National College of Ireland

Configuration Manual

MSc Research Project
MSc Data Analytics

Ankita Garud
Student ID: 24102563

School of Computing
National College of Ireland

Supervisor: Jorge Basilio

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	**Ankita Dattatray Garud**
Student ID:	**24102563**
Programme:	**MSc Data Analytics** **Year: 2025-26**
Module:	**Research Practicum**
Supervisor:	**Jorge Basilio**
Submission Due Date:	**28th Jan 2026**
Project Title:	**Comparative Analysis of User-Generated Video and Text Reviews of Healthcare Products Using Multimodal NLP**
Word Count:	**2258** **Page Count**: 15

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Ankita Garud

Date: 28th Jan 2026

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Ankita Garud
24102563

1 System Overview and Requirements

1.1 Project Overview:

This configuration manual outlines the environment and procedures needed to execute the code artefact relating to the MSc research project: Comparative Analysis of Multimodal NLP User-Generated Video and Text Review of Healthcare Products.

The code implements:

- Amazon healthcare text review pre-processing.
- Healthcare product review (audio extraction and transcription) downloading and extraction on YouTube.
- Text sentiment analysis using VADER and DistilRoBerta/Roberta based transformer encoder.
- Audio prosodic feature extractor The extraction of audio prosodic features on the YouTube review through Librosa.
- Comparison and platform visualisations and CSVs on space. The code of the project can be viewed in the Jupyter notebook:
- Finally, Final_Thesis.ipynb expects the cleaned up datasets and results to be the following.

1.2 Hardware Requirement:

Component	Minimum Requirement	Actual Environment Used (Project)	Recommended Requirement
Processor (CPU)	4-core CPU (Intel i5 / AMD equivalent)	Google Colab virtual CPU	6–8 core CPU if running locally
Memory (RAM)	12 GB	~12–15 GB RAM available in Google Colab runtime	16 GB+ for local execution
Storage	15 GB free space	Google Colab ephemeral disk (~50–70 GB)	25+ GB if running locally
Graphics (GPU)	Not required (CPU-only supported)	**NVIDIA Tesla T4 GPU (Google Colab)** with 16 GB VRAM	Any CUDA-enabled NVIDIA GPU (≥4 GB VRAM)
Operating System	Windows 10/11, macOS, or Linux	Google Colab Linux (Ubuntu-based environment)	Ubuntu 20.04+ for local
Internet	Required for downloading	Stable broadband used	Same

Connection	models & YouTube video	during development	

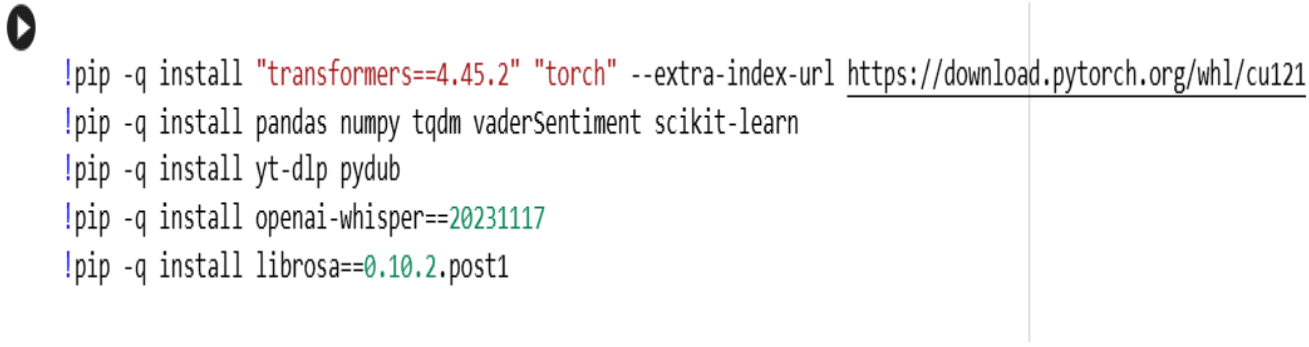
1.3 Software Requirement:

audio & Speech Libraries	Requirement	Version / Notes
Programming Language	Python	Version 3.10+ (project developed in 3.12)
Jupyter Environment	Jupyter Notebook / JupyterLab	Needed to run Final_Thesis.ipynb
Core Python Libraries	pandas, numpy, matplotlib, seaborn, tqdm, scikit-learn	Latest stable versions
NLP Libraries	vaderSentiment, transformers, torch, sentencepiece	Used for VADER + RoBERTa sentiment tasks
Audio & Speech Libraries	openai-whisper, librosa, soundfile	Required for transcription and prosodic analysis
YouTube & Media Tools	yt-dlp, ffmpeg-python	Needed for video/audio extraction
External Utilities	FFmpeg	Must be installed and added to system PATH
Visualization Libraries	wordcloud, matplotlib	For word clouds and plots
Execution Environment	Local machine OR Google Colab	GPU recommended

