลองที่ใช้ ชุดข้อมูลทดลองและทดสอบ

```
trainer = Trainer(
    model,
    training_args,
    train_dataset=encoded_dataset["train"],
    eval_dataset=encoded_dataset["val"],
    tokenizer=tokenizer,
    compute_metrics=compute_metrics
)
```

รูปที่ 11 แสดงตัวอย่างการตั้งค่า Trainer()

10. ให้การเรียนรู้แบบจำลองด้วย .train()

```
trainer.train()
```

```
You're using a BertTokenizerFast tokenizer. Please note that with a fast tokenizer, using the `__call__` me
[1840/1840 10:24, Epoch 40/40]
```

Epoch	Training Loss	Validation Loss	F1	Roc Auc	Accuracy
1	No log	0.393972	0.069371	0.524574	0.032967
2	No log	0.387262	0.075788	0.529629	0.032967
3	No log	0.377185	0.163010	0.555444	0.043956
4	No log	0.354077	0.259312	0.591489	0.043956
5	No log	0.333976	0.308931	0.609716	0.065934
6	No log	0.329523	0.455248	0.674812	0.065934
7	No log	0.308956	0.406398	0.657280	0.098901
8	No log	0.320146	0.403024	0.654648	0.109890
9	No log	0.301980	0.493443	0.705707	0.131868
10	No log	0.306768	0.495883	0.702692	0.142857

รูปที่ 12 แสดงตัวอย่างผลการให้การเรียนรู้เมื่อใช้คำสั่ง .train()

11. และวัดผลด้วย .evaluate()

```
trainer.evaluate()
```

```
[12/12 00:00]
{'eval_loss': 0.31220418214797974,
 'eval_f1': 0.5964830914285502,
 'eval_roc_auc': 0.7629086229564422,
 'eval_accuracy': 0.14285714285714285,
 'eval_runtime': 0.7394,
 'eval_samples_per_second': 123.071,
 'eval_steps_per_second': 16.229,
 'epoch': 40.0}
```

รูปที่ 13 แสดงตัวอย่างผลการให้การเรียนรู้เมื่อใช้คำสั่ง .evaluate()

12. บันทึกแบบจำลองด้วย save_model และเรียกมาใช้ด้วยคำสั่ง from_pretrain เช่นเดิมเพื่อการนำมาทำนาย

```
trainer.save_model("/content/drive/MyDrive/2110531-data-science-2023-01/best1")
```

```
checkpoint = "/content/drive/MyDrive/2110531-data-science-2023-01/best1"
```

```
tokenizer = AutoTokenizer.from_pretrained(checkpoint)
model = AutoModelForSequenceClassification.from_pretrained(checkpoint,
                                                           problem_type="multi_label_classification",
                                                           num_labels=18,
                                                           id2label=id2label,
                                                           label2id=label2id)
```

รูปที่ 14 แสดงตัวอย่างการบันทึกแบบจำลอง

13. จากนั้นวน text ละแถวในข้อมูลชุดทดสอบมาทำขั้นตอนเดียวกันการเตรียมข้อมูลก่อนเข้าแบบจำลองโดยย่อและนำไปสร้างเป็นตารางโดยมีการกำหนดลำดับของชื่อของ column ให้ตรงกับรูปแบบที่ต้องส่งตรวจ

```
predictions = []
for i in df_test['Text']:
  encoding = tokenizer(i, return_tensors="pt")
  encoding = {k: v.to(trainer.model.device) for k,v in encoding.items()}
  outputs = trainer.model(**encoding)
  logits = outputs.logits
  sigmoid = torch.nn.Sigmoid()
  probs = sigmoid(logits.squeeze().cpu())
  prediction = np.zeros(probs.shape)
  prediction[np.where(probs >= 0.5)] = 1
  predictions.append(np.array(prediction))
```

