



WOMEN AND NON-CHRISTIAN CHARACTERS IN ELIZABETHAN PLAYS

An NLP based approach for Sentimental Analysis using NER & RE



B.Tech THESIS PROJECT

Bhavya Kumar Garg (20UCC033)

Rashi Gupta (20UCC088)

Twishpeeka Rathore (20UCC110)



Purpose

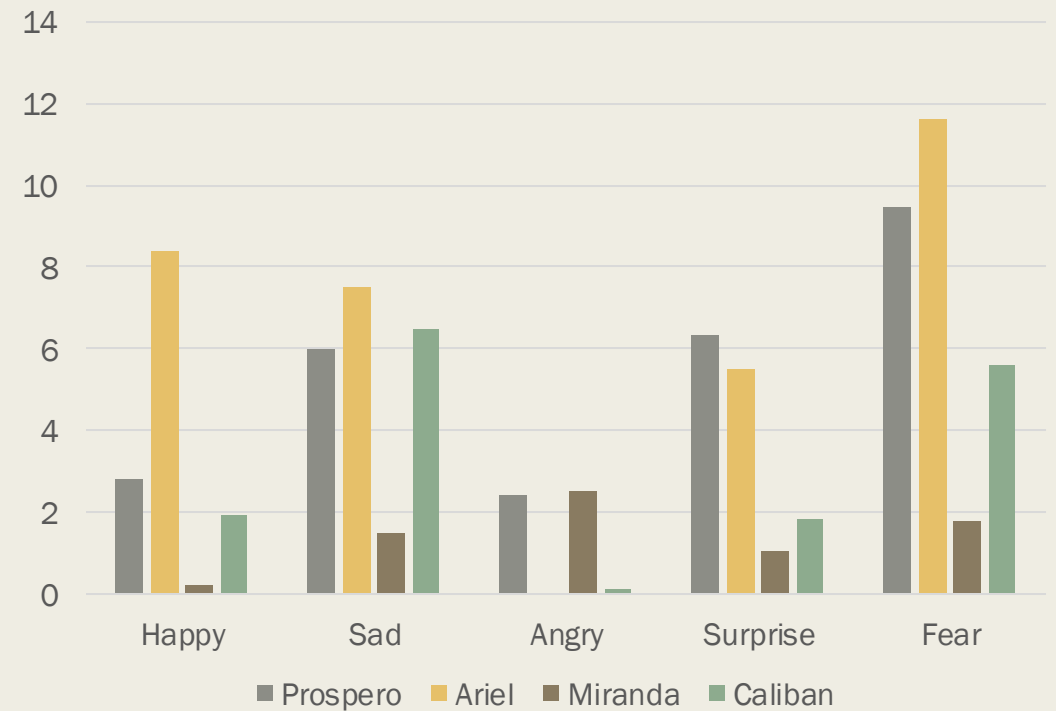
- So far, humans have been responsible for interpreting these characters, relying on their own understanding of the texts. However, there is significant potential for further analysis of the texts to gain valuable insights into how communities were perceived during the Elizabethan Era. Most conclusions drawn have been based on just a few well-known plays written by a small number of authors, which could be problematic given the large amount of literature yet to be explored
- Here, primarily focusing on women and non-christian characters.

RECAP

- Previously, we used lexicon-based approach to pull sentimental analysis of the characters.
- Lexicon-based approach to sentiment analysis uses pre-defined dictionaries of lexicons with assigned sentiment scores .
- The book used for analysis: 'THE TEMPEST'
- Knowledge Base used was the novel itself in Shakespeare English.
- Libraries used: NLTK ,VADER & Text2emotion
- Emotions analysed: Joy, Fear, Anger, Sorrow and Surprise.

Motivation for further research

- Inconsistencies between our results and the scholar reviews.
- Scope for further improvement in our analysis.



Areas of Improvement Identified



The diagram consists of two identical rectangular boxes with rounded corners, arranged side-by-side. Each box has a light gray background and a darker gray border. Inside each box, the text is centered and reads: 'DEPENDENCY DUE TO CONTEXT' for the left box and 'DEPENDENCY DUE TO GRAMMAR' for the right box. The text is in a bold, black, sans-serif font.