1.4 Python Environment Setup

Below is an example of how to set up a virtual environment and install dependencies

```
!pip -q install "transformers==4.45.2" "torch" --extra-index-url
https://download.pytorch.org/whl/cu121
!pip -q install pandas numpy tqdm vaderSentiment scikit-learn
!pip -q install yt-dlp pydub
!pip -q install openai-whisper==20231117
!pip -q install librosa==0.10.2.post1
```



1.5 FFmpeg installation

- Download FFmpeg from the official website: https://ffmpeg.org
- Extract the archive (e.g. to C:\ffmpeg).
- Add the bin folder to the system PATH (e.g. C:\ffmpeg\bin).
- Verify installation with "ffmpeg -version"

2 Project Structure and Data Dependencies

The entire implementation of this research project was performed in Google Colab and all the data sets and results files were saved in Google Drive under the folder My Drive/thesis/. Colab uses temporary runtimes, which means that Google Drive was attached at the beginning of each session to have access to input data, store generated outputs, and persist between executions.

The subsections that follow outline the actual folder organization and data interdependence in the project.

2.1 Project Folder Structure

2.1.1 Input Data

Folder / File	**Location in Drive**	**Description / Contents**
thesis/	MyDrive/thesis/	Main project directory containing all datasets, outputs, and processed files.
data/	MyDrive/thesis/data/	Contains all input datasets and intermediate processed data used for the analysis.
Health_Amazon.csv	thesis/data/	Raw Amazon healthcare review dataset downloaded from Kaggle.
Health_Amazon_cleaned.csv	thesis/data/	Cleaned and preprocessed Amazon reviews (duplicates removed, short reviews filtered).
youtube_audio.mp3	thesis/data/	Extracted YouTube audio file downloaded via yt-dlp in Colab.
youtube_transcript.txt	thesis/data/	Whisper-generated transcript of the YouTube review.

2.1.2 Output Data

Folder / File	**Location in Drive**	**Description / Contents**
outputs/	MyDrive/thesis/outputs/	All final processed outputs, figures, sentiment results, and summary tables.
Amazon_preprocessing_summary.csv	thesis/outputs/	Summary statistics generated after cleaning Amazon reviews.
Amazon_sentiment_features_fast.csv	thesis/outputs/	Amazon sentiment analysis (VADER + RoBERTa).
YouTube_sentiment_features_fast.csv	thesis/outputs/	YouTube transcript sentiment analysis.

comparative_summary.csv	thesis/outputs/	Comparison between Amazon and YouTube sentiment metrics.
sentiment_summary.csv	thesis/outputs/	Consolidated sentiment summary table.
YouTube_transcript_by_product.csv	thesis/outputs/	Transcript segments categorized by product.

2.1.3 Figures (plots & visualizations)

Folder / File	Location in Drive	Description / Contents
Figures (plots & visualizations)	thesis/outputs/	Includes histogram plots, waveform, spectrogram, word clouds, etc.
fig_A1_amazon_len.png	thesis/outputs/	Amazon review length distribution plot.
fig_A2_amazon_rating.png	thesis/outputs/	Amazon rating distribution plot.
fig_A3_youtube_topwords.png	thesis/outputs/	Most frequent words in the YouTube transcript.
fig_A4_youtube_waveform.png	thesis/outputs/	YouTube audio waveform visualization.
fig_A5_youtube_spectrogram.png	thesis/outputs/	Spectrogram of the YouTube audio.
fig_B5_amazon_wordcloud.png	thesis/outputs/	Word cloud generated from Amazon reviews.
fig_B6_youtube_wordcloud.png	thesis/outputs/	Word cloud generated from YouTube transcript.

2.1.4 YouTube Product-wise sentiment output

Folder / File	Location in Drive	Description / Contents
yt_products/	MyDrive/thesis/outputs/yt_products/	Product-specific transcript extractions used for fine-grained sentiment analysis.
acupressure_mat.txt	thesis/outputs/yt_products/	YouTube transcript segments about "Acupressure Mat".
cupping_massage_set.txt	thesis/outputs/yt_products/	Segments about "Cupping Massage Set".
dna_repair_tuning_fork.txt	thesis/outputs/yt_products/	Segments about "DNA Repair Tuning Fork".
dragons_blood_salve.txt	thesis/outputs/yt_products/	Segments about "Dragon's Blood Salve".
dreamegg_sound_machine.txt	thesis/outputs/yt_products/	Segments about "Dreamegg Sound Machine".
general.txt	thesis/outputs/yt_products/	Non-product conversational transcript (general narration).
germanium_earrings.txt	thesis/outputs/yt_products/	Segments about "Germanium Earrings".

liquid_oxygen.txt	thesis/outputs/yt_products/	Segments about “Liquid Oxygen Supplement”.
message_pill_capsules.txt	thesis/outputs/yt_products/	Segments about “Message Pill Capsules”.
mzu_sleep_mask.txt	thesis/outputs/yt_products/	Segments about “MZU Sleep Mask”.