รูปที่ 15 แสดงตัวอย่างการทำนายผลชุดทดสอบ

14. บันทึกผลลัพธ์ที่ได้ด้วยคำสั่ง to_csv

```
df_pred.to_csv('/content/drive/MyDrive/2110531-data-science-2023-01/predict.csv', index=False)
```

```
df_verify = pd.read_csv('/content/drive/MyDrive/2110531-data-science-2023-01/predict.csv')
print(df_verify)
```

```
          id  CE  ENV  BME  PE  METAL  ME  EE  CPE  OPTIC  NANO  CHE  MATENG  \
0    001eval   0    0    0   0      0   0   1    0      0     0    0       0
1    002eval   0    0    0   0      0   0   0    0      0     1    1       1
2    003eval   0    0    1   0      0   0   0    0      0     0    1       0
3    004eval   0    0    0   0      0   0   0    1      0     0    0       0
4    005eval   0    0    0   0      0   0   0    1      0     0    0       0
..       ...  ..  ...  ...  ..    ...  ..  ..  ...    ...   ...  ...     ...
146  147eval   0    1    0   0      0   0   0    0      0     0    1       1
147  148eval   0    0    0   0      0   0   1    1      0     0    0       0
148  149eval   0    0    0   0      1   1   0    0      0     0    1       0
149  150eval   0    0    0   0      0   0   0    0      0     0    1       0
150  151eval   0    1    0   0      0   0   0    0      0     0    0       0

     AGRI  EDU  IE  SAFETY  MATH  MATSCI
0       0    0   0       0     0       0
1       0    0   0       0     0       1
2       0    0   0       0     0       0
3       0    0   0       0     1       0
4       0    0   0       0     0       0
..    ...  ...  ..     ...   ...     ...
146     0    0   0       0     0       1
147     0    0   0       0     0       0
148     0    0   0       0     1       1
149     0    0   0       0     0       0
150     0    0   0       0     0       0

[151 rows x 19 columns]
```

รูปที่ 16 แสดงตัวอย่างผลสำเร็จที่ได้จากการทำนาย

Chapter 4: Results

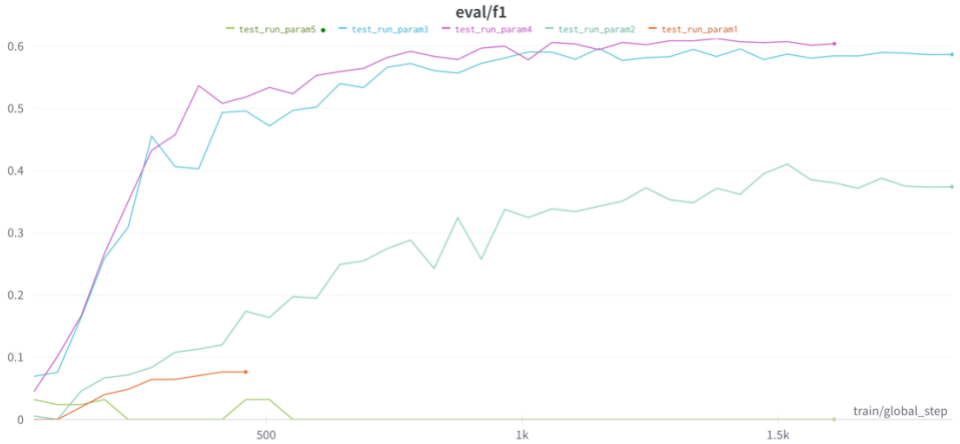
เลือกตัวอย่างผลการให้การเรียนรู้จากการปรับพารามิเตอร์ทั้งหมด 6 ผลการทดสอบโดยมีพารามิเตอร์ ดังนี้