**DEPENDENCY
DUE TO
CONTEXT**

**DEPENDENCY
DUE TO
GRAMMAR**

Context based dependency of emotions on different entities

Contextual Nuances:

- Example: In Shylock's famous speech about humanity, the phrase "If you prick us, do we not bleed?" might be challenging to interpret accurately without considering the broader context of the play.

Idiomatic Expressions:

- Example: In Portia's speech about mercy, the metaphorical use of "It droppeth as the gentle rain from heaven" might not be fully captured by a lexicon.

Complex Sentiment Shifts:

- Example: Antonio's melancholic reflections in one scene might contrast sharply with his joyful interactions in another, requiring a nuanced understanding of sentiment changes.

Language barrier or Dependency due to grammar

Previously we were working on the Shakespeare English as our knowledge base

Limited Coverage for Archaic Language:

- Example: Words like "thou," "thee," or "thy" may not have sentiment scores assigned in standard lexicons, impacting the analysis of sentiment in dialogue.

Character Ambiguity:

- Example: Shylock's character exhibits both vengeful and sympathetic qualities, and lexicon-based analysis may oversimplify the sentiment associated with him.

Relation-Extraction (RE)

- Relation extraction is a natural language processing (NLP) task that involves identifying and extracting semantic relationships between entities in text. The goal of relation extraction is to automatically identify and classify the relationships between entities, such as people, organizations, and locations, that are mentioned in the text.
- There are several techniques used for relation extraction, including rule-based systems, machine learning models, and deep learning models. Rule-based systems use handcrafted rules to identify and classify relationships, while machine learning models use algorithms to learn patterns in the data and automatically identify relationships. Deep learning models, such as neural networks, have been shown to be effective in relation extraction tasks, as they can learn complex patterns in the text data.

Need of RE in our work

- **Contextual Understanding:** Relations between characters can provide crucial contextual understanding. Extracting relationships helps in discerning the nuances of interactions, such as friendships, conflicts, alliances, or romantic entanglements, which can significantly influence the sentiment expressed in the text.
- **Fine-Grained Sentiment Analysis:** Knowing the relationships between characters allows for a more fine-grained sentiment analysis. Sentiments expressed within the context of relationships may carry different emotional weight. For example, a positive sentiment expressed between close friends may differ from a positive sentiment expressed between adversaries.
- **Character Dynamics:** Understanding the relationships between characters contributes to a deeper analysis of character dynamics. Sentiments and emotions are often intertwined with the relationships characters share, and relation extraction helps capture these dynamics more accurately.

How to implement RE

We can perform RE through the following steps:

1. Performing NER
2. Defining relationships between entities manually
3. Training the model with the dataset we created manually
4. Extracting the output from our model

Step 0 – Perform NER

- First, we will perform Named-Entity-Recognition (NER), It is a natural language processing (NLP) technique that involves identifying and categorizing named entities in text. Named entities are specific objects, people, places, organizations, and other entities that have a unique name or identifier. NER is used to automatically identify and extract these entities from unstructured text data.
- There are several tools and libraries available for named entity recognition (NER) that can be used for natural language processing tasks. Some of the commonly used tools for NER include SpaCy and NLTK.

NER - Results

Code

```
✓ 35s !pip install spacy  
!python -m spacy download en_core_web_sm
```

```
✓ 10s import spacy  
  
# Load the English language model  
nlp = spacy.load('en_core_web_sm')  
  
# Function to perform NER on a text file  
def perform_ner_on_file(file_path):  
    with open(file_path, 'r', encoding='utf-8') as file:  
        text = file.read()  
  
    # Process the text with SpaCy's NLP pipeline  
    doc = nlp(text)  
  
    # Display named entities  
    for ent in doc.ents:  
        print(f"Entity: {ent.text}, Label: {ent.label}")  
  
# Replace 'file_path' with the path to your text file  
file_path = 'Shylock - Bing Copilot.txt'  
perform_ner_on_file(file_path)
```

