ChatGPT - Aquest

The error you're encountering happens because the `eval_accuracy` metric is not being computed during evaluation, so the `Trainer` is unable to track the best model based on this metric. The default evaluation metric for Hugging Face models is **loss**, not accuracy, unless you specify it.

Solution:

1. **Add an accuracy metric to evaluation**: We need to modify the training configuration so that **accuracy** is computed during evaluation.
2. **Use `compute_metrics`**: This is a function that computes custom metrics, including accuracy.

Here's how to fix this issue:

1. Define a `compute_metrics` function to compute accuracy during evaluation.
2. Pass this function to the `Trainer`.

Updated Code with `compute_metrics`:

```python
from sklearn.metrics import accuracy_score
from transformers import Trainer, TrainingArguments, EarlyStoppingCallback

# Define the function to compute accuracy during evaluation
def compute_metrics(p):
    predictions, labels = p
    predictions = predictions.argmax(axis=-1)  # Convert logits to predicted labels
    accuracy = accuracy_score(labels, predictions)
    return {'eval_accuracy': accuracy}

# Update TrainingArguments with eval_strategy (since evaluation_strategy is deprecated)
training_args = TrainingArguments(
    output_dir='./results',          # output directory
    num_train_epochs=3,              # number of training epochs
    per_device_train_batch_size=8,   # batch size for training
    per_device_eval_batch_size=16,   # batch size for evaluation
    warmup_steps=500,                # number of warmup steps for learning rate scheduler
    weight_decay=0.01,               # strength of weight decay
    logging_dir='./logs',            # directory for storing logs
    logging_steps=10,
    eval_strategy="epoch",           # evaluation strategy to adopt during training (fixing depreca
    save_strategy="epoch",           # save model at the end of each epoch
    load_best_model_at_end=True,     # load the best model when training finishes
    metric_for_best_model="eval_accuracy",  # monitor accuracy for early stopping
    greater_is_better=True,          # we want higher accuracy
    learning_rate=5e-5,              # learning rate
)

# Early stopping callback
early_stopping_callback = EarlyStoppingCallback(early_stopping_patience=2)

# Initialize the Trainer with the compute_metrics function
trainer = Trainer(
    model=model,                         # the pre-trained model
    args=training_args,                  # training arguments
    train_dataset=train_dataset,         # training dataset
    eval_dataset=val_dataset,            # evaluation dataset
    callbacks=[early_stopping_callback],  # early stopping callback
    compute_metrics=compute_metrics      # add compute_metrics function for accuracy
)
```

Printed using ChatGPT to PDF, powered by PDFCrowd HTML to PDF API.

```
# Train the model
trainer.train()

# Evaluate the model
trainer.evaluate()

# Save the trained model
trainer.save_model("sentiment_analysis_bert_model")
```

Explanation of Changes:

1. **`compute_metrics` function**:
    - This function takes the model's predictions and labels, computes the accuracy, and returns it as a dictionary with the key `'eval_accuracy'`. This allows the `Trainer` to track accuracy during training and evaluation.
2. **Updated `TrainingArguments`**:
    - `eval_strategy="epoch"` is updated to `eval_strategy="epoch"` as per the deprecation notice.
    - `metric_for_best_model="eval_accuracy"` specifies that we want to track the accuracy metric to determine the best model.
    - `greater_is_better=True` tells the `Trainer` that we want higher accuracy values.
3. **`EarlyStoppingCallback`**:
    - It stops training if the accuracy does not improve for `early_stopping_patience=2` epochs.

Key Points:

- **Accuracy Computation**: The `compute_metrics` function is necessary to compute accuracy during evaluation since the default evaluation metric is loss.
- **Best Model Selection**: The `metric_for_best_model` is set to `eval_accuracy` so the model can save and load the best version based on accuracy.

This should resolve the `KeyError` related to missing `eval_accuracy` and properly track the accuracy during training.

Let me know if it works or if you need any further adjustments!