2.2 Google Colab Integration With Google Drive

Because the runtime resets in Google Colab, the notebook mounts Google Drive at the beginning of each session using:

```
from google.colab import drive
drive.mount('/content/drive')
```

Once mounted, files are accessed using absolute paths such as:

```
/content/drive/MyDrive/thesis/data/Health_Amazon.csv
/content/drive/MyDrive/thesis/outputs/
```

All generated CSV files, figures, word clouds, and product-level analysis files were automatically saved in the appropriate subdirectories described above.

2.3 Kaggle API Authentication

Amazon data was downloaded directly through a Colab Kaggle API through Kaggle.
Since the credentials cannot be persisted in Colab, the kaggle.json file has to be uploaded separately each time the runtime is restarted.

```
import json, os, shutil
os.makedirs("/root/.kaggle", exist_ok=True)
shutil.copy("/content/kaggle.json", "/root/.kaggle/kaggle.json")
os.chmod("/root/.kaggle/kaggle.json", 0o600)

!pip -q install kaggle
!kaggle datasets download -d tarekziad/health-amazon -p "{DATA.as_posix()}" --unzip

import glob
amz_candidates = glob.glob(str(DATA / "*.csv"))
amz_candidates
```

By using this code we can acess the data stored on Kaggle with the help of “kaggle.json” file.

2.4 Colab Runtime Notes for Reproducibility

- Google drive should be preferred being mounted every session.
- Every reset of the runtime requires re-uploading of kaggle.json.
- Whisper model, RoBERTa model or huge libraries can re-download based on session memory.
- All outputs saved will remain in MyDrive/thesis/, and not within /content/.

3 Execution Steps and Reproducibility

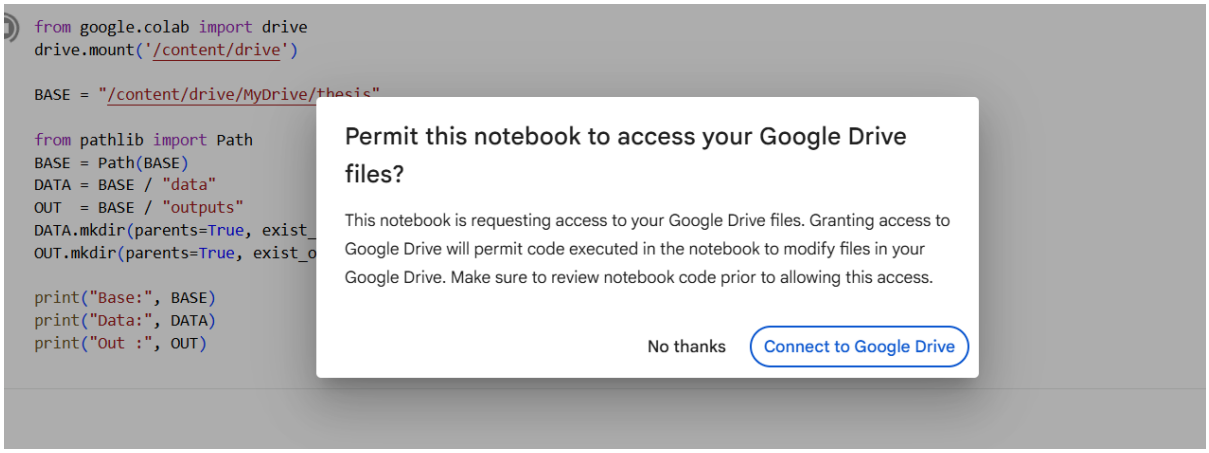
Here, this section gives the detailed guidelines that will be used in carrying out the project code to achieve all the results. All the analysis was performed in Google Colab, which is based on a NVIDIA T4-based GPU-accelerated Python environment on a cloud. Even though, Google Colab makes use of temporary runtimes, certain tasks such as installing Google drive and loading of the authentication file (kaggle.json) must be re-run in each re-opening of a notebook or session re-restart.

3.1 Initial Setup in Google Colab

Before executing any code:

1. In Colab, go to:
   Runtime → Change Runtime Type → Hardware Accelerator → GPU
2. Ensure the GPU type is **NVIDIA T4**.
3. Save and restart the session.

The GPU significantly accelerates Whisper transcription and transformer-based sentiment analysis.