Sample Output

```
✓ 10s Entity: Three thousand, Label: CARDINAL  
Entity: three months, Label: DATE  
Entity: Antonio, Label: PERSON  
Entity: Three thousand, Label: QUANTITY  
Entity: three months, Label: DATE  
Entity: Antonio, Label: PERSON  
Entity: Antonio, Label: PERSON  
Entity: Tripoli, Label: GPE  
Entity: India, Label: GPE  
Entity: third, Label: ORDINAL  
Entity: Mexico, Label: GPE  
Entity: fourth, Label: ORDINAL  
Entity: England, Label: GPE  
Entity: rocks2, Label: ORG  
Entity: Three thousand, Label: QUANTITY  
Entity: Antonio, Label: PERSON  
Entity: you4, Label: GPE  
Entity: Christian, Label: NORP  
Entity: Venice5, Label: GPE  
Entity: three thousand, Label: QUANTITY  
Entity: away7, Label: GPE  
Entity: Tubal, Label: ORG  
Entity: Jew, Label: NORP  
Entity: three thousand ducats, Label: QUANTITY  
Entity: three months, Label: DATE  
Entity: Jacob, Label: PERSON  
Entity: Laban, Label: ORG  
Entity: Jacob, Label: PERSON  
Entity: Abraham, Label: PERSON
```

Step I – Defining relationships

- Manually annotate a subset of the data with sentiment labels and relation annotations. This annotated dataset will be used to train and evaluate our model.
- Consider the following sentence:
 - **Sentence:** *"Portia, the wealthy heiress, marries Bassanio, her suitor."*
In this sentence, we can define a relation between the entities "Portia" and "Bassanio" as follows:
 - **Relations:** *"Portia marries Bassanio", "Portia is wealthy", etc.*
So, in this case, "romantic relationship" is the relation between the entities "Portia" and "Bassanio."
- The relation extraction task involves identifying and extracting such relationships from text. The goal is to recognize not only the entities involved but also the nature of their connection or interaction.

Step II – Training Model

- The annotation process involves:
 - *meticulously identifying instances of predefined semantic relationships in a text subset.*
 - *Each relation is paired with contextual examples to intricately guide the subsequent machine learning model.*
 - *This meticulous analysis, characterized by attention to linguistic nuances, enriches the model's understanding.*
- The contextual examples serve a dual purpose:
 - *Delineating relation characteristics*
 - *Instructing the model during training. This intellectually demanding task aims to endow the model with refined semantic comprehension for precise relation extraction.*

Training Examples

Jessica :

Lorenzo , for sure, and *my* love indeed, For whom do I *love* so much?

Entities: Lorenzo, Jessica

Relation: Romantic Relation

Shylock , *lends money* to *Antonio* , with a pound of his flesh as collateral

Entities: Shylock, Antonio

Relation: Business Relation

Step III – Extracting insights

The model aims to identify and categorize instances of specific semantic relationships between entities. The model utilizes our manually fed relationships to extract more complex relations between the entities.

Limitations of RE

- Need for a large manually created dictionary to start with.
- High dependency of accuracy on dictionary.
 - *Need for verification of data due to high variation in emotion scores*

Problem II – Dependency due to grammar

1. **"Are you crazy?"** - Negative sentiment due to the word "crazy," likely indicating a negative sentiment.
 1. Polarity: -0.6 (Example score for negative sentiment)
2. **"Are you serious?"** - Neutral or slightly positive sentiment. The phrase is more likely neutral or leaning towards a positive sentiment.
 1. Polarity: 0.2 (Example score for a slightly positive sentiment)
3. **"Are you kidding me?"** - Depending on context, this phrase could be neutral or slightly negative. The word "kidding" might imply a slightly less intense negative sentiment.
 1. Polarity: -0.2 (Example score for a slightly negative sentiment)

How it affects our work?

1. Elizabethan era “English” or “Shakespeare’s version” cannot be well analyzed using modern dictionaries.
2. Slight variations in choice of words during translation might make the sentences lose their emotions.

How we solved this?

