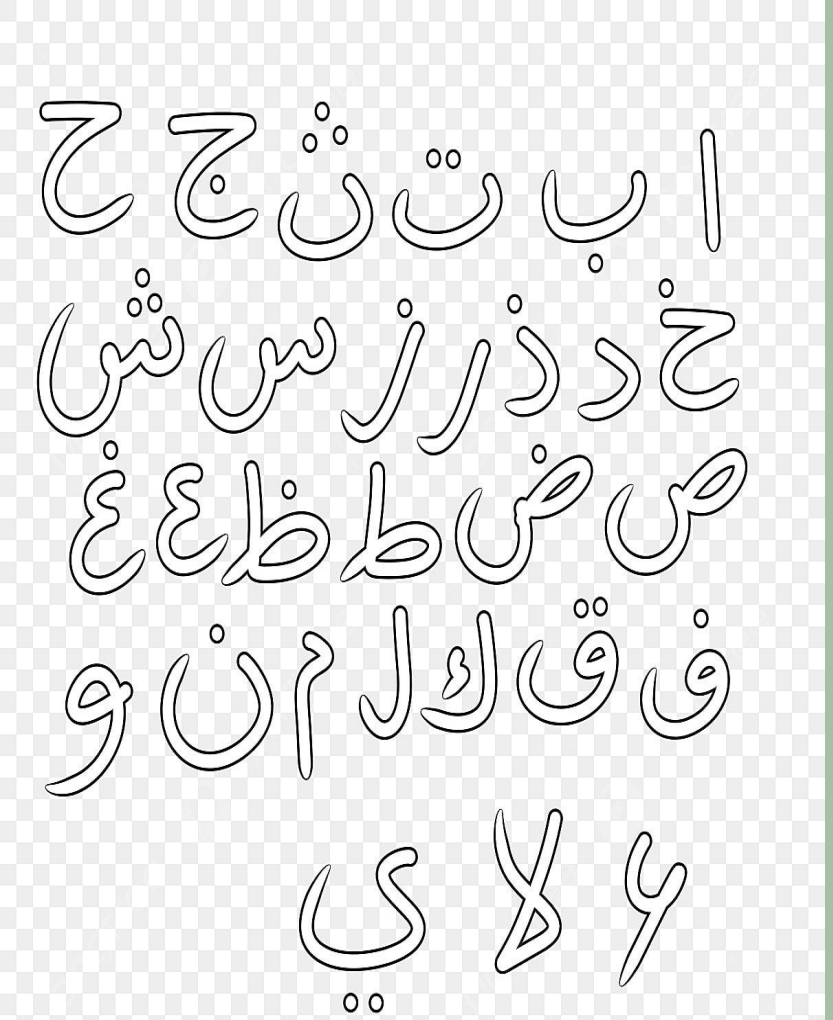
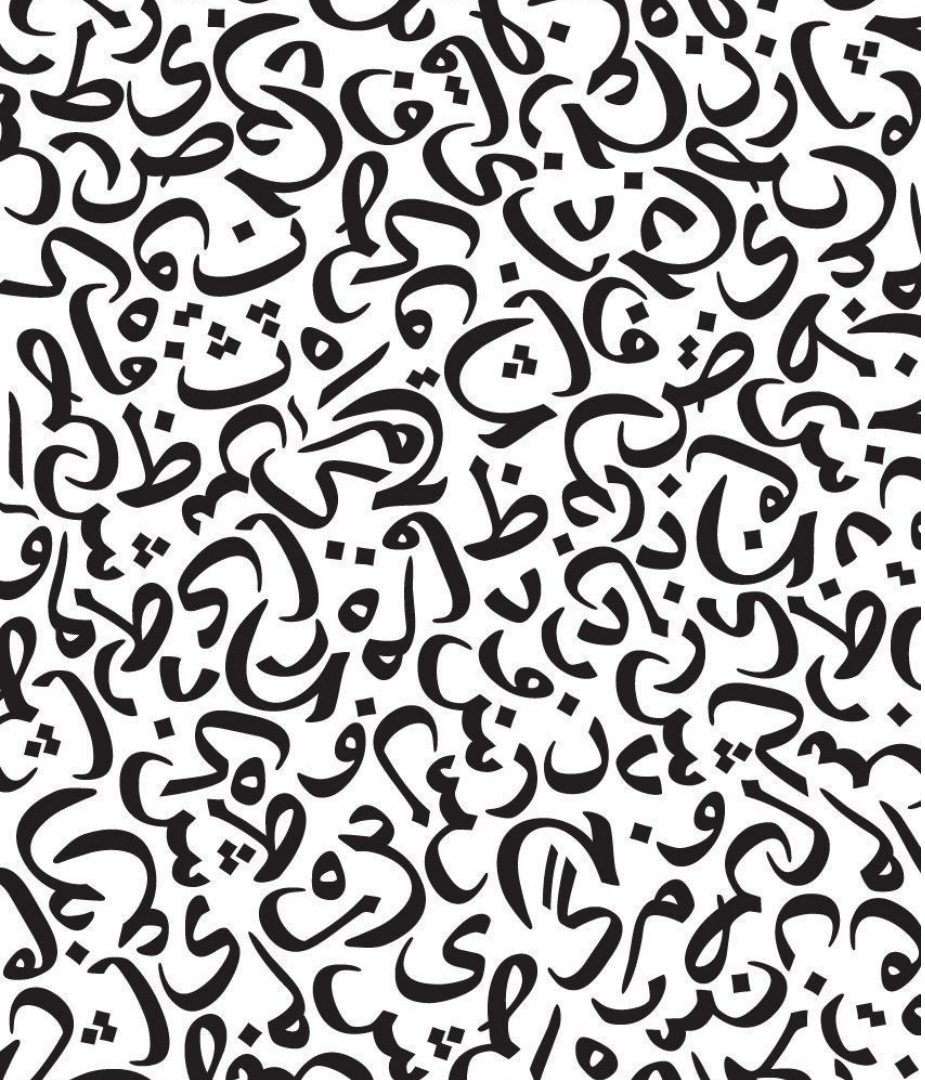



# Sentiment Analysis For Arabic Reviews

By Iman Attia & Ahmed Essam

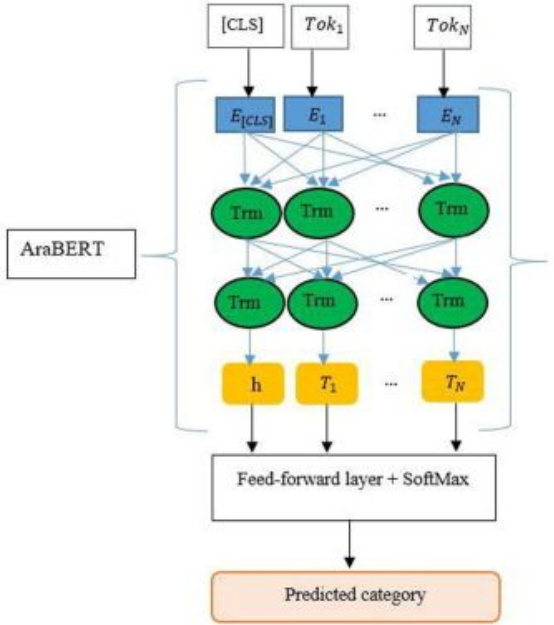
# Table of contents



01

# Original Model from literature

# AraBERTvo.2 Model Architecture





```
{
  "architectures": [
    "BertForMaskedLM"
  ],
  "attention_probs_dropout_prob": 0.1,
  "hidden_act": "gelu",
  "hidden_dropout_prob": 0.1,
  "hidden_size": 768,
  "initializer_range": 0.02,
  "intermediate_size": 3072,
  "max_position_embeddings": 512,
  "model_type": "bert",
  "num_attention_heads": 12,
  "num_hidden_layers": 12,
  "type_vocab_size": 2,
  "vocab_size": 64000
}
```