Mount Google Drive, all project files (data, outputs, transcripts, visualizations) are stored inside:
MyDrive/thesis/

To access these files inside Colab:
from google.colab import drive
drive.mount('/content/drive')

After mounting, all file paths follow:
/content/drive/MyDrive/thesis/

3.2 Kaggle Authentication (Required Every Session)

Because Colab clears credentials on every reset, the user must upload the **kaggle.json** file at the beginning of each session.

Copy to Kaggle Directory:

```
!mkdir -p ~/.kaggle
!cp kaggle.json ~/.kaggle/
!chmod 600 ~/.kaggle/kaggle.json

Download Dataset
!kaggle datasets download -d <dataset-name>
!unzip <file>.zip
/content/drive/MyDrive/thesis/data/
```

3.3 Data Loading and Preprocessing (Amazon Reviews)

In this step we are loading Amazon dataset and pre-processing it to get our dataset ready for analysis:
amazon_df = pd.read_csv('/content/drive/MyDrive/thesis/data/Health_Amazon.csv')

Preprocessing steps performed:

- Remove duplicates
- Remove reviews shorter than 5 words
- Clean text: punctuation, whitespace, URLs, emojis
- Rename columns to a consistent schema
- Generate new columns
    - Review_Text_Clean
    - Review_Text_Len_Words

Save cleaned dataset and pre-processing summery:
Health_Amazon_cleaned.csv
Amazon_preprocessing_summary.csv

After performing this step we will get clean Amazon dataset.

3.4 YouTube Audio Download, Processing, and Transcription

The YouTube video was processed using yt-dlp to extract .mp3 audio:

!yt-dlp -x --audio-format mp3 <youtube-url> -o "/content/drive/MyDrive/thesis/data/youtube_audio.mp3"

Whisper (small model) was used due to Colab GPU compatibility:
!pip install openai-whisper
!whisper "/content/drive/MyDrive/thesis/data/youtube_audio.mp3" --model small --language en --output_dir "/content/drive/MyDrive/thesis/data/"

Generated file:
youtube_transcript.txt

In above step, YouTube data has been downloaded in .mp3 format and processed using whisper to get the text file as a transcript.

3.5 NLP Processing and Sentiment Analysis

Till this step our both the datasets are cleaned, pre-processed and ready for our model building. Now we are moving to NLP models VADER and RoBERTa.

3.5.1 VADER Sentiment

Used for lightweight polarity scoring:

from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer

```
# VADER
sid = SentimentIntensityAnalyzer()
def ensure_vader(df, col):
    if "sent_vader" not in df.columns:
        df["sent_vader"] = df[col].astype(str).apply(lambda t: sid.polarity_scores(t)["compound"])
        print("→ computed VADER")
    else:
        print("VADER already present — skipped")
```

Applied to:

- Amazon cleaned reviews
- Individual transcript sentences
- Product-level extracted text

Output:

1. "Amazon_sentiment_features_fast.csv" for Amazon reviews.

title	review	score	review_text_clean	review_text_len	rating	sent_vader
Jobst Ultrasheer 15-20 Knee	Excellent stockings for lon	5	Excellent stockings for	24	5	0.6923
Jobst Ultrasheer 15-20 Knee	After I had a DVT my doct	5	After I had a DVT my d	114	5	0.8546
Jobst Ultrasheer 15-20 Knee	My doctor recommended	5	My doctor recommenc	64	5	-0.4716
Jobst Ultrasheer 15-20 Knee	Stockings fit well and are c	5	Stockings fit well and a	22	5	0.8209
Jobst Ultrasheer 15-20 Knee	Excellent product. Howeve	5	Excellent product. Hov	68	5	0.7787
Jobst Ultrasheer 15-20 Knee	Does a very good job of re	5	Does a very good job c	44	5	0.8764
Jobst Ultrasheer 15-20 Knee	It took almost 3 weeks to	3	It took almost 3 weeks	63	3	0.0111
Jobst Ultrasheer 15-20 Knee	sizes are much smaller tha	2	sizes are much smaller	40	2	0.4926
Jobst Ultrasheer 15-20 Knee	This model may be ok for	1	This model may be ok	31	1	-0.3134

2. "YouTube_sentiment_features_fast.csv"

doc_id	text	sent_vader
youtube_video_1	Germaniur	0.9999

In this step we get the sentiment scores for each cleaned amazon review and every stance of YouTube transcript.

3.5.2 Transformer (RoBERTa) Sentiment

Unlike the VADER model who struggles with the sarcasm and complex sentences our 2nd model will go in depth and cover all the context not just words.

