DistilBERT-Emotion-v4

Model Overview

Model: TFDistilBertForSequenceClassification
Architecture: DistilBERT (base, uncased)
Language: English
Classes: anger, disgust, fear, happiness, neutral, sadness, surprise
Params: ~66M
Framework: TensorFlow + Hugging Face Transformers
License: MIT

DistilBERT-Emotion-v4 is a fine-tuned transformer model designed to classify English text into one of seven emotion categories: *anger*, *disgust*, *fear*, *happiness*, *neutral*, *sadness*, and *surprise*.

It is built on top of the lightweight yet powerful distilbert-base-uncased, which is a model that combines the speed of DistilBERT and custom fine-tuning for emotion recognition tasks

Model Architecture:

This model is a **DistilBERT** encoder with a classification head. It uses 6 transformer layers, 12 attention heads, and a hidden size of 768. We used the TFDistilBertForSequenceClassification implementation from Hugging Face's Transformers library. The classification head is a dense layer with softmax activation with our with our number of output units matching the number of emotion classes in the dataset.

Data augmentation was used during the preprocessing step using synonym replacement from 'WordNet" to help increase the model's performance. The model uses a distilbert-base-uncased tokenizer, which lowercases and WordPiece-tokenizes the input.

Training:

A curated dataset (dataset_v2.csv) and a translated text dataset (group 11_url1.csv) were used to train the model. Text preprocessing included basic cleaning, unifying emotion label categories, and eliminating empty or wrong entries. WordNet was used to apply data augmentation with synonym replacement, changing specific words in training examples to enhance generalization. The model was trained for 3 epochs using the default batch size and the DistilBERT tokenizer.

Purpose:

The model is created to classify emotions based on text using tone analysis, this is useful in contexts of social media analysis, customer feedback and chatbots or conversational AI.

Its goals are:

- To predict emotional content in short and long English text.
- To offer a deployable model for fast inference.
- To contribute to emotion-aware NLP applications with an explainable and customizable base.

Development Context:

Key Assumptions and Constraints

One of the key assumptions during the development of this emotion prediction model was that training data would be representative of the real-world variety of emotional expressions in social media videos. However, a significant constraint arose from the contextual gap between the training and testing data. The model was trained on data that did not fully align with the specific slang terms and terms found in youtube videos and series. This mismatch contributed to suboptimal performance, particularly in distinguishing emotions like fear, which were often misclassified as neutral. The contextual gap between the data used for training and the testing data remains one of the biggest challenges in improving model accuracy.

Training Training Resources and conditions

The model was trained on an external server, leveraging GPU resources to accelerate the process. Training time amounted to approximately ~40 minutes for 7 epochs. Early stopping was implemented to prevent overfitting and ensure the model did not perform excessively well on the training set at the cost of generalizing poorly to unseen data. The constraints on training time and resources were balanced with the need for sufficient computation power to handle the complexity of the transformer-based architecture.

Intended Use:

Social Media Analysis*:*

e.g. the model can identify the emotions in text written by users on Twitter/Facebook/Instagram. This means that by interpreting posts, comments, or messages, it helps in catching the emotional sentiment behind the posts created by customers, and so brands' could modify their published content and marketing strategies according to how the customer feels.

Customer Feedback*:*

Emotion recognition in customer reviews or surveys can help businesses better understand customer satisfaction, and the emotional drivers behind feedback, which can help in product improvements and customer service strategies.

Chatbots & Conversational AI*:*

The model can enhance the emotional intelligence of chatbots by enabling them gain the ability to detect the emotional state of users, this allows for more empathetic and appropriate responses in customer support, virtual assistants, or mental health applications.

***Content Moderation**:*

The model can help in detecting hateful or charged language in user-generated content, helping to identify and manage potentially harmful or inappropriate content.

Limitations

1. ***Contextual Understanding Limitations:***
   The model may misinterpret nuanced language, sarcasm, or slang, especially in informal contexts like social media or video transcripts since the dataset has a mix of data sources.
2. ***Cultural and Language Bias:***
   The model is trained on English text and may not be effective for texts in other languages or cultures, potentially leading to inaccurate emotion detection.
3. ***Misclassification of Emotions:***
   Emotions like fear or surprise can be misclassified, especially in short texts with little context.
4. ***Complex Emotional States:***
   the mode could struggle to detect mixed or complex emotions, such as ambivalence or irony.

By correctly classifying emotions in text, the model enhances content strategies and audience engagement, improves content personalization, and tracks emotional trends by giving valuable insights for marketing.

Dataset Details:

Feature	**Description**
text	Raw user input text.

Feature	**Description**
dominant_emotion	Emotion label associated with the input text (e.g., happiness, sadness, etc.).
POS_Tags	Part-of-speech tags extracted from the text.
TF_IDF	Term Frequency-Inverse Document Frequency representation of the text.
Sentiment_Score	Numerical score representing the sentiment (positive, negative, neutral) of the text.
Pretrained_Embeddings	Pre-trained word embeddings from external models.
Custom_Embeddings	Domain-specific embeddings created for the task.
Cleaned_Text	Processed and normalized version of the original text.