# Original Model Evaluation Metrics

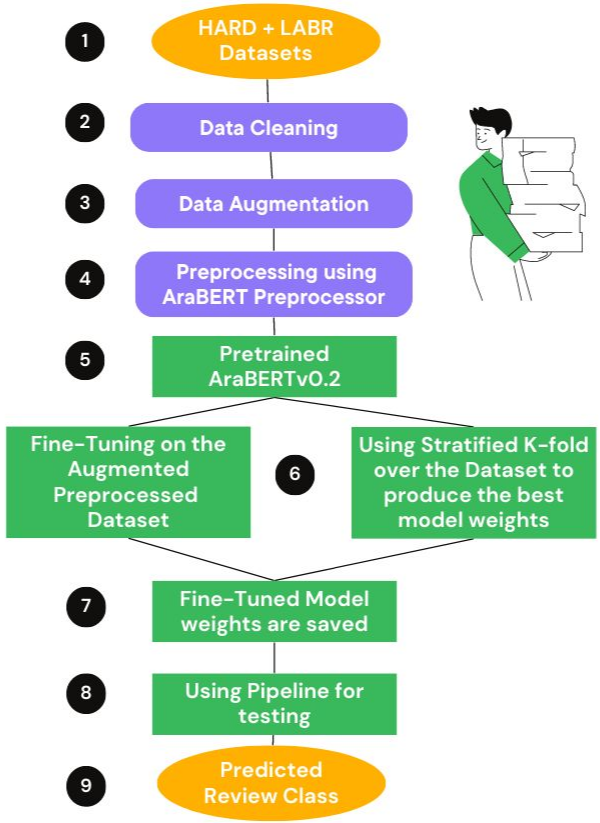
The following metrics are used by the original model:

1. macro_F1 Score
2. Precision
3. Recall
4. Accuracy

02

# Proposed/Final Model Architecture

# Proposed System



# Our Proposed Model Evaluation Metrics

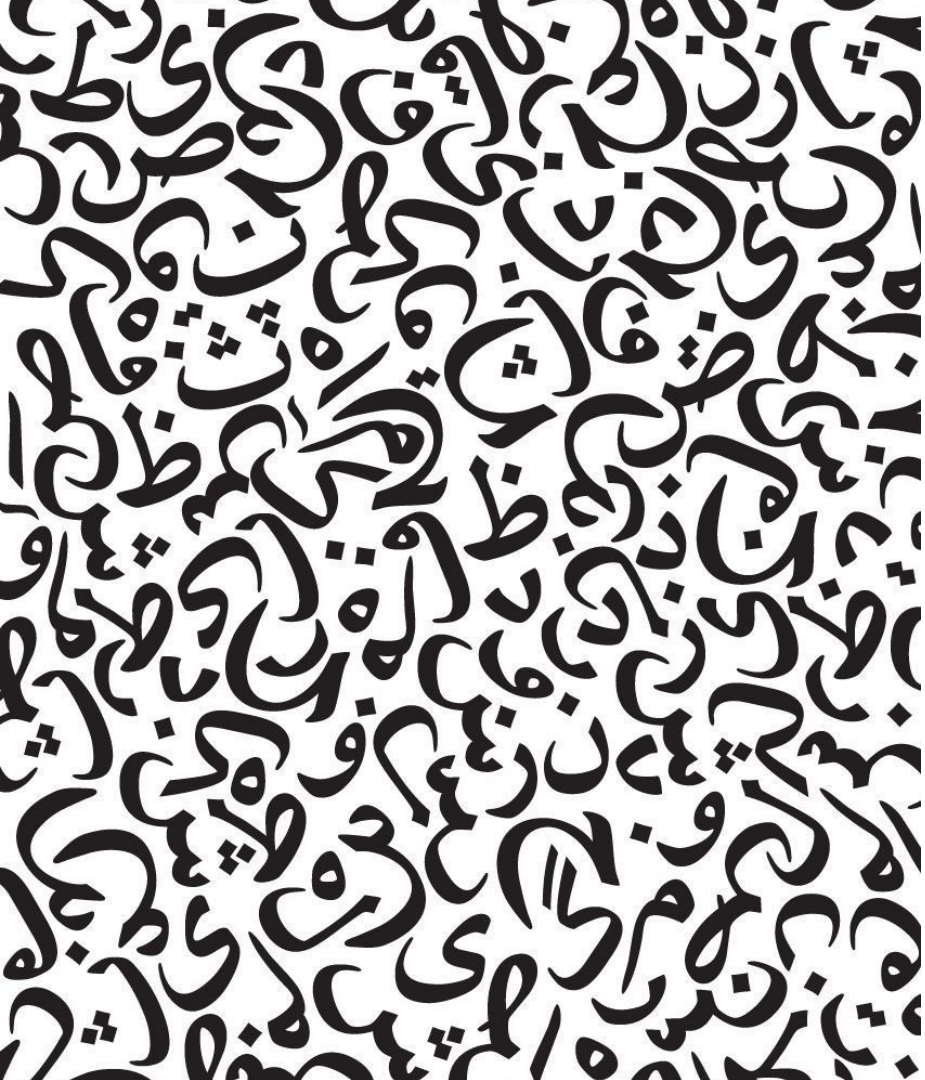
We replaced the Accuracy metric used by the original model by Metrics number 4 and 5, therefore our final model evaluation metrics are:

1. macro_F1 Score
2. Precision
3. Recall
4. **Confusion Matrix**
5. **Mean Square Error (MSE) and Mean Absolute Error( MAE) over the Confusion Matrix, But Why?**

# Final Model Architecture

Out[76]:

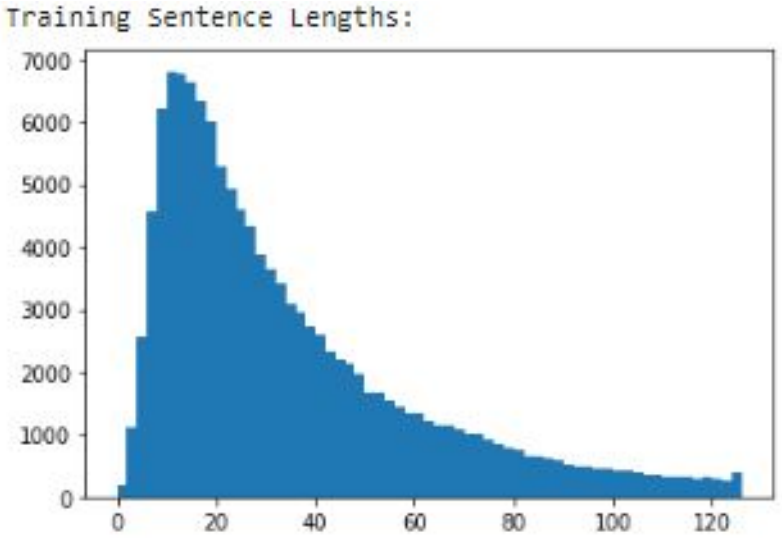
```
BertConfig {
  "_name_or_path": "aubmindlab/bert-base-arabertv02",
  "architectures": [
    "BertForMaskedLM"
  ],
  "attention_probs_dropout_prob": 0.3,
  "classifier_dropout": null,
  "hidden_act": "gelu",
  "hidden_dropout_prob": 0.3
  "hidden_size": 768,
  "id2label": {
    "0": "LABEL_0",
    "1": "LABEL_1",
    "2": "LABEL_2",
    "3": "LABEL_3",
    "4": "LABEL_4"
  },
  "initializer_range": 0.02,
  "intermediate_size": 3072,
  "label2id": {
    "LABEL_0": 0,
    "LABEL_1": 1,
    "LABEL_2": 2,
    "LABEL_3": 3,
    "LABEL_4": 4
  },
  "layer_norm_eps": 1e-12,
  "max_position_embeddings": 512,
  "model_type": "bert",
  "num_attention_heads": 12,
  "num_hidden_layers": 18,
  "pad_token_id": 0,
  "position_embedding_type": "absolute",
  "transformers_version": "4.12.2",
  "type_vocab_size": 2,
  "use_cache": true,
  "vocab_size": 64000
}
```