Using HuggingFace pipeline:

from transformers import AutoTokenizer, AutoModelForSequenceClassification

```
USE_MODEL_ID = "cardiffnlp/twitter-roberta-base-sentiment-latest"
BATCH_SIZE   = 128 if device.type=="cuda" else 16   # lower than DistilBERT

tokenizer = AutoTokenizer.from_pretrained(USE_MODEL_ID)
model = AutoModelForSequenceClassification.from_pretrained(USE_MODEL_ID).to(device).eval()
```

Model produces labels:

- Positive
- Neutral
- Negative

A calibration regression was applied to harmonize RoBERTa scores with VADER polarity.

Saved under: MyDrive/thesis/outputs/

1. Amazon_sentiment_features_fast.csv

title	review	score	review_text_clean	review_text_len	rating	sent_vader	sent_model_calib
Jobst Ultrasheer 15-20 Knee	Excellent stockings for lon	5	Excellent stockings for	24	5	0.6923	0.234327397
Jobst Ultrasheer 15-20 Knee	After I had a DVT my doct	5	After I had a DVT my d	114	5	0.8546	0.346491672
Jobst Ultrasheer 15-20 Knee	My doctor recommended	5	My doctor recommend	64	5	-0.4716	-0.570034895
Jobst Ultrasheer 15-20 Knee	Stockings fit well and are c	5	Stockings fit well and a	22	5	0.8209	0.323201863
Jobst Ultrasheer 15-20 Knee	Excellent product. Howeve	5	Excellent product. Hov	68	5	0.7787	0.294037769
Jobst Ultrasheer 15-20 Knee	Does a very good job of re	5	Does a very good job c	44	5	0.8764	0.361557484
Jobst Ultrasheer 15-20 Knee	It took almost 3 weeks to	3	It took almost 3 weeks	63	3	0.0111	-0.236444657
Jobst Ultrasheer 15-20 Knee	sizes are much smaller tha	2	sizes are much smaller	40	2	0.4926	0.096316271
Jobst Ultrasheer 15-20 Knee	This model may be ok for	1	This model may be ok	31	1	-0.3134	-0.460704098

2. YouTube_sentiment_features_fast.csv

doc_id	text	sent_vader	sent_model
youtube_video_1	Germaniur	0.9999	0.865314

3.5.3 Audio Feature Extraction (YouTube)

To catch the vocal emotions from the extracted audio we are using Librosa.

librosa was used for:

- Waveform extraction
- Spectrogram generation
- Tempo estimation
- RMS energy
- Zero-crossing rate

Example:

```
import librosa
y, sr = librosa.load('/content/drive/MyDrive/thesis/data/youtube_audio.mp3')
rms = librosa.feature.rms(y=y)
```

1. For Waveform:

```
if YT_MP3.exists():
    y, sr = librosa.load(YT_MP3, sr=None)
    dur = librosa.get_duration(y=y, sr=sr)
    print(f"[Audio] Duration: {dur/60:.1f} min | Sampling rate: {sr} Hz | Samples: {len(y):,}")

    plt.figure(figsize=(10,3))
    librosa.display.waveshow(y, sr=sr, color="#FF9900")
    plt.title("YouTube Audio Waveform")
    plt.xlabel("Time (s)"); plt.ylabel("Amplitude")
    plt.tight_layout()
    plt.savefig(OUT / "fig_A4_youtube_waveform.png", dpi=200)
    plt.show()


else:
    print("youtube_audio.mp3 not found — run yt-dlp step first.")
```

2. For spectrogram:

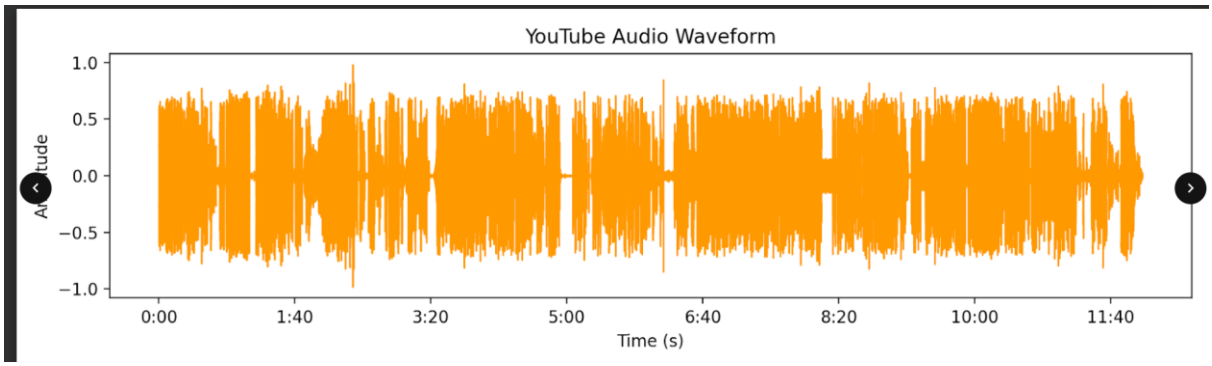
```
 # 3. spectrogram
if YT_MP3.exists():
    y, sr = librosa.load(YT_MP3, sr=None)
    dur = librosa.get_duration(y=y, sr=sr)
    print(f"[Audio] Duration: {dur/60:.1f} min | Sampling rate: {sr} Hz | Samples: {len(y):,}")
    D = librosa.amplitude_to_db(np.abs(librosa.stft(y)), ref=np.max)
    plt.figure(figsize=(8,4))
    librosa.display.specshow(D, sr=sr, x_axis='time', y_axis='log', cmap='inferno')
    plt.title("Spectrogram (YouTube Audio)")
    plt.colorbar(format="%+2.0f dB")
    plt.tight_layout()
    plt.savefig(OUT / "fig_A5_youtube_spectrogram.png", dpi=200)
    plt.show()

else:
    print("youtube_audio.mp3 not found — run yt-dlp step first.")
```