ตารางที่ 1 แสดงผลการทดลองปรับพารามิเตอร์

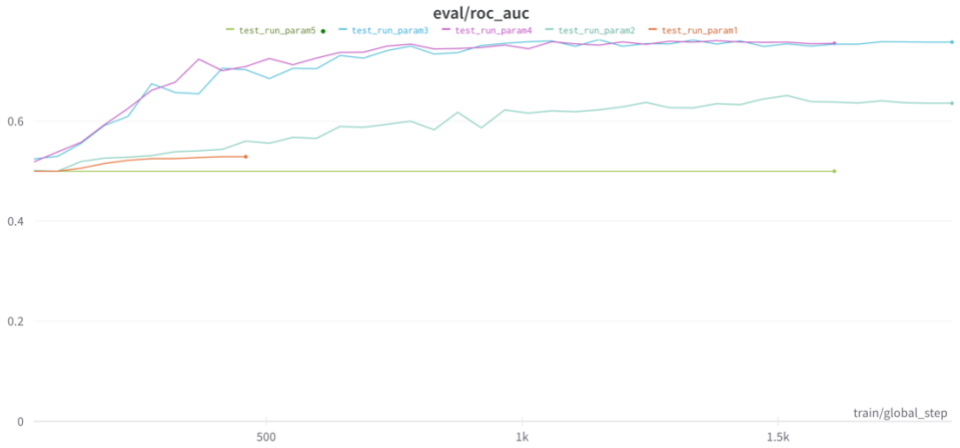
Model	Parameter				Eval Loss	Final Macro f1-score
	Batch size	Learning rate	Epochs	Weight decay		
1	8	1e-5	10	1e-3	0.3867	0.0764
2	8	1e-5	40	1e-3	0.3239	0.3739
3	8	1e-4	40	1e-3	0.3119	0.5867
4	8	1e-4	35	1e-4	0.3206	0.6039
5	8	1e-3	35	1e-4	0.4268	0.0321
6	16	1e-4	35	1e-4	Out of memory	

จากตารางพบว่าพารามิเตอร์ในแบบจำลองที่ 4 ได้ผล Macro f1-score สูงสุด จึงเลือกเป็นแบบจำลองที่นำไปทำนายผล

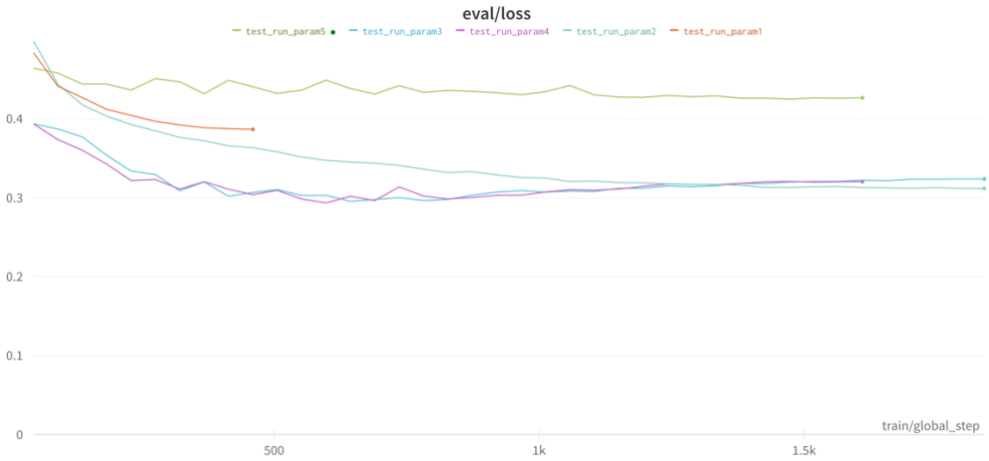
กราฟเปรียบเทียบผลแต่ละแบบจำลอง



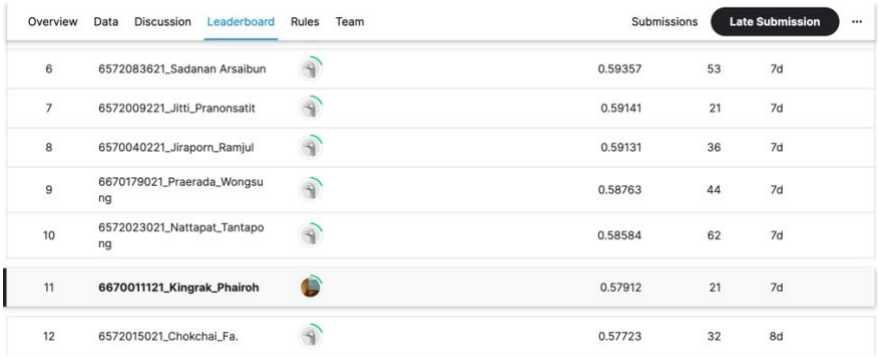
รูปที่ 17 กราฟแสดงผลเปรียบเทียบค่า Macro f1-score แต่ละแบบจำลอง