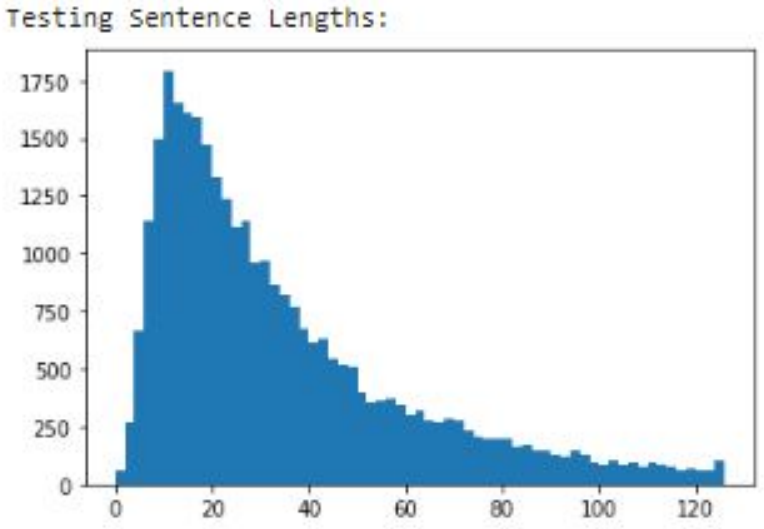


03

# Experiments Results

# Tokenization Experiment Results

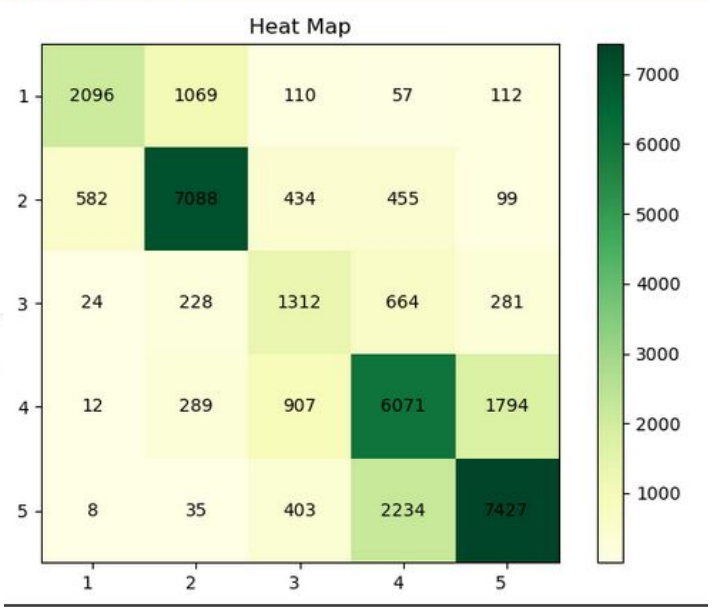



Based on the obtained results, the optimal value for the maximum sentence length value was decided to be 128.

# Original Model + Dataset without Augmentation

| Epoch | Training Loss | Validation Loss | Macro F1 | Precision | Recall | Mse | Mae |
|---|---|---|---|---|---|---|---|
| 1 | 0.518200 | 0.691505 | 0.672471 | 0.680655 | 0.670828 | 0.529934 | 0.358823 |
| 2 | 0.490800 | 0.705915 | 0.670444 | 0.683307 | 0.660791 | 0.522210 | 0.350626 |



# Original Model + Dataset without Augmentation: Stratified K-Fold K = 5

For more details:
https://drive.google.com/drive/folders/11pi-h-GfnTMMZoIZNf0bRDlWMpaiPVQH?usp=sharing

```
In [50]: all_results

Out[50]: [{'eval_loss': 0.6427580714225769,
  'eval_macro_f1': 0.6645222435461182,
  'eval_precision': 0.6786077248302471,
  'eval_recall': 0.6550847704135817,
  'eval_runtime': 243.2824,
  'eval_samples_per_second': 111.118,
  'eval_steps_per_second': 0.871,
  'epoch': 2.0},
 {'eval_loss': 0.6336689591407776,
  'eval_macro_f1': 0.6703106820500471,
  'eval_precision': 0.6848695606299462,
  'eval_recall': 0.6601832945413929,
  'eval_runtime': 242.1702,
  'eval_samples_per_second': 111.628,
  'eval_steps_per_second': 0.875,
  'epoch': 2.0},
 {'eval_loss': 0.6431758403778076,
  'eval_macro_f1': 0.6625399022794172,
  'eval_precision': 0.6773276187214844,
  'eval_recall': 0.6522371834999138,
  'eval_runtime': 243.0615,
  'eval_samples_per_second': 111.219,
  'eval_steps_per_second': 0.872,
  'epoch': 2.0},
 {'eval_loss': 0.6431753039360046,
  'eval_macro_f1': 0.6660182427024578,
  'eval_precision': 0.6764239580495653,
  'eval_recall': 0.6583477215204432,
  'eval_runtime': 243.2135,
  'eval_samples_per_second': 111.145,
  'eval_steps_per_second': 0.872,
  'epoch': 2.0},
 {'eval_loss': 0.6416257619857788,
  'eval_macro_f1': 0.666804663729899,
  'eval_precision': 0.6788351708456811,
  'eval_recall': 0.6584490231615513,
  'eval_runtime': 242.2533,
  'eval_samples_per_second': 111.586,
  'eval_steps_per_second': 0.875,
  'epoch': 2.0}]
```

## Choosing Best Model Based on Macro F1

```
In [45]: all_results

Out[45]: [{'eval_loss': 0.6501943469047546,
  'eval_macro_f1': 0.6549400284865332,
  'eval_precision': 0.6779045872677578,
  'eval_recall': 0.6436055249357192,
  'eval_mse': 0.563237524507084,
  'eval_mae': 0.3676987385787741,
  'eval_runtime': 243.4181,
  'eval_samples_per_second': 111.056,
  'eval_steps_per_second': 0.871,
  'epoch': 2.0},
 {'eval_loss': 0.6336689591407776,
  'eval_macro_f1': 0.6703106820500471,
  'eval_precision': 0.6848695606299462,
  'eval_recall': 0.6601832945413929,
  'eval_mse': 0.5220656234972071,
  'eval_mae': 0.3482040469056383,
  'eval_runtime': 243.9437,
  'eval_samples_per_second': 110.817,
  'eval_steps_per_second': 0.869,
  'epoch': 2.0},
 {'eval_loss': 0.6553136706352234,
  'eval_macro_f1': 0.6388861171125579,
  'eval_precision': 0.675710557182538,
  'eval_recall': 0.6237323309247513,
  'eval_mse': 0.551437132393741,
  'eval_mae': 0.3644064661709762,
  'eval_runtime': 243.5226,
  'eval_samples_per_second': 111.008,
  'eval_steps_per_second': 0.871,
  'epoch': 2.0},
 {'eval_loss': 0.6544126868247986,
  'eval_macro_f1': 0.6613061653463002,
  'eval_precision': 0.6732779562738591,
  'eval_recall': 0.6521438018282775,
  'eval_mse': 0.5473142941698728,
  'eval_mae': 0.36375406925125775,
  'eval_runtime': 243.789,
  'eval_samples_per_second': 110.883,
  'eval_steps_per_second': 0.87,
  'epoch': 2.0},
 {'eval_loss': 0.6535153388977051,
  'eval_macro_f1': 0.6503959399621972,
  'eval_precision': 0.6757822523310288,
  'eval_recall': 0.6360706240399017,
  'eval_mse': 0.5404335602249186,
  'eval_mae': 0.36168245042912106,
  'eval_runtime': 243.8816,
  'eval_samples_per_second': 110.841,
  'eval_steps_per_second': 0.869,
  'epoch': 2.0}]
```