Figures generated:

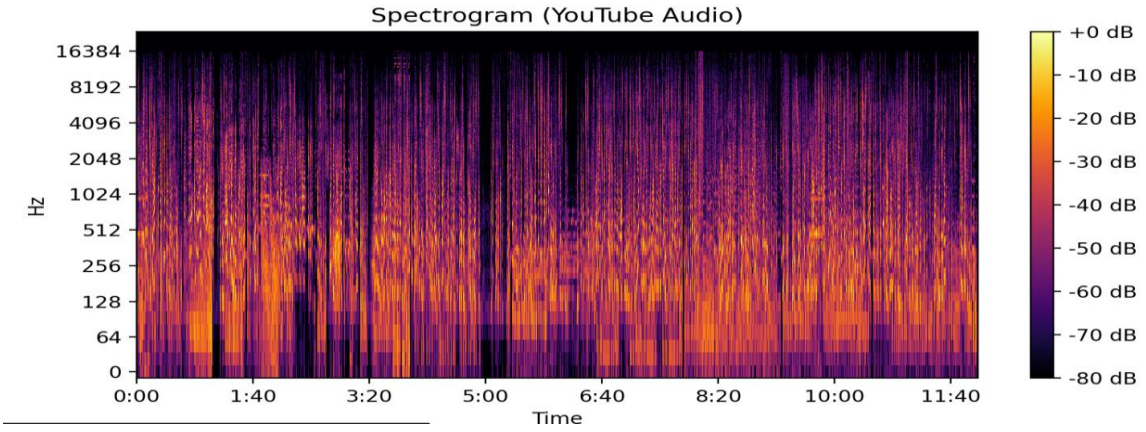
- Audio waveform
- Spectrogram

Stored in:
thesis/outputs/

1. Waveform



2. Spectogram



3.5.4 Product-Level Transcript Segmentation and Analysis

For the case study, product-wise sentiment analysis. Because we are just getting overall sentiments using a transcript and to see which product got positive or which is getting negative score we performed this step:

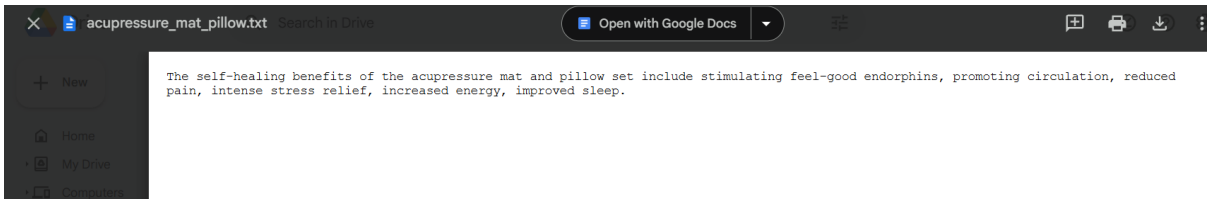
Using a keyword dictionary, transcript segments were grouped into separate files:
Stored in:
thesis/outputs/yt_products/

```
sentences = [s.strip() for s in re.split(r"(?<=[.!?])\s+", raw_text) if s.strip()]

def assign_product(sentence: str) -> str:
    hits = []
    for prod, pat in PRODUCT_PATTERNS.items():
        if pat.search(sentence):
            hits.append(prod)
    if not hits:
        return "general"
    counts = {prod: len(PRODUCT_PATTERNS[prod].findall(sentence)) for prod in hits}
    return max(counts, key=counts.get)

assigned = [assign_product(s) for s in sentences]

buckets = defaultdict(list)
for s, prod in zip(sentences, assigned):
    buckets[prod].append(s)

df_prod = (
    pd.DataFrame(
        [{"product": k, "text": " ".join(v), "n_sentences": len(v)} for k, v in buckets.items()]
    )
    .sort_values("product")
    .reset_index(drop=True)
)
```

Using above code we get the 11 different files for each distinct product.