รูปที่ 18 กราฟแสดงผลเปรียบเทียบค่า roc-auc แต่ละแบบจำลอง



รูปที่ 19 กราฟแสดงผลเปรียบเทียบค่า Loss แต่ละแบบจำลอง



รูปที่ 20 แสดง Leaderboard ใน Kaggle competition

จากภาพผลแสดงอันดับใน Kaggle นิสิตได้คะแนน Macro f1-score เท่ากับ 0.57912 จัดเป็นอันดับที่ 11 จาก 30

Chapter 5: Discussion

จากการทดสอบพบว่าเมื่อมีการปรับจำนวนรอบการให้เรียนรู้ (Epoch) Loss มีค่าลดลงเล็กน้อยและ Macro f1-score มีค่าสูงขึ้นถึงประมาณ 4 เท่าต่อมาเมื่อต้องการลดความ overfit จึงมีการทดสอบลดค่า learning rate ในแบบจำลองที่ 3 พบว่าค่า loss ลดลงเล็กน้อยแต่ค่า Macro f1-score มีค่าสูงขึ้นอีกสองเท่าเท่ากับ 0.5867 จากนั้นมีการทดสอบลดจำนวนรอบให้การเรียนรู้เนื่องจากเมื่อสังเกตที่ค่า loss และ Macro f1-score ในส่วนท้ายค่อนข้างคงที่ การลดจำนวนรอบให้การเรียนรู้จะช่วยลดความ overfit และการใช้หน่วยความจำโดยไม่จำเป็น และมีการลดค่า weight decay ผลที่ได้พบว่าค่า loss เพิ่มขึ้นเล็กน้อยแต่ค่า Macro f1-score เพิ่มขึ้นร้อยละ 3.4 เป็นค่าเท่ากับ 0.6039 ต่อมาจากแนวโน้มการแปรผกผันของค่า Macro f1-score และ learning rate จึงทดสอบเพิ่มค่า learning rate ซึ่งผลที่ได้พบว่าค่า loss สูงขึ้นมากตรงข้ามกับค่า Macro f1-score ที่ลดลงอย่างมากแสดงถึงการลด learning rate เป็น 0.001 มีผลลบต่อการทำนายของแบบจำลอง นั่นคือการปรับ learning rate ให้เหมาะสมมีผลต่อการทำนายของแบบจำลองอย่างมาก สุดท้ายเมื่อนำแบบจำลองที่ 4ไปทำนายกับข้อมูลชุดทดสอบได้ผล Macro f1-score เท่ากับ 0.5791 ซึ่งมีค่าใกล้เคียงกับผลที่ได้จากการทดสอบข้อมูลชุด Validation

