4/14/2024

Assignment Report

Analysis of DistilBERT, albert-base-v2, and FLAN-T5 for Multi-Label Text Classification

Patil, Praful Venkatesh
PVP22001

Abstract:

This report compares how well two different language models, bert-base-uncased and albert-base-v2, did on a task where each piece of text could have more than one label. We also tried out FLAN-T5, a new kind of model, on the same task. Our goal is to clearly explain how well each model worked, point out any interesting findings, and draw conclusions based on what we learned from each experiment.

Models selected for this study:

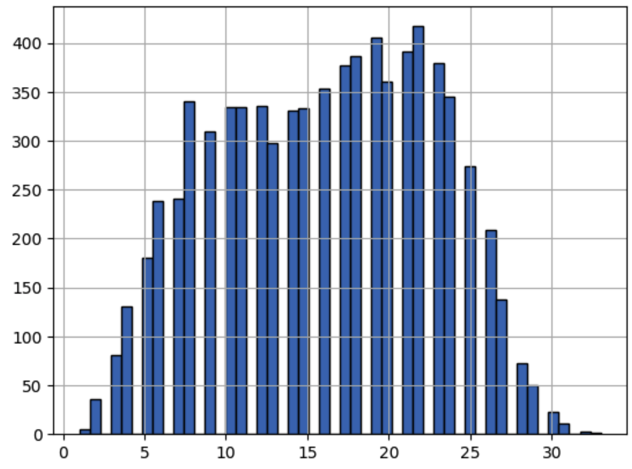
- bert-base-uncased
- albert-base-v2
- google/flan-t5-base

Methodology Implemented:

- Environment Setup:
  Installs necessary Python libraries such as `torchtext`, `transformers`, `evaluate`, `wandb`, and `datasets`.
- Data Loading:
  Loads a dataset from a CSV file into a pandas Data Frame.
  Analyses the distribution of word lengths in the dataset using a histogram.
- Data Splitting:
  Splits the data into training, validation, and test sets using the `train_test_split` method from scikitlearn.
- Dataset Conversion:
  Converts the split data into Hugging Face `Dataset` objects for compatibility with the Transformers library.
- Tokenization:
  Initializes a tokenizer for a specified pre-trained model checkpoint.
  Applies the tokenizer to the dataset to convert text into a format suitable for model input.
- Model Configuration:
  Loads a pretrained model for sequence classification with a specified number of labels.
  Sets the model's configuration for multi-label classification and updates label mappings.
- Training Setup:
  Configures training parameters such as epochs, batch size, and learning rate.

- Model Training:
  Trains the model using Weights & Biases for logging and tracks the best model checkpoint.
- Validation set Evaluation
- Test set Evaluation
- Load the best model checkpoint.
- Evaluates the model on the test dataset using the `Trainer` and logs results.
- Inference:
  Performs inference with the trained model on new text input.
  Processes the output to determine the final predicted labels based on a threshold.

Distribution of word length per tweet:



The dataset's word distribution shows most tweets are succinct, mainly ranging from 10 to 20 words, peaking at 15 words. This briefness aligns with platform limits and fits within the token capacities of BERT, ALBERT, and T5 models, allowing full tweet analysis without data loss. The dataset's occasional longer tweets are outliers, not the standard trend.

Experiment 1: bert-base-uncased

The 'bert-base-uncased' model is a foundational NLP model by Google with 12 layers and 110 million parameters. It handles text without case sensitivity and is pre-trained on extensive English data, capturing complex language contexts effectively.

Training Results:

- Loss: 0.3218
- Accuracy: 87.03%
- F1 Score: 0.4540
- Precision: 0.5218
- Recall: 0.4240

- Hamming Loss: 0.1296

Testing Results:

- Loss: 0.3264
- Accuracy: 86.72%
- F1 Score: 0.4556
- Precision: 0.5190
- Recall: 0.4300
- Hamming Loss: 0.1327

The 'bert-base-uncased' model showed robust performance in both training and testing phases, indicating its strong generalization capabilities for text classification tasks. Its consistent metrics across both phases suggest it's reliable and can maintain performance on unseen data.

Link to WandB for bert-base uncased: https://api.wandb.ai/links/utd659/mqr8en8m

Experiment 2: Albert-base-v2

Albert-base-v2' is an optimized version of BERT for better efficiency. It maintains the structure of BERT but uses fewer parameters, focusing on speed and inter-sentence coherence, beneficial for understanding sentence relationships with improved resource efficiency.

Training Results:

- Loss: 0.3544
- Accuracy: 85.31%
- F1 Score: 0.3579
- Precision: 0.5522
- Recall: 0.3335
- Hamming Loss: 0.1468

Testing Results:

- Loss: 0.3574
- Accuracy: 85.10%
- F1 Score: 0.3601
- Precision: 0.4885
- Recall: 0.3356
- Hamming Loss: 0.1489

The 'albert-base-v2' model demonstrated a slightly lower performance compared to 'bert-base-uncased', with modest accuracy in both training and testing. It seems to be slightly less precise in its predictions, suggesting a potential area for improvement in model tuning or data pre-processing.