## Choosing Best Model Based on MSE

# Fine-Tuned Model + Dataset without Augmentation

```
[96] #start the training
     trainer.train()

***** Running training *****
  Num examples = 135163
  Num Epochs = 2
  Instantaneous batch size per device = 16
  Total train batch size (w. parallel, distributed & accumulation) = 32
  Gradient Accumulation steps = 2
  Total optimization steps = 8448
```

[8448/8448 1:53:12, Epoch 2/2]

| Epoch | Training Loss | Validation Loss | Macro F1 | Precision | Recall | Accuracy |
|---|---|---|---|---|---|---|
| 1 | 0.659800 | 0.644547 | 0.668231 | 0.684728 | 0.657483 | 0.715161 |
| 2 | 0.593400 | 0.638626 | 0.672826 | 0.685965 | 0.662903 | 0.719156 |

```
***** Running Evaluation *****
  Num examples = 33791
  Batch size = 128
Saving model checkpoint to ./train/checkpoint-4224
Configuration saved in ./train/checkpoint-4224/config.json
Model weights saved in ./train/checkpoint-4224/pytorch_model.bin
***** Running Evaluation *****
  Num examples = 33791
  Batch size = 128
Saving model checkpoint to ./train/checkpoint-8448
Configuration saved in ./train/checkpoint-8448/config.json
Model weights saved in ./train/checkpoint-8448/pytorch_model.bin


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


Loading best model from ./train/checkpoint-8448 (score: 0.6728263040239597).
TrainOutput(global_step=8448, training_loss=0.6411350893251824, metrics={'train_runtim
1.244, 'total_flos': 1.7889270947056128e+16, 'train_loss': 0.6411350893251824, 'epoch'
```

```
[109] # pipe("Some Text")
      pipe("شيء بشع ، مش عارفة اقول ايه حسبي الله و نعم الوكيل")

[[{'label': 1, 'score': 0.04524393752217293},
  {'label': 2, 'score': 0.07740205526351929},
  {'label': 3, 'score': 0.017723921686410904},
  {'label': 4, 'score': 0.4159225523471832},
  {'label': 5, 'score': 0.443707495927810 67}]]
```

```
[ ] pipe("تجربة جميلة!")

[[{'label': 1, 'score': 0.04527897387742996},
  {'label': 2, 'score': 0.07736202329397202},
  {'label': 3, 'score': 0.017712419852614403},
  {'label': 4, 'score': 0.41595980525016785},
  {'label': 5, 'score': 0.4436868131160736}]]
```

```
[111] pipe("تجربة جميلة جدا لقد سعدت بوجودي في هذا المكان ارشحه و بشدة لجميع اصدقائي !")

[[{'label': 1, 'score': 0.04525246098637581},
  {'label': 2, 'score': 0.07739639282226562},
  {'label': 3, 'score': 0.017721906304359436},
  {'label': 4, 'score': 0.4159254729747772},
  {'label': 5, 'score': 0.4437038004398346}]]
```

```
pipe("تجربة متوسطة لا بأس بها!")

[[{'label': 1, 'score': 0.04526425153017044},
  {'label': 2, 'score': 0.07737893611192703},
  {'label': 3, 'score': 0.017717478796839714},
  {'label': 4, 'score': 0.415942519903183},
  {'label': 5, 'score': 0.4436967968940735}]]
```

# Fine-Tuned Model + Dataset without Augmentation

## Learning rate

| | Macro F1 Score | Precision | Recall | Accuracy |
|---|---|---|---|---|
| **Learning rate = 1e-5** | 0.6645 | 0.6788 | 0.6436 | 0.7172 |
| **Learning rate = 1e-4** | 0.2521 | 0.2372 | 0.2057 | 0.2254 |

# Fine-Tuned Model + Dataset without Augmentation

## Batch Size & Number of Epochs

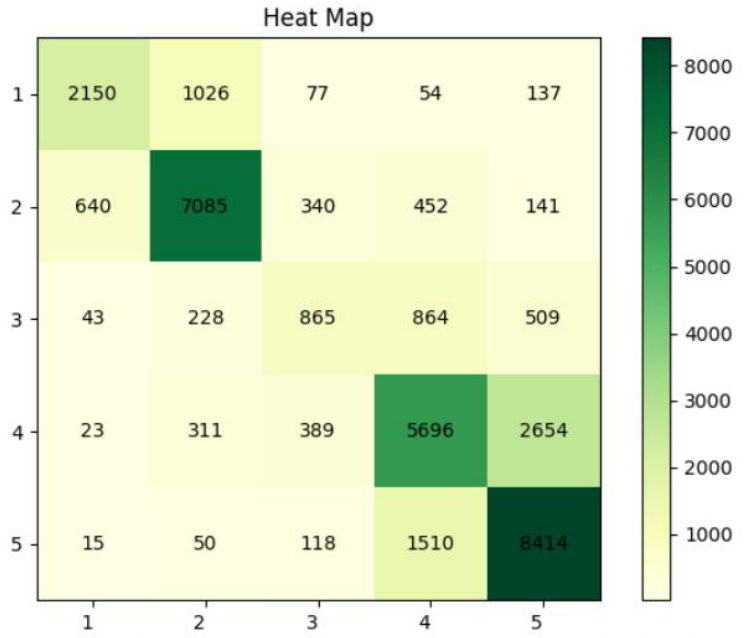
1. We tried different batch sizes: 8, 16, 32, 64. The **optimal batch size: 16**

2. We tried different number of epochs, but the accuracy didn't change much with increasing epochs, so we were training on **2 epochs to save time**

# Fine-Tuned Model + Dataset without Augmentation

## num_hidden_laye

| Epoch | Training Loss | Validation Loss | Macro F1 | Precision | Recall | Mse | Mae |
|---|---|---|---|---|---|---|---|
| 1 | 0.650500 | 0.639980 | 0.662296 | 0.684134 | 0.649528 | 0.548519 | 0.357580 |



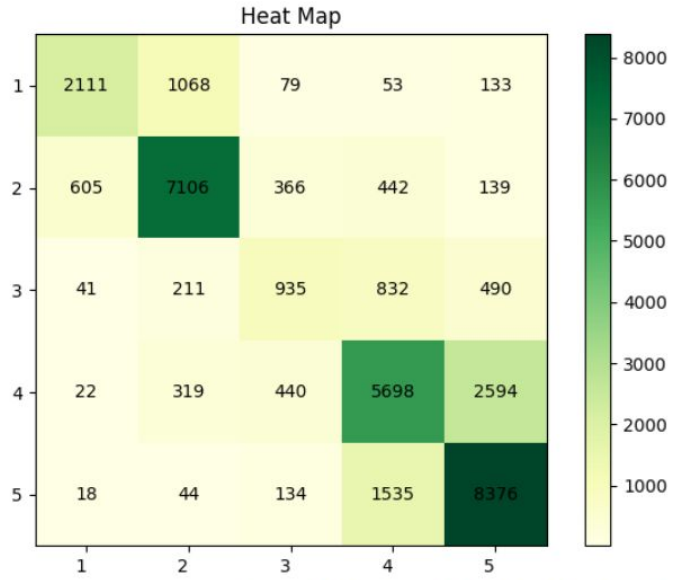
# Fine-Tuned Model + Dataset without Augmentation num_hidden_layers = 18

```
[50] # pipe("Some Text")
     pipe("شيء بشع ، مش عارفة اقول ايه حسبي الله و نعم الوكيل")

[[{'label': 1, 'score': 0.4190787971019745},
  {'label': 2, 'score': 0.24736180901527405},
  {'label': 3, 'score': 0.08399207144975662},
  {'label': 4, 'score': 0.10334315896034241},
  {'label': 5, 'score': 0.14622412621974945}]]

     pipe("تجربة جميلة!")

[[{'label': 1, 'score': 0.025452319532632828},
  {'label': 2, 'score': 0.05893201380968094},
  {'label': 3, 'score': 0.2580272853374481},
  {'label': 4, 'score': 0.39261776208877563},
  {'label': 5, 'score': 0.26497066020965576}]]

     pipe("تجربة جميلة جدا لقد سعدت بوجودي في هذا المكان ارشحه و بشدة لجميع اصدقائي !")

[[{'label': 1, 'score': 0.033385783433914185},
  {'label': 2, 'score': 0.028596464544534683},
  {'label': 3, 'score': 0.029811125248670578},
  {'label': 4, 'score': 0.2041410207748413},
  {'label': 5, 'score': 0.7040656208992004}]]

[53] pipe("تجربة متوسطة لا بأس بها!")

[[{'label': 1, 'score': 0.06277720630168915},
  {'label': 2, 'score': 0.35132700204849243},
  {'label': 3, 'score': 0.3776682913303375},
  {'label': 4, 'score': 0.16395290195941925},
  {'label': 5, 'score': 0.04427459463477135}]]
```