Chapter 6: Conclusion

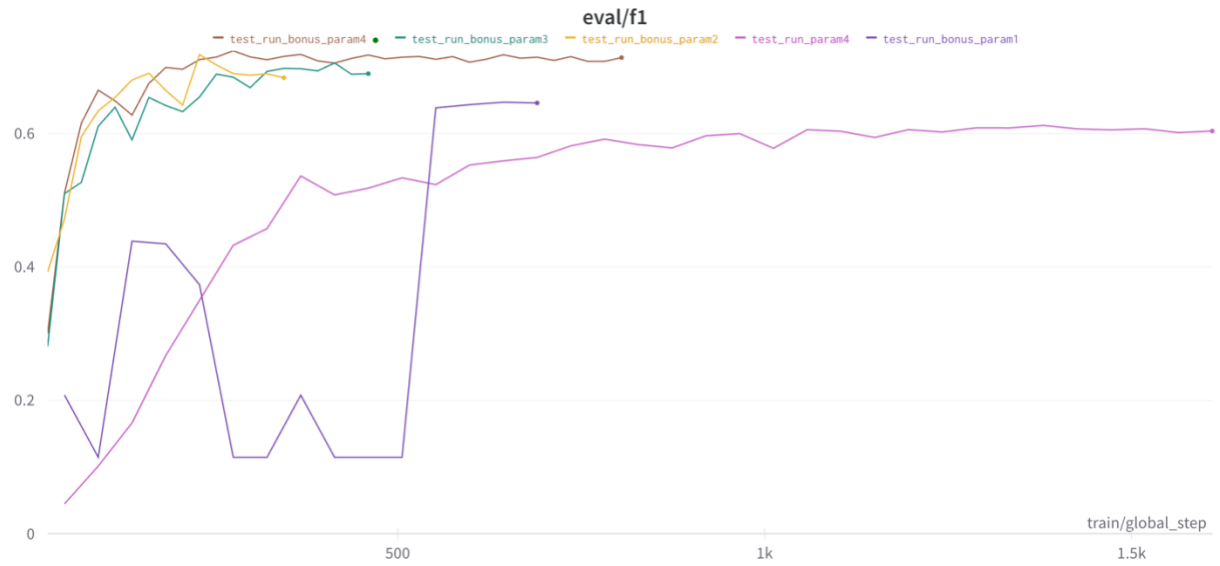
จากการทดสอบแบบจำลองที่ได้ค่า Macro f1-score ที่สูงสุดมีการตั้งค่า Batch size เท่ากับ 8, Learning rate เท่ากับ 1e-4, Epochs เท่ากับ 35 และ Weight decay เท่ากับ 1e-4 โดยได้ Macro f1-score เท่ากับ 0.6039 และเมื่อนำไปทำนายกับข้อมูลชุดทดสอบได้ผล Macro f1-score เท่ากับ 0.5791 ซึ่งสอดคล้องกัน และจากการปรับพารามิเตอร์พบว่าจำนวนรอบการให้เรียนรู้และ learning rate มีผลอย่างมากต่อการทำนายแบบจำลองแต่ต้องหากปรับเพิ่มรอบมากไปแบบจำลองอาจ overfit และใช้หน่วยความจำอย่างไม่จำเป็น และการปรับ learning rate มากหรือน้อยเกินไปจะทำให้แบบจำลองเรียนรู้ได้อย่างไม่มีประสิทธิภาพ

อย่างไรก็ตามการปรับพารามิเตอร์ในการทดสอบครั้งนี้ค่อนข้างไม่ละเอียดและขาดการพิจารณาในส่วนของความไม่สมดุลของข้อมูลเนื่องจากขาดการจัดการเวลาในเวลาที่จำกัดและการให้การเรียนรู้แบบจำลองแต่ละครั้งค่อนข้างใช้เวลา

Bonus

ตารางที่ 2 แสดงผลการทดลองปรับพารามิเตอร์ที่ดีที่สุดใน dataset เดิมเทียบกับปรับพารามิเตอร์ใน dataset bonus

Model	Parameter				Eval Loss	Macro f1-score
	Batch size	Learning rate	Epochs	Weight decay		
4	8	1e-4	35	1e-4	0.3206	0.6039
Bonus/1	8	1e-4	15	1e-4	0.3867	0.6460
Bonus/2	16	1e-4	15	1e-4	0.3239	0.6837
Bonus/3	16	1e-4	20	1e-4	0.3119	0.69



รูปที่ 21 แสดงผลเปรียบเทียบค่า Macro f1-score ผลจาก dataset เดิมในแบบจำลองที่ได้ค่า Macro f1-score สูงสุดเทียบกับ dataset bonus

จากการทดสอบพบว่าเนื่องจาก dataset bonus มีจำนวน labels น้อยกว่า dataset เดิมนั่นคือใช้หน่วยความจำในการประมวลที่น้อยกว่าจึงสามารถปรับขนาด batch size เพิ่มขึ้นได้และสามารถใช้รอบการให้การเรียนรู้ที่ไม่สูงเท่าเดิมก็ได้ผลการประเมินที่สูงกว่าเพราะแบบจำลองเรียนรู้รูปแบบข้อมูลของแค่ labels 7 labels แต่ถ้าเพิ่มจนมีรอบการเรียนรู้ที่เท่ากันก็จำให้มีความแม่นยำมากขึ้นแต่ต้องพิจารณาถึงความ overfit