Examples:

- acupressure_mat.txt
- cupping_massage_set.txt
- liquid_oxygen.txt
- general.txt



Each was analyzed separately using:

- VADER
- RoBERTa

```
# VADER per product
from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer
sid = SentimentIntensityAnalyzer()
df_prod["sent_vader"] = df_prod["text"].apply(lambda t: sid.polarity_scores(t)["compound"])

#RoBERTa per product
try:
    _ = sentiment_batch_scores
    df_prod["sent_roberta"] = sentiment_batch_scores(
        df_prod["text"].astype(str).tolist(), batch_size=8, max_len=256, show_progress=False
    )
except NameError:
    df_prod["sent_roberta"] = pd.NA

OUT_CSV = OUT / "YouTube_transcript_by_product.csv"
df_prod.to_csv(OUT_CSV, index=False)
print("Saved ->", OUT_CSV)
```

Final aggregated results saved for each product as:

“YouTube_transcript_by_product.csv”

	product	text	n_sentences	sent_vader	sent_roberta
0	acupressure_mat_pillow	The self-healing benefits of the acupressure m...	1	0.8910	0.931448
1	cupping_massage_instrument	We have the intelligent breathing, cupping, ma...	3	0.7430	-0.613250
2	dna_repair_tuning_fork	This is a 528 Hertz healing-tuned pipe DNA rep...	2	0.3612	0.035557
3	dragons_blood_sage	Three piece dragon's blood sage smudging kit b...	2	0.8689	0.739511
4	dreamegg_sound_machine	We have the dream egg, which is a travel sound...	3	0.6908	0.957890
5	general	Boosting the immune system, rejuvenating cellu...	169	0.9998	0.790927
6	germanium_earrings	Germanium therapy aids in restoring balance an...	5	0.9371	0.542576
7	liquid_oxygen	Liquid oxygen. Liquid oxygen. In our oxygen de...	4	0.3612	-0.000087
8	message_pill_capsules	Message pill co. Positive messages in a bottle...	4	0.8739	0.322300
9	mzu_sleep_mask	We have the MZU Sleep Mask Health 3D Eye Mask.	1	0.0000	0.105093
10	tourmaline_energy_shield	It's also grounding and helpful to balancing a...	5	-0.5859	-0.261239

3.5.5 Comparative Analysis and Visualization

The notebook generates:

- Word clouds
- Review length histograms
- Rating distribution plots
- Audio waveform & spectrogram
- Sentiment distribution comparisons

Example:

1. Sentiment comparison:

```
pear  = cal["sent_vader"].corr(cal["sent_model_true"], method="pearson") if len(cal) else np.nan
spear = cal["sent_vader"].corr(cal["sent_model_true"], method="spearman") if len(cal) else np.nan
amz_mean = float(df_amz["sent_model_calib"].mean()) if "sent_model_calib" in df_amz.columns else np.nan
yt_mean  = float(df_yt["sent_model"].mean()) if "sent_model" in df_yt.columns else np.nan

print(f"Agreement on sample (VADER vs model): Pearson={pear:.3f}, Spearman={spear:.3f}")
print("Amazon mean (calibrated):", round(amz_mean, 3))
print("YouTube mean (model):    ", round(yt_mean, 3))
```

```
Agreement on sample (VADER vs model): Pearson=0.549, Spearman=0.565
Amazon mean (calibrated): 0.102
YouTube mean (model):     0.865
```

