ML Project Report: Predicting News Genre using Transformers and KNN

Dataset:

This project uses the **AG News Subset**, a widely used benchmark dataset for text classification tasks. It is publicly available on the TensorFlow Datasets GitHub repository.

The AG News dataset contains **short news articles** categorized into four distinct genres, making it well-suited for a multi-class classification problem. It consists of two separate CSV files—one for training and one for testing.

Dataset Statistics

- **Training Set**: 119,999 samples
- **Testing Set**: 7,599 samples
- **Number of Classes**: 4

Each data sample is structured with the following columns:

1. **label**: This is the “target” variable for the model and represents news genre in numerical form. It contains values ranging from 0 to 3.
2. **headline**: Contains the headline of a news article.
3. **description**: contains a short description of the news article.
4. **news_genre**: contains four news genres “World”, “Sports”, “Business”, “Sci/Tech”

Following is the mapping of the “news_genre” column to the “label” column. This is important because the output of a model’s prediction will be a numerical value.

Label	Genre
0	World
1	Sports
2	Business
3	Sci/Tech

Due to **limited GPU resources**, only a subset of the data was used for training. Specifically, **20,000 examples** were randomly selected from the original training dataset to reduce computational requirements while still maintaining a meaningful sample size for learning.

To evaluate training loss over iterations and hyperparameter tuning for DistilBERT, **15% of this subset** (i.e., 3,000 examples) was split into a **validation set** using the `train_test_split` function from the `sklearn.model_selection` module. The remaining 17,000 samples were used for training the DistilBERT model. The **testing set was unused** during the training phase and is only used while testing KNN accuracy.

Design of Experiment:

The goal of this project was to accurately **predict the genre of news articles** using both the *headline* and a *short description* as inputs. The core idea was to leverage the power of deep learning-based **feature extraction** to generate meaningful semantic representations of the textual data, which could then be used with a traditional machine learning classifier for final prediction.

Phase 1: Feature Extraction Using Transfer Learning

For the first stage of modeling, I selected **DistilBERT**, a transformer-based model that serves as a compact yet powerful alternative to the original BERT architecture. DistilBERT retains 97% of BERT's language understanding capabilities while being significantly smaller and faster. It was chosen due to its balance of performance and efficiency.

DistilBERT is **pre-trained** on large corpora of English text using masked language modeling, enabling it to learn rich linguistic patterns and contextual information. In this project, I **fine-tuned**

the model on the news genre classification task by adding a task-specific output layer. This layer adapts the model to the nuances of categorizing news headlines and descriptions into discrete genres.

The training process involves adjusting the weights of the model based on the labeled dataset so that it can learn to associate textual inputs with genre labels. Once trained, the model outputs **feature embeddings** from its final hidden layer. These embeddings are high-dimensional vector representations that encode the semantic content of the input texts. In this project, we primarily used the **[CLS] token embedding** to represent the entire input sequence.

Phase 2: Classification Using K-Nearest Neighbors (KNN)

After extracting features from the fine-tuned DistilBERT model, I used a **K-Nearest Neighbors (KNN)** classifier as the second stage of the pipeline.

The motivation for using KNN lies in its simplicity and interpretability, especially in the context of semantically rich vector representations. Since the embeddings encode contextual meaning, instances that are close in this feature space are likely to belong to the same genre.

The feature extraction process was applied to both the training and testing datasets to ensure that they lie in the same embedding space. The `extract_features` function handled this task by running the input texts through the trained DistilBERT model and collecting the [CLS] token output.

Modeling Decisions & Variations

To explore the model's performance across configurations, I experimented with two major variables:

1. **Cased vs. Uncased DistilBERT Models**
    - The **cased** version preserves capitalization, which can provide additional syntactic and semantic cues, especially in named entities or proper nouns.
    - The **uncased** version converts all text to lowercase, reducing vocabulary size and potential noise from capitalization inconsistency.
2. **Variation in KNN Hyperparameter K**

- The number of neighbors (K) in KNN plays a significant role in balancing bias and variance.
- I evaluated K = 3, 5, and 7 for both cased and uncased DistilBERT to identify the most robust configuration.