# Fine-Tuned Model + Dataset without Augmentation

num_attention_head

| Epoch | Training Loss | Validation Loss | Macro F1 | Precision | Recall | Mse | Mae |
|---|---|---|---|---|---|---|---|
| 1 | 0.650600 | 0.640637 | 0.665070 | 0.685028 | 0.652620 | 0.544790 | 0.356278 |



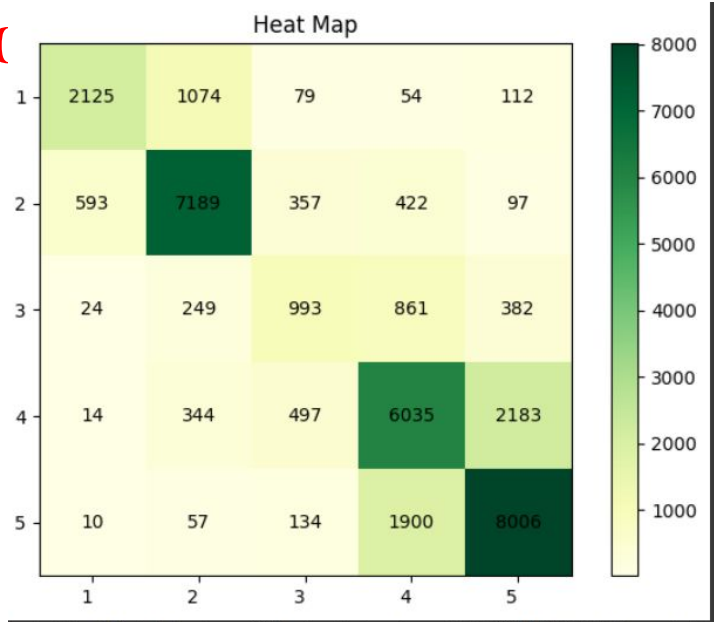
# Fine-Tuned Model + Dataset without Augmentation num_hidden_layers = 18

```
[50] # pipe("Some Text")
     pipe("شيء بشع ، مش عارفة اقول ايه حسبي الله و نعم الوكيل")

[[{'label': 1, 'score': 0.4190787971019745},
  {'label': 2, 'score': 0.24736180901527405},
  {'label': 3, 'score': 0.08399207144975662},
  {'label': 4, 'score': 0.10334315896034241},
  {'label': 5, 'score': 0.14622412621974945}]]

     pipe("تجربة جميلة!")

[[{'label': 1, 'score': 0.025452319532632828},
  {'label': 2, 'score': 0.05893201380968094},
  {'label': 3, 'score': 0.2580272853374481},
  {'label': 4, 'score': 0.39261776208877563},
  {'label': 5, 'score': 0.26497066020965576}]]

     pipe("تجربة جميلة جدا لقد سعدت بوجودي في هذا المكان ارشحه و بشدة لجميع اصدقائي !")

[[{'label': 1, 'score': 0.033385783433914185},
  {'label': 2, 'score': 0.028596464544534683},
  {'label': 3, 'score': 0.029811125248670578},
  {'label': 4, 'score': 0.2041410207748413},
  {'label': 5, 'score': 0.7040656208992004}]]

[53] pipe("تجربة متوسطة لا بأس بها!")

[[{'label': 1, 'score': 0.06277720630168915},
  {'label': 2, 'score': 0.35132700204849243},
  {'label': 3, 'score': 0.3776682913303375},
  {'label': 4, 'score': 0.16395290195941925},
  {'label': 5, 'score': 0.04427459463477135}]]
```

# Fine-Tuned Model + Dataset without Augmentation

hidden_dropout_pro

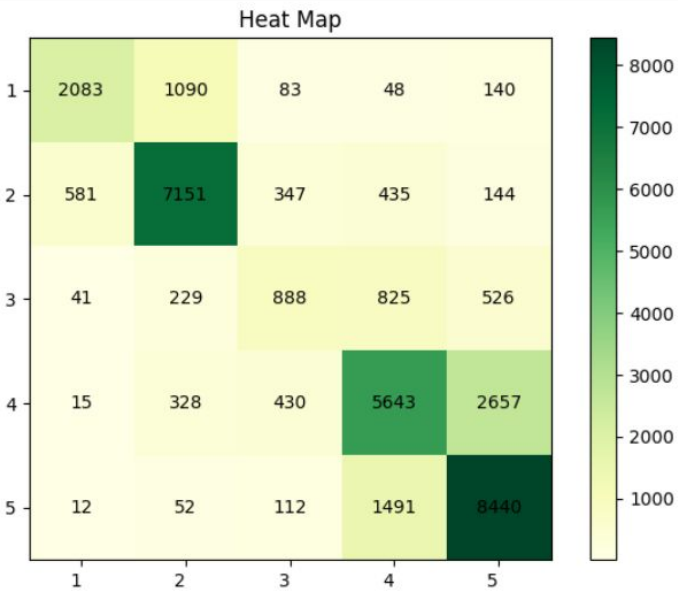
| Epoch | Training Loss | Validation Loss | Macro F1 | Precision | Recall | Mse | Mae |
|---|---|---|---|---|---|---|---|
| 1 | 0.612300 | 0.634376 | 0.672375 | 0.690085 | 0.660081 | 0.509130 | 0.344411 |



# Fine-Tuned Model + Dataset without Augmentation

## hidden_size = 10

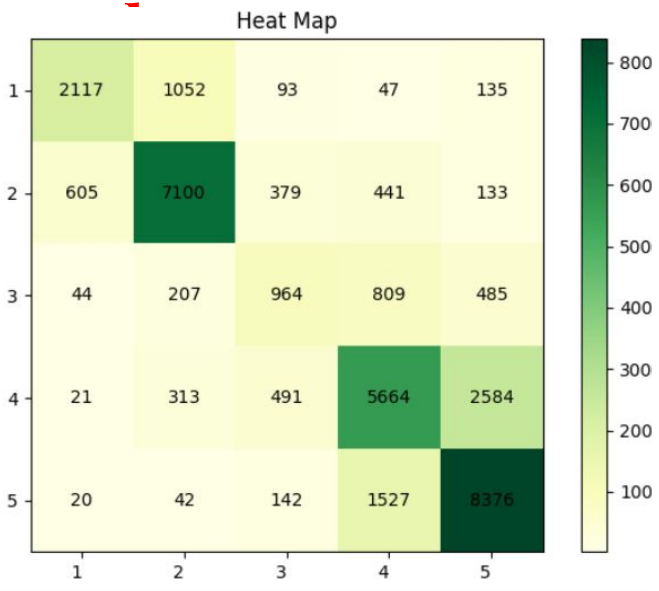
| Epoch | Training Loss | Validation Loss | Macro F1 | Precision | Recall | Mse | Mae |
|---|---|---|---|---|---|---|---|
| 1 | 0.652400 | 0.640208 | 0.661815 | 0.685031 | 0.648341 | 0.547868 | 0.357640 |