1. Comparing detailed translations of different translations.
2. Sources – Latest LLM's
 1. *Chat GPT*
 2. *Perplexity*
 3. *Bing Copilot*

Expanding the scope of Lexicon based Analysis

1. Choice of Lexicon : **NRC Word-Emotion Association Lexicon (aka EmoLex)**.
2. The NRC Emotion Lexicon is a list of English words and their associations with eight basic emotions (anger, fear, anticipation, trust, surprise, sadness, joy, and disgust) and two sentiments (negative and positive). The annotations were manually done by crowdsourcing.

```
!pip install text2emotion
!pip uninstall emoji -y
!pip install emoji==0.6.0
!pip install textblob nltk
```

```
from textblob import TextBlob
import matplotlib.pyplot as plt

# Function to perform sentiment analysis using TextBlob
def perform_sentiment_analysis(text):
    blob = TextBlob(text)
    return blob.sentiment.polarity, blob.sentiment.subjectivity

# Function to perform emotion analysis using the NRC lexicon
def perform_emotion_analysis(text):
    emotion_categories = {
        'anger': 0,
        'anticipation': 0,
        'disgust': 0,
        'fear': 0,
        'joy': 0,
        'negative': 0,
        'positive': 0,
        'sadness': 0,
        'surprise': 0,
        'trust': 0
    }

    # Load NRC lexicon
    lexicon_path = 'NRC-Emotion-Lexicon-Wordlevel-v0.92.txt'
    with open(lexicon_path, 'r') as file:
```



```
lexicon_path = 'NRC-Emotion-Lexicon-Wordlevel-v0.92.txt'
with open(lexicon_path, 'r') as file:
    lines = file.readlines()
    word_emotion_map = {}
    for line in lines:
        word, emotion, value = line.strip().split('\t')
        if word not in word_emotion_map:
            word_emotion_map[word] = {}
        word_emotion_map[word][emotion] = int(value)

words = text.lower().split()
for word in words:
    if word in word_emotion_map:
        for emotion in emotion_categories.keys():
            emotion_categories[emotion] += word_emotion_map[word].get(emotion, 0)

return emotion_categories

# Read text from a local file
def read_text_file(file_path):
    with open(file_path, 'r') as file:
        return file.read()

# File path of the text file
file_path = 'Jessica - Original.txt' |
```




```
# Read text from the file
sample_text = read_text_file(file_path)

# Perform sentiment analysis
sentiment_polarity, sentiment_subjectivity = perform_sentiment_analysis(sample_text)
print("Sentiment Analysis Scores (TextBlob):")
print(f"Polarity: {sentiment_polarity}, Subjectivity: {sentiment_subjectivity}")

# Perform emotion analysis
emotion_scores = perform_emotion_analysis(sample_text)
print("\nEmotion Analysis Scores (NRC lexicon):")
print(emotion_scores)

# Plotting Sentiment Analysis Scores
sentiment_labels = ['Polarity', 'Subjectivity']
sentiment_scores = [sentiment_polarity, sentiment_subjectivity]

plt.figure(figsize=(8, 6))
plt.bar(sentiment_labels, sentiment_scores, color=['skyblue', 'lightgreen'])
plt.title('Sentiment Analysis Scores')
plt.ylabel('Score')
plt.show()

# Plotting Emotion Analysis Scores
emotions = list(emotion_scores.keys())
scores = list(emotion_scores.values())

plt.figure(figsize=(10, 6))
plt.bar(emotions, scores, color='lightblue')
plt.title('Emotion Analysis Scores')
plt.ylabel('Frequency')
plt.xlabel('Emotion')
plt.xticks(rotation=45)
plt.show()
```

