Bonusabgabe Anwendung der KI WS 24/25

Autoren:

David Wolf
ubgza
2462699

Finn Liesching
ugsna
2485390

In der .ZIP befindet sich eine README.md in der die Ausführung des Codes erläutert wird.

Als Basis unseres Modells verwenden wir das vortrainierte „base-uncased“ Modell aus der Reihe der BERT-Modelle. „Base“ bezieht sich dabei auf die Modellgröße, die mit ca. 110 Millionen Parametern moderat gehalten ist, wobei „uncased“ angibt, dass es sich um ein Modell handelt, das nicht zwischen Groß- und Kleinschreibung unterscheidet. Diese Unterscheidung ist in deinem Fall auch nicht notwendig, da wir im Preprocessing-Schritt alle Zeichen in „lowercase“ umwandeln. Zudem entfernen wir Links, Zahlen, Hashtags, HTML-Tags, Mentions, Zeichensetzung sowie URLs per Regex, da diese keinen Einfluss auf die Klassifizierung der Tweets haben. Der Datensatz wird anschließend in ca. 20% Validierungs-Daten und 80% Trainings-Daten unterteilt.

Beim Trainieren des Modells werden die Daten zunächst tokenisiert. Hierbei verwenden wir ebenfalls den Tokenizer des „bert-base-uncased“ Modells. Die Output-Layer definieren wir mit einer Größe von drei, um das Modell auf die Klassifizierung in die drei gewünschten Klassen 0, 1 und 2 anzupassen. Die Anzahl der Trainingsdurchläufe beträgt zwei, wobei wir die Daten in Batches der Größe 32 aufteilen und 10% der Daten zum Warmup verwendet werden. Nach 500 Steps wird anhand der Validierungs-Daten die Modell-Performance evaluiert. Hierbei ist vor allem die „Accuracy“ interessant. Um die Laufzeit des Trainingsvorgangs zu optimieren und die Memory-Auslastung im Rahmen zu halten, ist die Inputlänge auf 32 begrenzt. Zudem verwendet das Modell bei Möglichkeit die GPU in Form von CUDA bzw. MPS.

Nach durchgeführtem Training wird das angepasste Modell im Verzeichnis /saved_model gespeichert. Durch Ausführen der predict.py-Datei wird die test_with_label.csv im Verzeichnis /results generiert. Das Preprocessing ist hierbei identisch zum Preprocessing der Trainings-Daten.