Fine-Tuning BERT for Disaster Tweet Classification

Dataset Overview

The dataset contains labeled tweets used to identify whether a tweet refers to a real disaster:

- **id**: Unique identifier for each tweet
- **keyword**: A keyword from the tweet (may be missing)
- **location**: Location where the tweet originated (may be missing)
- **text**: The tweet content
- **target**: Binary label – `1` if the tweet indicates a disaster, `0` otherwise

Dataset size: **10,000 labeled tweets**

Objective

To fine-tune transformer-based models to classify tweets as either disaster-related or not, using two different approaches:

1. Supervised fine-tuning with KerasNLP (DistilBERT)
2. PEFT (Parameter-Efficient Fine-Tuning) techniques

1. Supervised Fine-Tuning using KerasNLP (DistilBERT)

Model:

- **Architecture**: DistilBERT (a lightweight version of BERT)
- **Framework**: KerasNLP
- **Training Strategy**: Full fine-tuning on all layers using labeled data.

Details:

- Pre-trained DistilBERT model loaded via KerasNLP.
- The model was fine-tuned on the dataset using supervised learning.
- Tokenizer: keras_nlp.models.DistilBertTokenizer
- Optimizer: AdamW
- Loss: Binary Crossentropy
- Metrics: Accuracy

Result:

- **Validation Accuracy**: **0.81**

2. Parameter-Efficient Fine-Tuning (PEFT)

Model:

- **Base Model**: DistilBERT (same as above)
- **PEFT Type**: Adapter-based or LoRA (Low-Rank Adaptation)

- **Framework**: Hugging Face Transformers + PEFT library

Details:

- Instead of updating all parameters, only a small subset (adapters or LoRA layers) are trained.
- This significantly reduces training time and computational resource usage while retaining accuracy.
- Tokenization and preprocessing are consistent with the first method.

Result:

- **Validation Accuracy**: **0.82**

Comparison of Results

Fine-Tuning Method	**Accuracy**
Supervised(KerasNLP)	0.81
PEFT	0.82

Conclusion

- Both techniques performed well on disaster tweet classification.
- **PEFT** slightly outperformed full supervised fine-tuning in terms of accuracy while being more parameter-efficient and faster to train.
- PEFT is therefore a favorable choice for low-resource or production settings requiring fast iteration and deployment.