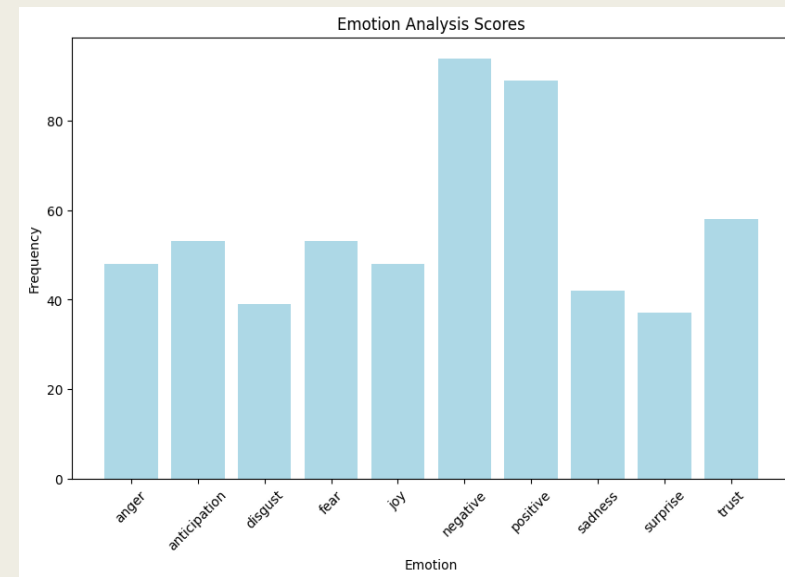
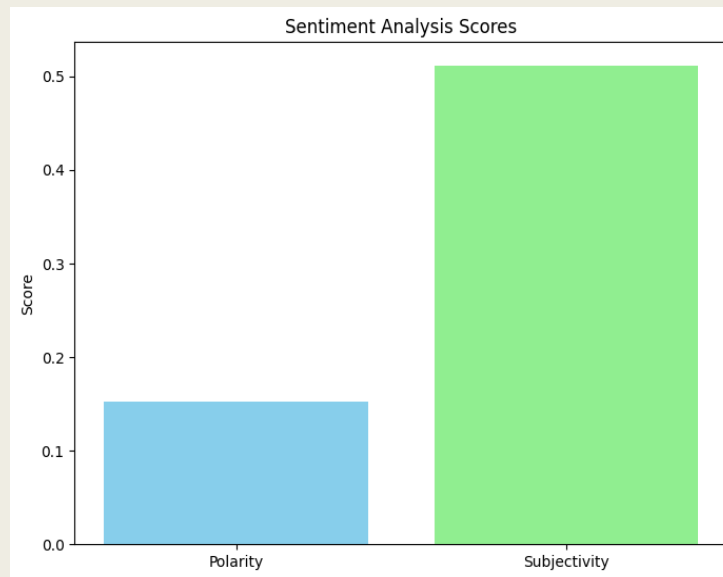
Shylock – Original Text

Sentiment Analysis Scores (TextBlob):

Polarity: 0.15223666796704577, Subjectivity: 0.5111244924326322

Emotion Analysis Scores (NRC lexicon):

{'anger': 48, 'anticipation': 53, 'disgust': 39, 'fear': 53, 'joy': 48, 'negative': 94, 'positive': 89, 'sadness': 42, 'surprise': 37, 'trust': 58}



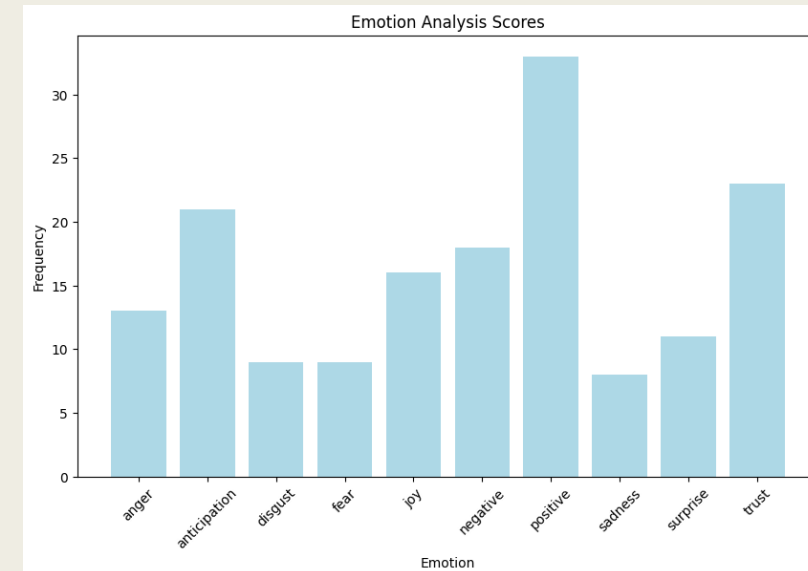
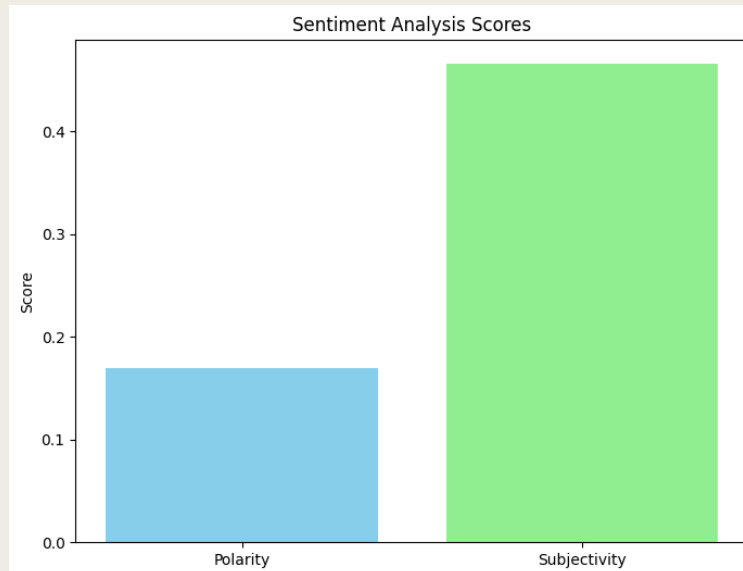
Shylock – Perplexity Translation

