Lab: Exploring Transformers for Natural Language Processing and Vision Tasks

Level: M2 (Master's 2)

Duration: 4 hours

Objective

In this lab you will:

1. Understand the core architecture of the Transformer model (attention mechanism, positional encoding, encoder-decoder structure).
2. Implement a Transformer for a Natural Language Processing (NLP) task such as text classification or machine translation.
3. Apply a Vision Transformer (ViT) for image classification.
4. Reflect on the differences between Transformer architectures in text and vision applications.

Part 1: Theoretical Overview (30 minutes)

try to cover:

- **Core Components of Transformers:**
  - Self-attention mechanism
  - Multi-head attention
  - Positional encoding
  - Feedforward layers
- **Variants of Transformers:**
  - BERT (Bidirectional Encoder Representations from Transformers) for NLP.
  - Vision Transformers (ViT) for images.

Part 2: Implementing a Transformer for NLP (90 minutes)

Task: Text Classification with Transformers

1. **Dataset:** Use a public dataset like IMDb for sentiment analysis (positive/negative reviews).
2. **Steps:**
   - Preprocess the text (tokenization using Hugging Face `transformers` library).
   - Load a pretrained Transformer model like `bert-base-uncased` from Hugging Face.
   - Fine-tune the model for text classification.
3. **Code Template:**

```
from transformers import BertTokenizer, TFBertForSequenceClassification
from tensorflow.keras.optimizers import Adam
from sklearn.model_selection import train_test_split
import tensorflow as tf

# Load dataset (example using IMDb)
(x_train, y_train), (x_test, y_test) =
tf.keras.datasets.imdb.load_data(num_words=10000)

# Preprocess data
tokenizer = BertTokenizer.from_pretrained("bert-base-uncased")
x_train = tokenizer([" ".join(map(str, review)) for review in x_train],
padding=True, truncation=True, return_tensors="tf")
x_test = tokenizer([" ".join(map(str, review)) for review in x_test],
padding=True, truncation=True, return_tensors="tf")

# Load pretrained model
model = TFBertForSequenceClassification.from_pretrained("bert-base-uncased",
num_labels=2)

# Compile model
optimizer = Adam(learning_rate=5e-5)
model.compile(optimizer=optimizer, loss=model.compute_loss,
metrics=["accuracy"])

# Train model
model.fit(x_train, y_train, epochs=3, batch_size=32, validation_data=(x_test,
y_test))
```

4. **Expected Outcome:**
   - Fine-tuned model should classify reviews as positive or negative with reasonable accuracy.

Part 3: Applying Vision Transformers (ViT) (90 minutes)

Task: Image Classification with Vision Transformers

1. **Dataset:** Use a dataset like CIFAR-10 for image classification.
2. **Steps:**
   - Preprocess the images into patches.
   - Use a pretrained ViT model like `vit-base-patch16-224` from Hugging Face.
   - Fine-tune the model for classification.
3. **Code Template:**

```
from transformers import ViTFeatureExtractor,
TFAutoModelForImageClassification
from tensorflow.keras.optimizers import Adam
import tensorflow as tf
import numpy as np

# Load CIFAR-10 dataset
(x_train, y_train), (x_test, y_test) = tf.keras.datasets.cifar10.load_data()
x_train, x_test = x_train / 255.0, x_test / 255.0

# Preprocess images
feature_extractor = ViTFeatureExtractor.from_pretrained("google/vit-base-
patch16-224")
x_train = feature_extractor(x_train, return_tensors="tf")["pixel_values"]
x_test = feature_extractor(x_test, return_tensors="tf")["pixel_values"]

# Load pretrained ViT model
model = TFAutoModelForImageClassification.from_pretrained("google/vit-base-
patch16-224", num_labels=10)

# Compile model
optimizer = Adam(learning_rate=5e-5)
model.compile(optimizer=optimizer, loss="sparse_categorical_crossentropy",
metrics=["accuracy"])

# Train model
model.fit(x_train, y_train, epochs=3, batch_size=32, validation_data=(x_test,
y_test))
```

4. **Expected Outcome:**
   - The model should classify images into one of the 10 CIFAR-10 categories.

Part 4: Reflection and Discussion (30 minutes)

Discussion Points:

1. What are the differences in how Transformers process text versus images?
2. How does the self-attention mechanism adapt to different data modalities?
3. What are the limitations of Transformers, and how can they be mitigated?

Deliverables

1. Fine-tuned models for text classification and image classification.
2. Visualizations of results (e.g., classification accuracy, confusion matrix).
3. Answers to the reflection questions.