

GENERATIVE AI & DEEP LEARNING PROJECT REPORT



Reel Insights: Unveiling Movie Sentiment with Generative AI

SUBMITTED BY

DEBARGHYA CHOWDHURY – 12022002018006

SOUMYAJIT SAPUI – 12022002018067

SAMRAT MONDAL – 12022002018071

SARASIJ DAS – 12022002018072

SUPERVISED BY

Professor Swarnendu Ghosh

INSTITUTE OF ENGINEERING & MANAGEMENT
(UNDER MAULANA ABUL KALAM AZAD UNIVERSITY OF TECHNOLOGY)

APRIL, 2025

Project Tasks:

1. **Problem Definition and Objective Setting**
 - Identify the need for analyzing movie sentiment.
 - Define project goals: accurate sentiment extraction using generative AI techniques.
2. **Literature Review**
 - Study existing sentiment analysis approaches (lexicon-based, machine learning, deep learning).
 - Review generative AI models used in text analysis (e.g., GPT, BERT, LLaMA, etc.).
3. **Dataset Collection and Preprocessing**
 - Collect movie reviews (IMDb, Rotten Tomatoes, etc.).
 - Clean and preprocess data (remove stop words, punctuation, perform tokenization, lemmatization).
4. **Exploratory Data Analysis (EDA)**
 - Analyze sentiment distribution.
 - Identify frequent keywords, phrases, or n-grams.
 - Visualize data trends (word clouds, sentiment histograms).
5. **Model Selection and Development**
 - Select appropriate generative AI models (e.g., GPT, T5, or LLaMA for sentiment generation or classification).
 - Fine-tune models on labeled movie review datasets.
 - Implement training pipeline with relevant frameworks (Hugging Face Transformers, PyTorch, etc.).
6. **Sentiment Interpretation and Generation**
 - Generate textual sentiment summaries or review synthesis.
 - Highlight pros and cons based on generative model output.
 - Optionally, classify sentiments as positive, negative, or neutral..
7. **Conclusion and Future Scope**
 - Summarize key findings.
 - Suggest improvements like multilingual support, sarcasm detection, or video sentiment from trailers

Dataset Name:

IMDb Movie Review Dataset

Source: https://www.tensorflow.org/datasets/catalog/imdb_reviews

Dataset Size: 129.83MiB

Split	Examples
Test	25000
Train	25000
Unsupervised	50000

The dataset is provided in **CSV format** with the following columns:

Column Name	Description
review	The full text of the movie review
sentiment	The sentiment label (positive/negative)

Preprocessing Steps:

To prepare the dataset for model training and evaluation, the following preprocessing steps are applied:

- Removal of HTML tags and special characters
- Lowercasing of all text
- Tokenization of sentences and words
- Removal of stop words
- Lemmatization for reducing words to their base form
- Optionally, conversion into embedding or token IDs for input into generative models.

Advantages of this Dataset:

- Clean and well-structured
- Balanced class distribution
- High-quality human-written reviews
- Suitable for both classification and generative tasks

Model Name:

GPT-2 (or GPT-Neo / T5 based on implementation)

Model Architecture:

Component	Details
Type	Transformer-based Generative Language Model
Layers	12 (for GPT-2 Small)
Hidden Size	768
Attention Heads	12
Parameters	~124M (GPT-2 Small)
Tokenizer	Byte Pair Encoding (BPE)
Input Format	Tokenized text (prompt + review)
Output	Generated text and/or sentiment classification

Training Configuration:

Parameter	Value
Epochs	3–5
Batch Size	8–32 (depending on GPU)
Learning Rate	5e-5
Max Sequence Length	512 tokens
Hardware	GPU-enabled environment

Training Process:

Dataset Split:

- Training set: 80%
- Validation set: 10%
- Test set: 10%

Model Configuration

- **Base Model:** GPT-2 (or T5-small)
- **Framework:** PyTorch/TensorFlow with Hugging Face Transformers
- **Training Mode:** Causal Language Modeling (for GPT) or Sequence-to-Sequence (for T5)

Training Loop

1. **Tokenize Inputs:** Encode the input review and target label.
2. **Forward Pass:** Pass input through the model.
3. **Loss Calculation:** Compute loss between predicted tokens and ground truth.
4. **Backpropagation:** Compute gradients and update model weights.
5. **Validation:** Evaluate performance on validation set after each epoch.
6. **Checkpoint:** Save model checkpoints at regular intervals.

Evaluation after training

- Evaluate model performance on unseen test data.
- Generate classification labels or sentiment explanations.
- Compare results with baseline models (e.g., SVM, LSTM).