# Fine-Tuned Model + Dataset without Augmentation

attention_probs_dropout_

| Epoch | Training Loss | Validation Loss | Macro F1 | Precision | Recall | Mse | Mae |
|---|---|---|---|---|---|---|---|
| 1 | 0.651400 | 0.641038 | 0.665737 | 0.683458 | 0.654392 | 0.544317 | 0.356278 |



# Data Augmentation: Back Translation

In [11]:

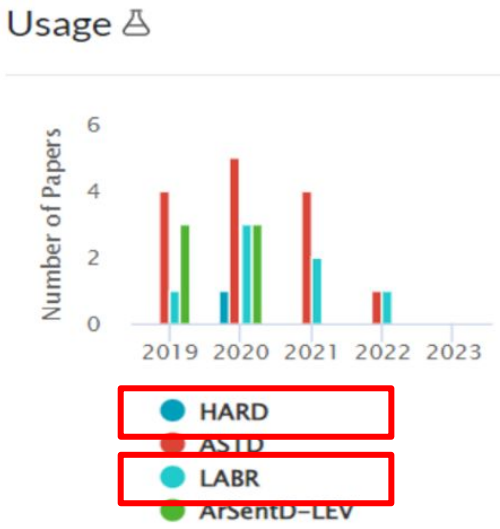
```python
from transformers import MarianMTModel, MarianTokenizer
ar_en_tokenizer = MarianTokenizer.from_pretrained('Helsinki-NLP/opus-mt-ar-en')
# Initialize tokenizer and model for Arabic to English translation
ar_en_model = MarianMTModel.from_pretrained('Helsinki-NLP/opus-mt-ar-en')

# Initialize tokenizer and model for English to Arabic translation
en_ar_tokenizer = MarianTokenizer.from_pretrained('Helsinki-NLP/opus-mt-en-ar')
en_ar_model = MarianMTModel.from_pretrained('Helsinki-NLP/opus-mt-en-ar')


def back_translate_arabic_to_english_and_back(input_text):
    # Translate input text from Arabic to English
    encoded_input = ar_en_tokenizer.encode(input_text, return_tensors='pt')
    translated = ar_en_model.generate(encoded_input)
    en_text = ar_en_tokenizer.decode(translated[0], skip_special_tokens=True)

    # Translate English text back to Arabic
    encoded_input = en_ar_tokenizer.encode(en_text, return_tensors='pt')
    translated = en_ar_model.generate(encoded_input)
    ar_text = en_ar_tokenizer.decode(translated[0], skip_special_tokens=True)

    return ar_text
```

# Dataset Before VS. After Data Augmentation

```
5    50177
4    45503
2    43752
1    17321
3    12201
Name: label, dtype: int64
[2 5 1 4 3]
object
```

```
5    50177
4    45503
2    43752
1    25941
3    24402
Name: label, dtype: int64
[2 5 1 4 3]
```

# Finalized Augmented Dataset

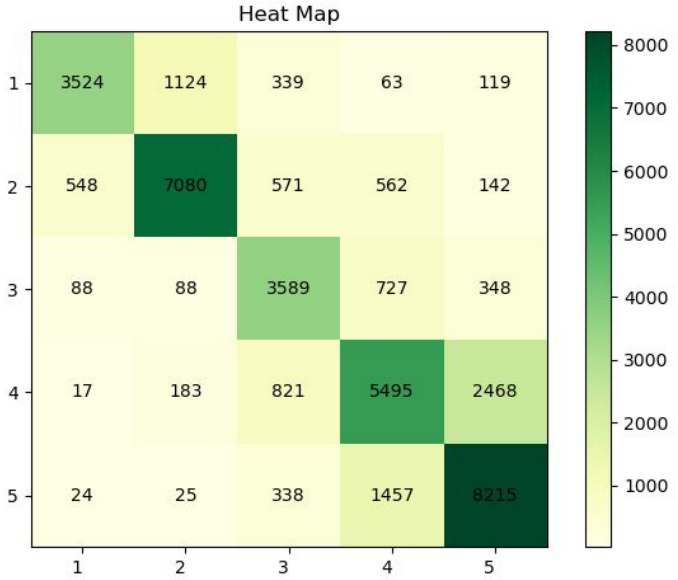



| | text | label |
|---|---|---|
| 0 | .ممتاز". النظافة والطاقم متعاون" | 2 |
| 1 | ...استثنائي. سهولة إنهاء المعاملة في الاستقبال. ل | 5 |
| 2 | ...استثنائي. انصح بأختيار الاسويت و بالاخص غرفه ر | 5 |
| 3 | ... استغرب تقييم الفندق كخمس نجوم". لا شي. يستحق" | 1 |
| 4 | ...جيد. المكان جميل وهاديء. كل شي جيد ونظيف بس كا | 4 |
| ... | ... | ... |
| 189770 | ...الكثير من المآسي، الكثير من الألم، النهاية صعب | 3 |
| 189771 | ...يقدم شكسبير تقنية لم يراها أحد من قبل المسرح ف | 3 |
| 189772 | ...والقصة جميلة، على الرغم من أن فصولها قصيرة جدا | 3 |
| 189773 | ...لقد تأثرت بقدرة إميلي نصر إله الله على وصف أفك | 3 |
| 189774 | ...وبيوت الناس تدخل وقلوبهم تدخل ويداك تتصافحان و | 3 |

189775 rows × 2 columns

# Original Model + Dataset with Augmentation

| Epoch | Training Loss | Validation Loss | Macro F1 | Precision | Recall | Mse | Mae |
|---|---|---|---|---|---|---|---|
| 0 | 0.617600 | 0.610479 | 0.730962 | 0.739070 | 0.729370 | 0.520274 | 0.338111 |
| 1 | 0.546100 | 0.596906 | 0.736808 | 0.743910 | 0.731661 | 0.499038 | 0.326887 |



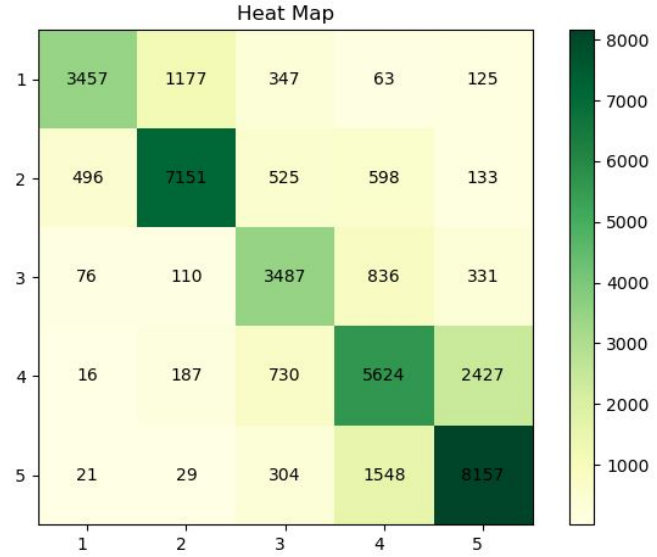
# Original Model + Dataset with Augmentation

```
In [51]: # pipe("Some Text")
         pipe("شيء بشع")

Out[51]: [[{'label': 1, 'score': 0.6866488456726074},
           {'label': 2, 'score': 0.028496138751506805},
           {'label': 3, 'score': 0.20206588506698608},
           {'label': 4, 'score': 0.031180646270513535},
           {'label': 5, 'score': 0.051608480513095856}]]

In [53]: pipe("تجربة جميلة!")

Out[53]: [[{'label': 1, 'score': 0.0046085393987596035},
           {'label': 2, 'score': 0.02114017866551876},
           {'label': 3, 'score': 0.2025539129972458},
           {'label': 4, 'score': 0.465402752161026},
           {'label': 5, 'score': 0.30629467964172363}]]

In [54]: pipe("تجربة جميلة جدا لقد سعدت بوجودي في هذا المكان ارشحه و بشدة لجميع اصدقائي !")

Out[54]: [[{'label': 1, 'score': 0.014071810990571976},
           {'label': 2, 'score': 0.0012757601216435432},
           {'label': 3, 'score': 0.0006819891277700663},
           {'label': 4, 'score': 0.026197051629424095},
           {'label': 5, 'score': 0.9577733278274536}]]

In [55]: pipe("تجربة متوسطة لا بأس بها!")

Out[55]: [[{'label': 1, 'score': 0.09717616438865662},
           {'label': 2, 'score': 0.7438490390777588},
           {'label': 3, 'score': 0.10853373259305954},
           {'label': 4, 'score': 0.045564860105514526},
           {'label': 5, 'score': 0.0048761432990431786}]]
```