By conducting these experiments, I was able to observe how **both language model choice and KNN configuration** influenced the overall classification accuracy. This two-stage architecture (DistilBERT + KNN) allowed me to blend the power of deep semantic understanding with a straightforward and interpretable classification mechanism.

Results and Interpretation:

This project involved training a DistilBERT model and using its learned semantic representations (i.e., [CLS] token embeddings) as feature vectors for a downstream classification task using the K-Nearest Neighbors (KNN) algorithm. Several modeling decisions were tested to evaluate the impact of different model variations and hyperparameters on final accuracy. Below is a breakdown of the configurations and performance outcomes for each modeling decision.

Modeling Decision 1:

Models: DistilBERT *cased* + KNN (K = 5)
Training Time: [3188/3188 steps, 18:04 mins, Epoch 1/1]
Training Loss Progression:

- Step 500: 0.3523
- Step 1000: 0.2529
- Step 1500: 0.2219
- Step 2000: 0.1920
- Step 2500: 0.1889
- Step 3000: 0.1814

KNN Accuracy: **94.21%**

This configuration demonstrated excellent convergence in loss and produced one of the highest classification accuracies, showcasing that the model successfully captured meaningful patterns in the data.

Modeling Decision 2:

Models: DistilBERT *cased* + KNN (K = 3)
KNN Accuracy: **93.87%**

Using a smaller value of K slightly reduced the classification performance. This is expected, as smaller K values are often more sensitive to noise and can lead to overfitting on localized data points.

Modeling Decision 3:

Models: DistilBERT *cased* + KNN (K = 7)
KNN Accuracy: **94.22%**

With a larger K, the model maintained a high level of accuracy, slightly improving over K=3 and K=5. This suggests that a moderately larger neighborhood size may provide better generalization on this dataset.

Modeling Decision 4:

Models: DistilBERT *uncased* + KNN (K = 5)
KNN Accuracy: **93.95%**

Switching from a cased to an uncased variant of DistilBERT with the same K value resulted in a minor drop in performance. This suggests that in this particular dataset, case sensitivity may slightly improve representation learning.

Modeling Decision 5:

Models: DistilBERT *uncased* + KNN (K = 3)
Training Time: [3188/3188 steps, 16:59 mins, Epoch 1/1]
Training Loss Progression:

- Step 500: 0.3354
- Step 1000: 0.2397
- Step 1500: 0.2100
- Step 2000: 0.1856
- Step 2500: 0.1805
- Step 3000: 0.1750

KNN Accuracy: **94.17%**

This model maintained strong accuracy, slightly outperforming its cased counterpart at K=3. The consistent downward trend in training loss once again confirms effective learning during fine-tuning.

Modeling Decision 6:

Models: DistilBERT *uncased* + KNN (K = 7)
KNN Accuracy: **94.49%**

This configuration yielded the highest KNN accuracy across all modeling decisions, indicating that the uncased variant with a broader neighborhood (K=7) captured the underlying structure of the data most effectively.

Interpretation & Insights

The results collectively demonstrate that **DistilBERT performs exceptionally well on this dataset**, whether using cased or uncased versions. The **consistently decreasing training loss** indicates successful learning of semantic representations during the fine-tuning process. Importantly, **KNN proved to be an effective classifier**, suggesting that the learned [CLS] token embeddings capture class-separable information that is amenable to distance-based classification.

One of the most critical steps in this pipeline is **feature extraction**, accomplished via the `extract_features` function. This function utilizes the pretrained DistilBERT model to extract the [CLS] token from the final hidden layer for each input. These [CLS] outputs represent the entire input sequence and serve as condensed feature vectors. By extracting embeddings for

both training and test datasets, we ensure that all data points lie in the same semantic vector space — a prerequisite for valid and meaningful distance measurements in KNN classification.