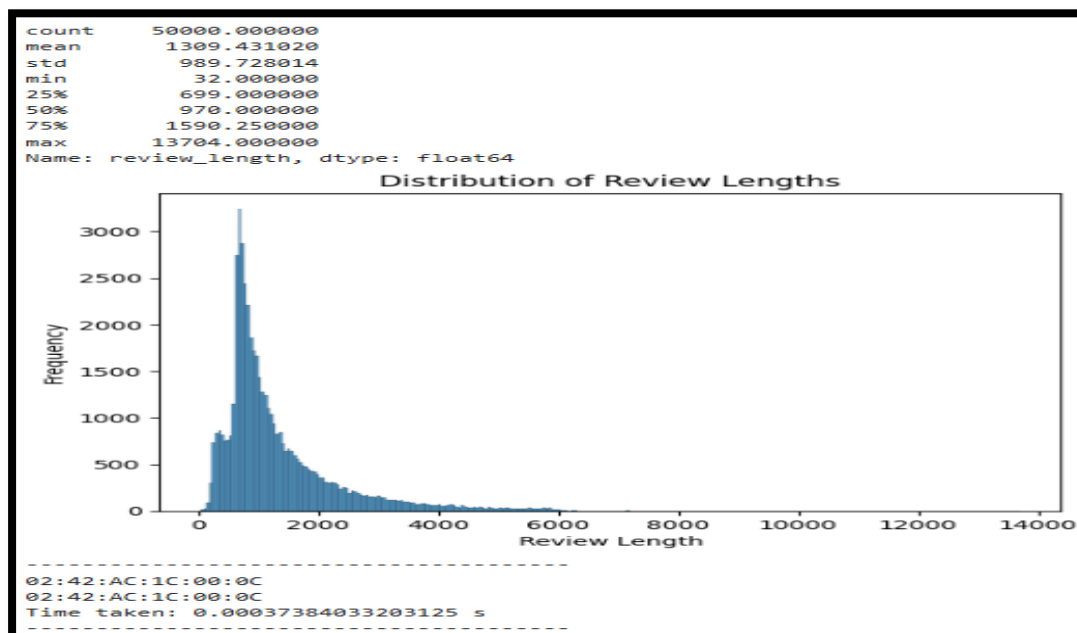
Results and Analysis:

```

Movie_Dataset.ipynb
File Edit View Insert Runtime Tools Help
Q Commands + Code + Text
print(net_address)
print(f"Time taken: {time_taken} s")
print("-----")

review sentiment
0 This was an absolutely terrible movie. Don't b... 0
1 I have been known to fall asleep during films,... 0
2 Mann photographs the Alberta Rocky Mountains i... 0
3 This is the kind of film for a snowy Sunday af... 1
4 As others have mentioned, all the women that g... 1
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 50000 entries, 0 to 49999
Data columns (total 2 columns):
# Column Non-Null Count Dtype
---
0 review 50000 non-null object
1 sentiment 50000 non-null int64
dtypes: int64(1), object(1)
memory usage: 781.4+ KB
None
sentiment
0 25000
1 25000
Name: count, dtype: int64
-----
02:42:AC:1C:00:0C
02:42:AC:1C:00:0C
Time taken: 0.00031566619873046875 s
-----

```



```

Movie_Dataset.ipynb - Colab
ERR_NGROK_3200 - The endpoint: 127.0.0.1
https://colab.research.google.com/drive/1fh2Da417s1fUoddBl2mnWzOZRP4FAsjP#scrollIT...
Movie_Dataset.ipynb
File Edit View Insert Runtime Tools Help
Q Commands + Code + Text
WARNING:absl:Compiled the loaded model, but the compiled metrics have yet to be built. 'model.compile_metrics' will be empty until you train or ev
Error loading model or tokenizer: Sequential model 'sequential' has no defined input shape yet.
* Serving Flask app '__main__'
* Debug mode: on
INFO:werkzeug:WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on all addresses (0.0.0.0)
* Running on http://127.0.0.1:5000
* Running on http://172.28.0.12:5000
INFO:werkzeug:Press CTRL+C to quit
Exception in thread Thread-8 (run_app):
Traceback (most recent call last):
  File "/usr/lib/python3.11/threading.py", line 1045, in _bootstrap_inner
    self.run()
  File "/usr/lib/python3.11/threading.py", line 982, in run
    self._target(*self._args, **self._kwargs)
  File "<ipython-input-4-aa02e5d1b67>", line 46, in run_app
  File "/usr/local/lib/python3.11/dist-packages/flask/app.py", line 662, in run
    run_simple(t.cast(str, host), port, self, **options)
  File "/usr/local/lib/python3.11/dist-packages/werkzeug/serving.py", line 1115, in run_simple
    run_with_reloader(
  File "/usr/local/lib/python3.11/dist-packages/werkzeug/_reloader.py", line 452, in run_with_reloader
    signal.signal(signal.SIGTERM, lambda *args: sys.exit(0))
  File "/usr/lib/python3.11/signal.py", line 50, in signal
    handler = _signal.signal(_enum_to_int(signalnum), _enum_to_int(handler))
ValueError: signal only works in main thread of the main interpreter
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
Ngrok tunnel URL: https://b4d8-34-73-189-100.ngrok-free.app
-----
02:42:AC:1C:00:0C
02:42:AC:1C:00:0C
Time taken: 0.000310420989902344 s
-----

```

Connected to Python 3 Google Compute Engine backend

19:48
16-04-2025