Additional Dataset Info:

- **Total Rows**: 117,260
- **Primary Task**: Emotion classification from user-generated text
- **Emotion Labels**: Include a range such as *happiness, sadness, anger, fear, surprise*, and others

Performance Metrics and Evaluation:

The model achieves an accuracy of **~71.76% on the test set.**

Emotion	Precision	Recall	F1-Score	Support
anger	0.00	0.00	0.00	21
disgust	0.00	0.00	0.00	0
fear	0.00	0.00	0.00	47
happiness	1.00	0.01	0.02	115
neutral	0.75	0.94	0.83	694
sadness	0.00	0.00	0.00	10
surprise	0.37	0.44	0.40	55
Accuracy			0.72	942
Macro avg	0.30	0.20	0.18	942
Weighted avg	0.70	0.72	0.64	942

For a detailed breakdown of model misclassifications and insights, please refer to the Error Analysis Report

Explainability and Transparency:

What This Code Does

This code helps us understand which words in a sentence are most important for a model's prediction. It uses a method called **Improved Gradient × Input with Layer-wise Relevance Propagation (LRP)**.

1. Figuring Out Important Words (gradient_x_input)

- The code looks at how changing each word would affect the model's prediction.
- It gives a score to each word to show how much it helped the model make a decision.

- If the setup isn't working right, it will give an error to let you know.

2. Keeping Relevance Flow Stable (Modified Attention & Layer Norm)

- Transformers (like BERT) are tricky because they use attention and normalization.
- The code changes how attention and normalization work so the importance (relevance) flows through the model more clearly and correctly.
- This makes the explanations more trustworthy.

3. Testing Word Importance (perturb_input_and_evaluate)

- The code removes or changes words in the input and sees how the model's confidence changes.
- It can:
  - Remove less important words first.
  - Or remove most important words first.
- Words are replaced with padding ([PAD]) to keep the sentence length the same.
- If the model loses confidence when a word is removed, that word was probably important!

Showing the Results

- Bar Graph: Shows how important each word is — taller bars mean more important.
- Attention Heatmap: Shows which words are paying attention to each other — helps us see how the model understands the sentence.
- Confidence Line Plot: Shows how the model's confidence changes as words are removed.
  - A red line at 0.5 shows when the model starts to get unsure.
  - Shaded red area shows when confidence drops a lot.

What's Special About "Improved LRP"?

- Normal LRP doesn't work well in transformers because they're complex.

- This improved version fixes that by handling attention and normalization better.
- The result: more stable and accurate explanations of why the model made its choice.

Recommendation for use:

Practical Guidance:

- Deployment: Ensure computational resources can handle the model's requirements, especially for real-time applications.
- Fine-Tuning: Tailor the model with domain-specific data for improved accuracy (e.g., customer feedback, social media).
- Processing Type: Use batch processing for large datasets and real-time processing for live applications.

Operational Risks:

- Misclassification: The model may misinterpret nuanced emotions, especially with slang or irony.
- Bias: Be cautious of potential biases, as the model may not handle diverse dialects or cultures well.
- Speed: Real-time applications must be optimized for efficiency to prevent delays.

Use Cases:

Media Companies: Analyze audience sentiment, personalize content recommendations, and monitor social media.

Client Stakeholders: Analyze customer feedback, enhance chatbot interactions, and improve content moderation.

Sustainability considerations

Training large transformer models like DistilBERT requires a lot of compute resources, which can result in high energy consumption and environmental impact. Key areas of concern are:

- **Carbon Footprint:**

Training transformer models, especially on large datasets, uses a lot of GPU, which is energy intensive. The environmental cost gets worse if training is done over long periods or with multiple hyperparameter tuning sessions.

Deployment on cloud platforms can also contribute to carbon emissions if the infrastructure isn't energy efficient or uses fossil fuels for power generation.

- **Energy Consumption:**

The energy used to train models like DistilBERT is substantial. Although DistilBERT is a lighter version of BERT, it still requires a lot of computing resources, especially for tasks like emotion classification which involves large datasets and complex model architectures.

GPUs, although faster use more energy compared to CPUs, which contributes to a higher carbon footprint.

Recommendations for Minimizing Environmental Impact:

- **Efficient Hardware:** Using energy-efficient hardware (e.g., newer GPUs or TPUs) for lower energy consumption during training and inference.

- **Model Pruning**: Reducing model size by removing irrelevant parts to lower energy consumption and inference time.

- **Quantization**: Convert models to lower precision to speed up inference and reduce computational cost.

- **Use of Pre-trained Models:** Fine-tuning pre-trained models instead of training from scratch reduces computational resources and training time.

- **Cloud Resource Optimization:** Opt for cloud providers with green energy options and use preemptible instances or early stopping to minimize energy usage during training.

- **Efficient Inference:** Use batch processing and edge deployment to lower operational energy consumption after deployment.