Sentiment Analysis Scores (TextBlob):

Polarity: 0.16926540537651644, Subjectivity: 0.46576612687723795

Emotion Analysis Scores (NRC lexicon):

{'anger': 13, 'anticipation': 21, 'disgust': 9, 'fear': 9, 'joy': 16, 'negative': 18, 'positive': 33, 'sadness': 8, 'surprise': 11, 'trust': 23}



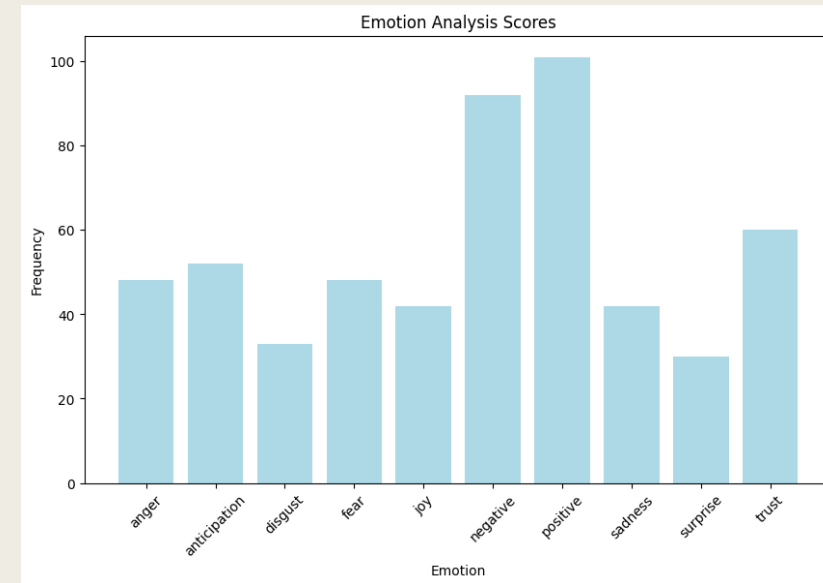
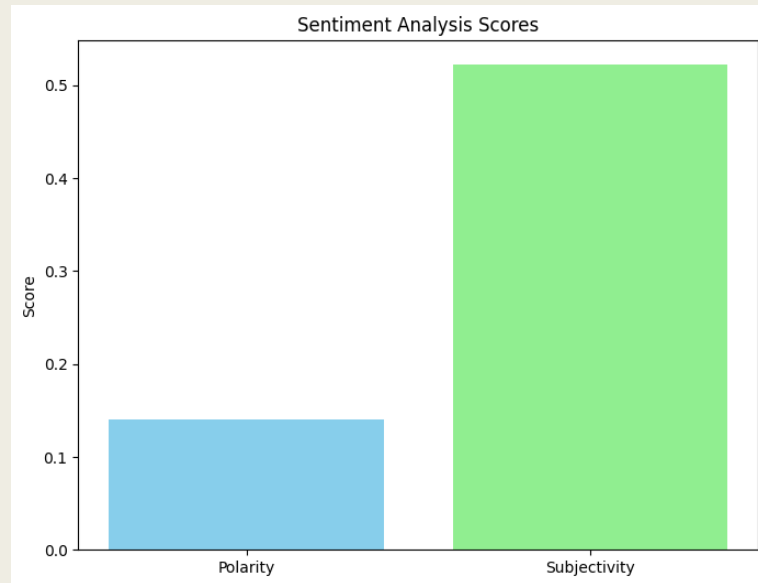
Shylock – Chat GPT Translation