# Fine-Tuned Model + Dataset with Augmentation

num_hidden_layers = 18, attention_probs_dropout_prob = 0.3
and hidden_dropout_prob = 0.3

| Epoch | Training Loss | Validation Loss | Macro F1 | Precision | Recall | Mse | Mae |
|---|---|---|---|---|---|---|---|
| 0 | 0.618600 | 0.612107 | 0.730068 | 0.740387 | 0.725876 | 0.519721 | 0.338348 |
| 1 | 0.550900 | 0.596446 | 0.737008 | 0.744676 | 0.731206 | 0.498511 | 0.327151 |



# Fine-Tuned Model + Dataset with Augmentation

num_hidden_layers = 18,
attention_probs_dropout_prob = 0.3
and hidden_dropout_prob = 0.3

```
In [68]: # pipe("Some Text")
         pipe("شيء بشع")

Out[68]: [[{'label': 1, 'score': 0.7259764671325684},
           {'label': 2, 'score': 0.1009884774684906},
           {'label': 3, 'score': 0.10653195530176163},
           {'label': 4, 'score': 0.02888152003288269},
           {'label': 5, 'score': 0.0376216396689415}]]

In [69]: pipe("تجربة جميلة!")

Out[69]: [[{'label': 1, 'score': 0.019848495721817017},
           {'label': 2, 'score': 0.03558487072587013},
           {'label': 3, 'score': 0.35487204790115356},
           {'label': 4, 'score': 0.32264527678489685},
           {'label': 5, 'score': 0.26704928278923035}]]

In [70]: pipe("تجربة جميلة جدا لقد سعدت بوجودي في هذا المكان ارشحه و بشدة لجميع اصدقائي !")

Out[70]: [[{'label': 1, 'score': 0.03264261782169342},
           {'label': 2, 'score': 0.007129787001758814},
           {'label': 3, 'score': 0.014228833839297295},
           {'label': 4, 'score': 0.10995500534772873},
           {'label': 5, 'score': 0.8360437750816345}]]

In [71]: pipe("تجربة متوسطة لا بأس بها!")

Out[71]: [[{'label': 1, 'score': 0.23254595696926117},
           {'label': 2, 'score': 0.46282801032066345},
           {'label': 3, 'score': 0.19583679735660553},
           {'label': 4, 'score': 0.0797029659152031},
           {'label': 5, 'score': 0.02908635139465332}]]
```

# Fine-Tuned Model + Dataset with Augmentation

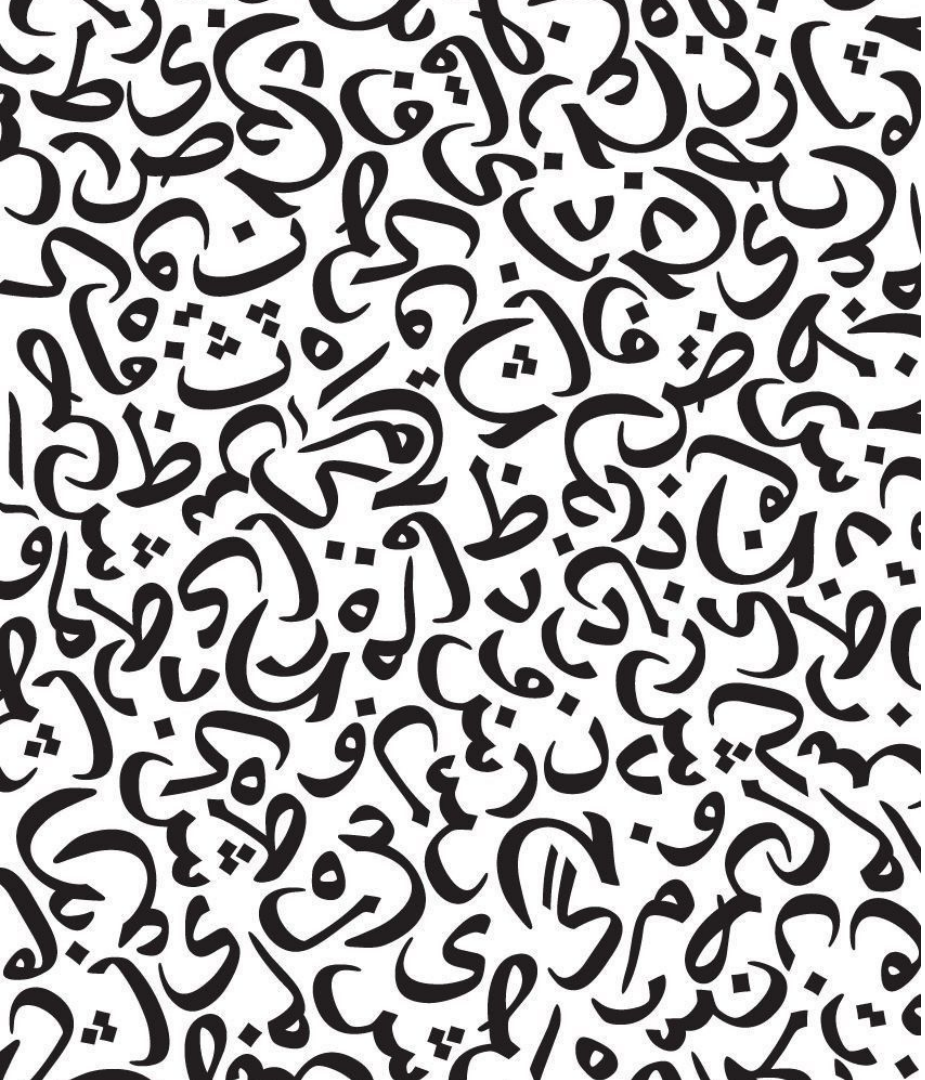
num_hidden_layers = 18, attention_probs_dropout_prob = 0.3
and hidden_dropout_prob = 0.3

## + Colloquial Language Test Cases

```
In [72]: pipe("مش حلو ومش وحش يعني هو شغال ,, الأوض مش أحسن حاجة بس اللي هو لو رايح علشان تنبسط هيبقى الموضوع بالنسبالك عادي")

Out[72]: [[{'label': 1, 'score': 0.18289142847061157},
           {'label': 2, 'score': 0.7359523773193359},
           {'label': 3, 'score': 0.06162182614207268},
           {'label': 4, 'score': 0.014566424302756786},
           {'label': 5, 'score': 0.004967943765223026}]]

In [73]: pipe("لا مش احسن حاجة خالص ")

Out[73]: [[{'label': 1, 'score': 0.6014888882637024},
           {'label': 2, 'score': 0.335239440202713},
           {'label': 3, 'score': 0.03949267417192459},
           {'label': 4, 'score': 0.01300591416656971},
           {'label': 5, 'score': 0.010773004963994026}]]
```