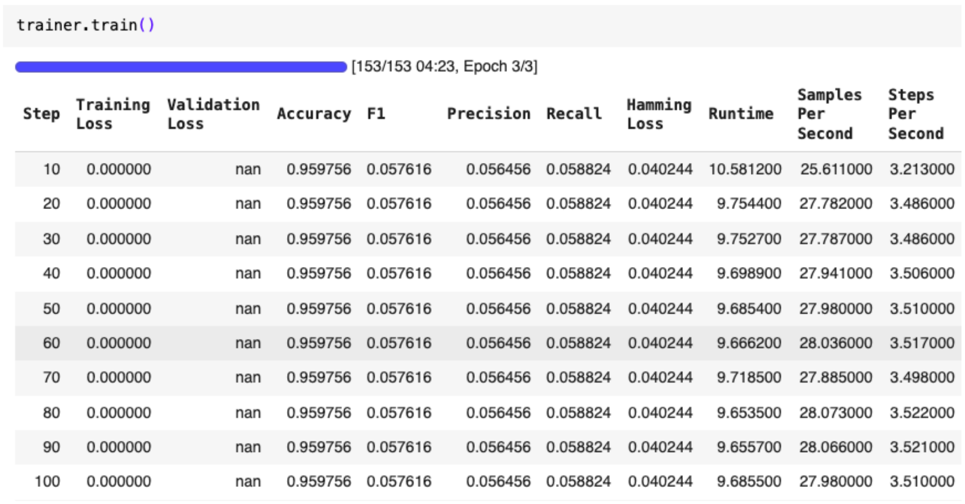
Link to WandB for albert-base-v2: https://api.wandb.ai/links/utd659/99finvyn

Experiment 3: google/flan-t5-base

The `google/flan-t5-base` model, a variant of the T5 architecture, is specifically fine-tuned to comprehend and execute instructions, making it adept for a broad spectrum of tasks framed as text-to-text problems. This includes classification tasks where its ability to accurately categorize text is leveraged, demonstrating its flexibility and efficiency in handling complex natural language processing challenges.

Challenges faced when implementing this model for multilabel classification:

- **Memory Error Issues:** Even with Colab Pro and access to high-end GPUs like A100 or V100, memory errors were persistent, indicating a demand for resources beyond what the available hardware could provide.
- **Sample Size and Batch Adjustments:** To mitigate memory issues, the sample size was reduced, and `per_device_train_batch_size` was adjusted to 8. Additionally, `eval_steps` were changed to 10 to manage resource constraints.
- **Inconsistent Loss Metrics:** Despite adjustments to batch size and evaluation steps, the training process exhibited anomalies, with train and validation loss metrics not being reported correctly, resulting in a lack of variability in accuracy across different evaluation steps.



```
trainer.train()
```

```
[153/153 04:23, Epoch 3/3]
```

Step	Training Loss	Validation Loss	Accuracy	F1	Precision	Recall	Hamming Loss	Runtime	Samples Per Second	Steps Per Second
10	0.000000	nan	0.959756	0.057616	0.056456	0.058824	0.040244	10.581200	25.611000	3.213000
20	0.000000	nan	0.959756	0.057616	0.056456	0.058824	0.040244	9.754400	27.782000	3.486000
30	0.000000	nan	0.959756	0.057616	0.056456	0.058824	0.040244	9.752700	27.787000	3.486000
40	0.000000	nan	0.959756	0.057616	0.056456	0.058824	0.040244	9.698900	27.941000	3.506000
50	0.000000	nan	0.959756	0.057616	0.056456	0.058824	0.040244	9.685400	27.980000	3.510000
60	0.000000	nan	0.959756	0.057616	0.056456	0.058824	0.040244	9.666200	28.036000	3.517000
70	0.000000	nan	0.959756	0.057616	0.056456	0.058824	0.040244	9.718500	27.885000	3.498000
80	0.000000	nan	0.959756	0.057616	0.056456	0.058824	0.040244	9.653500	28.073000	3.522000
90	0.000000	nan	0.959756	0.057616	0.056456	0.058824	0.040244	9.655700	28.066000	3.521000
100	0.000000	nan	0.959756	0.057616	0.056456	0.058824	0.040244	9.685500	27.980000	3.510000

- **Uniform Accuracy Readings:** A consistently high accuracy of approximately 95% was reported at every evaluation step, which did not align with expectations for variability during training, suggesting potential issues with the training dynamics or data representation.
- **'Nan' Logits Output:** Encountered an issue where the logits output from the T5 model were consistently 'nan', which indicates a possible issue with the model

configuration or the training process that could not be resolved with default pretrained settings.

- **Difficulty in Configuration Adjustments:** Struggled with making effective configuration adjustments due to limitations in understanding or access to the model's internal settings, impeding the resolution of the 'nan' outputs issue.

Link to WandB for google/flan T5: https://api.wandb.ai/links/utd659/74ikn8u8

Conclusion:

- BERT-base-uncased showed consistent performance, indicating strong generalization on unseen data.
- ALBERT-base-v2, while efficient, demonstrated slightly less precision, suggesting further tuning could be beneficial.
- Challenges with google/flan-t5-base included persistent memory errors despite using high-end GPUs like A100 or V100.
- Reducing the sample size and adjusting batch sizes were necessary steps to manage resource limitations for flan-t5-base.
- The flan-t5-base model outputted 'nan' values for logits consistently, indicating potential issues with the training or data representation.
- The training and validation loss metrics for flan-t5-base were not accurately reported, causing a lack of expected variability in the model's accuracy across evaluations.

The following link provides Weights & Biases (WandB) dashboard where metrics of all three models are compared: https://api.wandb.ai/links/utd659/0ecpi4ws