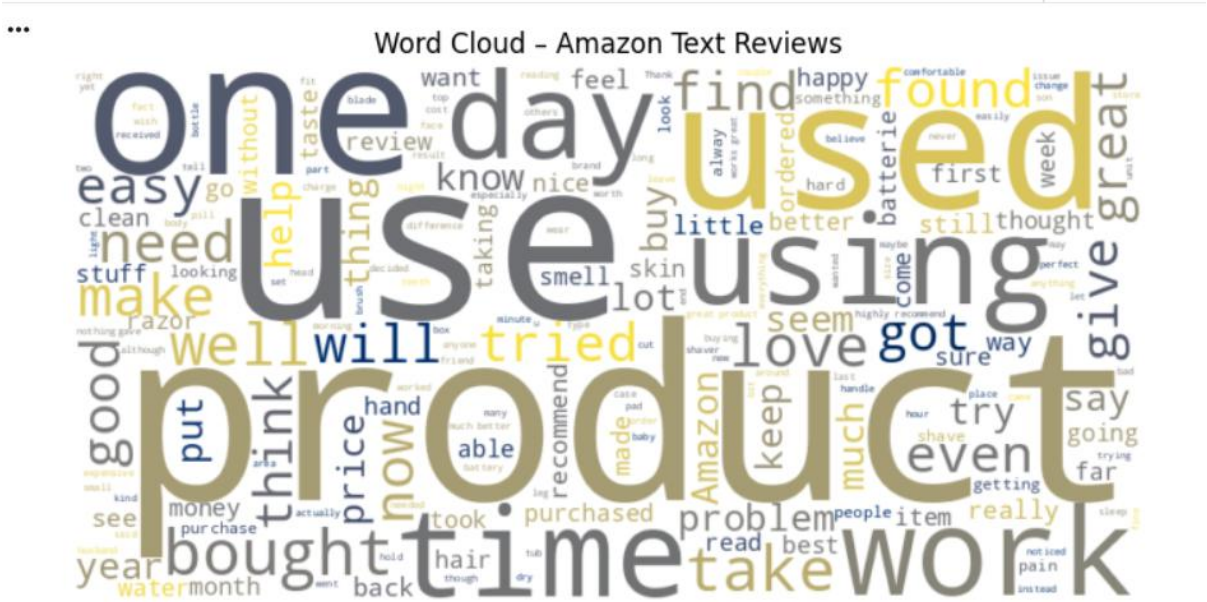
Outputs stored inside:
thesis/outputs/

metric	value
amazon_rows	378589
youtube_rows	1
amazon_mean_calibrated	0.102144
youtube_mean_model	0.865314
pearson_vader_vs_model_sample	0.548782
spearman_vader_vs_model_sample	0.564967

2. word cloud

```
from wordcloud import WordCloud
import matplotlib.pyplot as plt

# Amazon word cloud
CLEAN = DATA / "Health_Amazon_clean.csv"
df_amz = pd.read_csv(CLEAN)
amz_text = " ".join(df_amz["review_text_clean"].astype(str))

plt.figure(figsize=(7,5))
wc_amz = WordCloud(width=800, height=400, background_color="white",
                   colormap="cividis").generate(amz_text)
plt.imshow(wc_amz, interpolation="bilinear"); plt.axis("off")
plt.title("Word Cloud – Amazon Text Reviews")
plt.tight_layout(); plt.savefig(OUT / "fig_B5_amazon_wordcloud.png", dpi=200)
plt.show()
```

Outputs stored inside:
thesis/outputs/



Key summary files:

- sentiment_summary.csv
- comparative_summary.csv

Key visualizations include:

- fig_A1_amazon_len.png
- fig_A2_amazon_rating.png
- fig_B5_amazon_wordcloud.png
- fig_B6_youtube_wordcloud.png

3.5.6 Reproducibility Guidelines

To ensure full reproducibility:

Task	Reason
Mount Google Drive	Gives access to datasets & outputs

Upload kaggle.json	Authentication resets every Colab session
Reinstall Whisper / yt-dlp (if needed)	Colab may clear installed packages
Re-download models	Depends on runtime state
Set GPU runtime	Required for fast Whisper/transformers

Deterministic Output Notes

- Random seeds were set where sampling occurred
- Model versions were fixed through pip installs
- Output paths were consistently stored in Google Drive
- GPU type (T4) does not affect numerical sentiment results

References

FFmpeg (2024) *FFmpeg: A complete, cross-platform solution to record, convert and stream audio and video*. Available at: https://ffmpeg.org .

Hugging Face (2024) *Transformers: State-of-the-art Natural Language Processing*. Available at: https://huggingface.co/transformers .

Kaggle (2024) *Kaggle API Documentation*. Available at: https://www.kaggle.com/docs/api .

Librosa (2024) *Librosa: Python library for audio and music analysis*. Available at: https://librosa.org .

OpenAI (2023) *Whisper: Robust Speech Recognition via Sequence-to-Sequence Models*. Available at: https://github.com/openai/whisper .

Python Software Foundation (2024) *Python Programming Language*. Available at: https://www.python.org .

VADER Sentiment (Hutto, C.J. and Gilbert, E.E., 2014) *VADER: A Parsimonious Rule-Based Model for Sentiment Analysis of Social Media Text*. Proceedings of ICWSM, pp. 216–225.

yt-dlp (2024) *yt-dlp: Command-line program to download videos and audio*. Available at: https://github.com/yt-dlp/yt-dlp .

Google Colab (2024) *Google Colaboratory: Machine Learning and Data Science Environment*. Available at: https://colab.research.google.com .

PyTorch (Paszke, A. et al., 2019) *PyTorch: An Imperative Style, High-Performance Deep Learning Library*. NeurIPS, pp. 8024–8035.

Scikit-learn (Pedregosa, F. et al., 2011) *Scikit-learn: Machine Learning in Python*. Journal of Machine Learning Research, 12, pp. 2825–2830.

pydub (2024) *Pydub – Manipulate audio with Python*. Available at: https://github.com/jiaaro/pydub .