04

# Live Demo



05

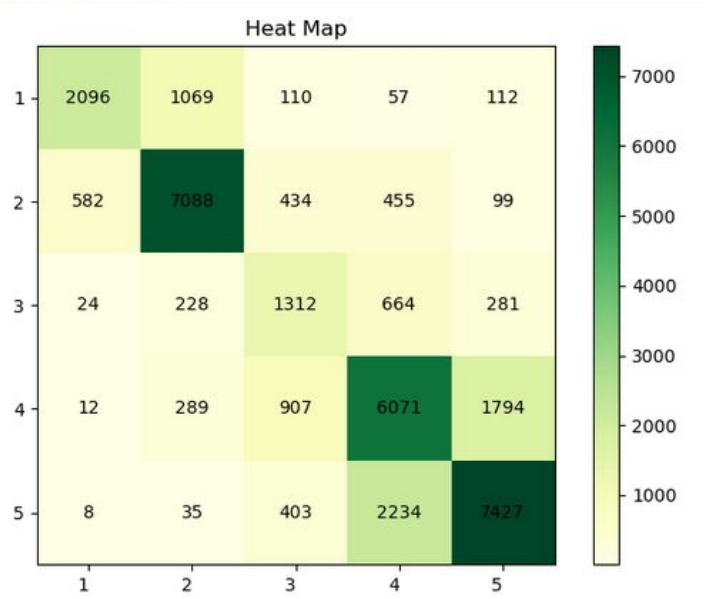
# Conclusion & Future Work

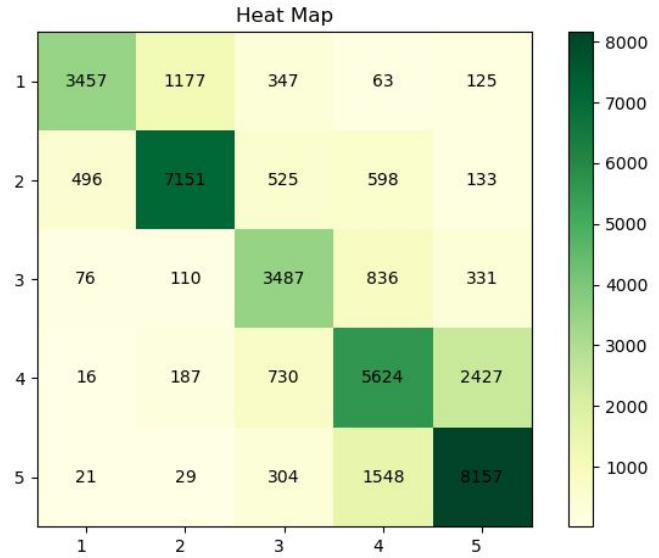
# Conclusion

| Evaluation Metric | Original Model | Our Best Model (Fine-Tuned +Augmented) |
|---|---|---|
| **Training Loss** | 0.490800 | 0.550900 |
| **Validation Loss** | 0.705915 | 0.596446 |
| **Macro F1** | 0.670444 | 0.737008 |
| **Precision** | 0.683307 | 0.744676 |
| **Recall** | 0.660791 | 0.731206 |
| **MSE** | 0.522210 | 0.498511 |
| **MAE** | 0.350626 | 0.327151 |

# Conclusion

## Original Model Heatmap VS. Our Best Model Heatmap



# Future Directions

For future considerations, we will be:

1. Appling **Interpretability Analysis** on our model to understand how it is making its predictions and which words have the most powerful effect on the model's decision.

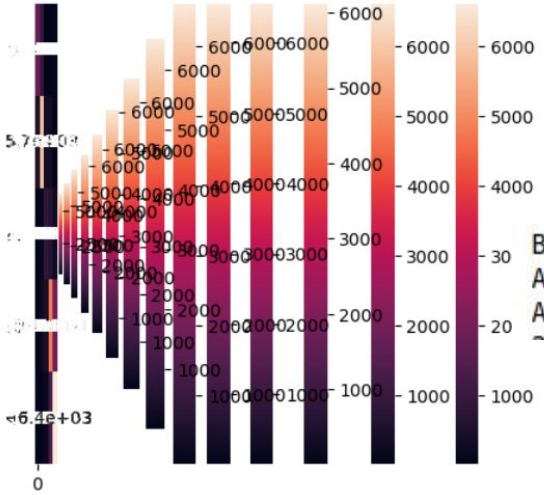
2. Experimenting with **more hyper-parameters** and see the combined effects of changing more than one hyperparameter together.

3. Implementing **a utility application** for our model, most probably a website, that will be available for the public usage and will contribute to the Arabic Sentiment Analysis Research.

06

# Each team member contribution

# A Sample of the Errors Faced throughout the Project



```
RuntimeError: CUDA out of memory. Tried to allocate 200.00 MiB (GPU 0; 15.78 GiB total
capacity; 14.56 GiB already allocated; 38.44 MiB free; 14.80 GiB reserved in total by
PyTorch) If reserved memory is >> allocated memory try setting max_split_size_mb to avoid
fragmentation. See documentation for Memory Management and PYTORCH_CUDA_ALLOC_CONF
```

```
Before:  ضعيف جداً . الاثاث قديم وريحة الدخان في كا مكان
An error occurred: the JSON object must be str, bytes or bytearray, not NoneType
After:  ['1 ,'ضعيف جداً . الاثاث قديم وريحة الدخان في كا مكان]
```

```
~/anaconda3/lib/python3.9/json/encoder.py in default(self, o)
    177
    178         """
--> 179         raise TypeError(f'Object of type {o.__class__.__name__} '
    180                         f'is not JSON serializable')
    181

TypeError: Object of type ndarray is not JSON serializable
```

```
Trainer is attempting to log a value of "[[2028 1091  122   78  125]
 [ 569 6912  458  594  125]
 [  63   89 1041  794  522]
 [  23  222  478 6012 2338]
 [  18   33  136 1900 8020]]" of type <class 'numpy.ndarray'> for key "eval/confusion_matrix" as a scalar. This invocation of Tensorboard's writer.add_scalar() is incorrect so we dropped this attribute.
Saving model checkpoint to ./train/checkpoint-3379
Configuration saved in ./train/checkpoint-3379/config.json
Model weights saved in ./train/checkpoint-3379/pytorch_model.bin
```

```
TypeError                                 Traceback (most recent call last)
/tmp/ipykernel_2884/4195924049.py in <module>
      1 #start the training
----> 2 trainer.train()

~/anaconda3/lib/python3.9/site-packages/transformers/trainer.py in train(self, resume_from_checkpoint, trial, ignore_keys_for_eval, **kwargs)
   1389
   1390             self.control = self.callback_handler.on_epoch_end(args, self.state, self.control)
-> 1391             self._maybe_log_save_evaluate(tr_loss, model, trial, epoch, ignore_keys_for_eval)
   1392
   1393             if DebugOption.TPU_METRICS_DEBUG in self.args.debug:

~/anaconda3/lib/python3.9/site-packages/transformers/trainer.py in _maybe_log_save_evaluate(self, tr_loss, model, trial, epoch, ignore_keys_for_eval)
   1493
   1494         if self.control.should_save:
-> 1495             self._save_checkpoint(model, trial, metrics=metrics)
   1496             self.control = self.callback_handler.on_save(self.args, self.state, self.control)
   1497
```

# Each team member contribution

| Iman | Ahmed |
| --- | --- |
| Creating Training Datasets and splitting the data between the training and testing data. | Preprocessing the Data using ArabertPreprocessor() and applying tokenization experiments. |
| Writing the init_model() and the compute_metrics(p) functions to instantiate the model. | Data augmentation of the dataset. Used pipeline from transformers to predict using the saved model. |
| Working on K-fold function with cross-validation to find the best hyper parameters. | Setting up the TrainingArguments and the transformers trainer method to start the regular training. |
| Writing the function that ensemble all the cross validation models generated from the K-fold. | Improved the K-fold experiments to use heat maps and MSE as the main matrices to determine the best model. |
| Experimenting different learning rates, number of attention heads, hidden dropout probability, and the augmented data | Fine tuning the hidden layer activation function, batch sizes, attention_probs_dropout_prob, and number of hidden layers. |
| Fixed some of the errors that we faced during training. | Fixed some of the errors that we faced during fine tuning. |

# Thanks!

Do you have any questions?

iman.elsadany@aucegypt.edu
AhmedEssam78@aucegypt.edu