Sentiment Analysis Scores (TextBlob):

Polarity: 0.1398376857968621, Subjectivity: 0.5218754014129741

Emotion Analysis Scores (NRC lexicon):

{'anger': 48, 'anticipation': 52, 'disgust': 33, 'fear': 48, 'joy': 42, 'negative': 92, 'positive': 101, 'sadness': 42, 'surprise': 30, 'trust': 60}



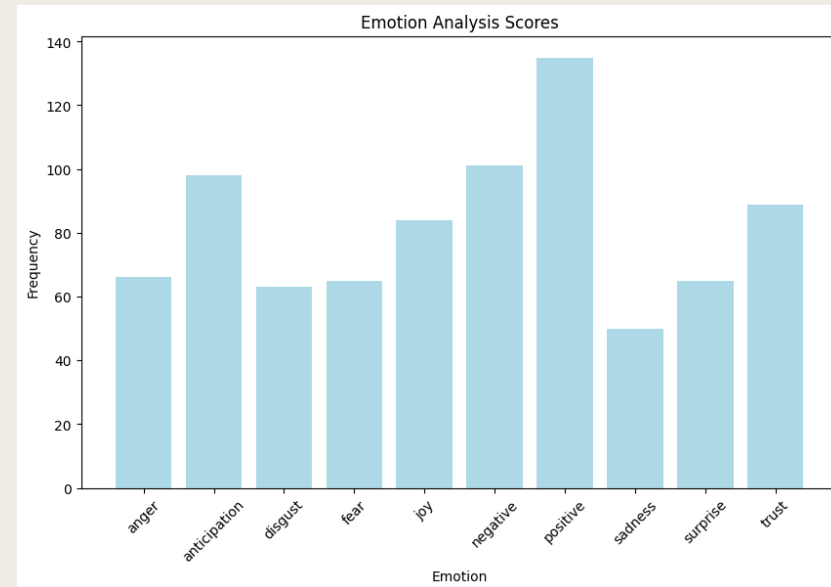
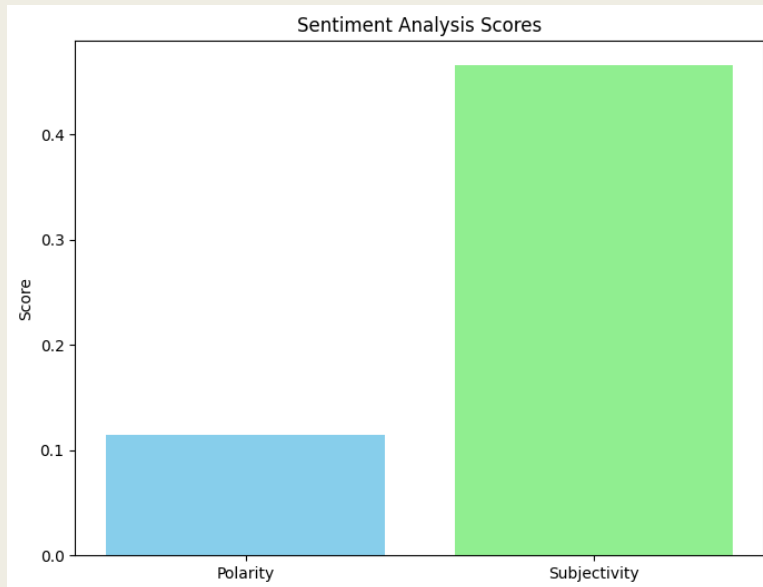
Shylock – Bing Copilot Translation

Sentiment Analysis Scores (TextBlob):

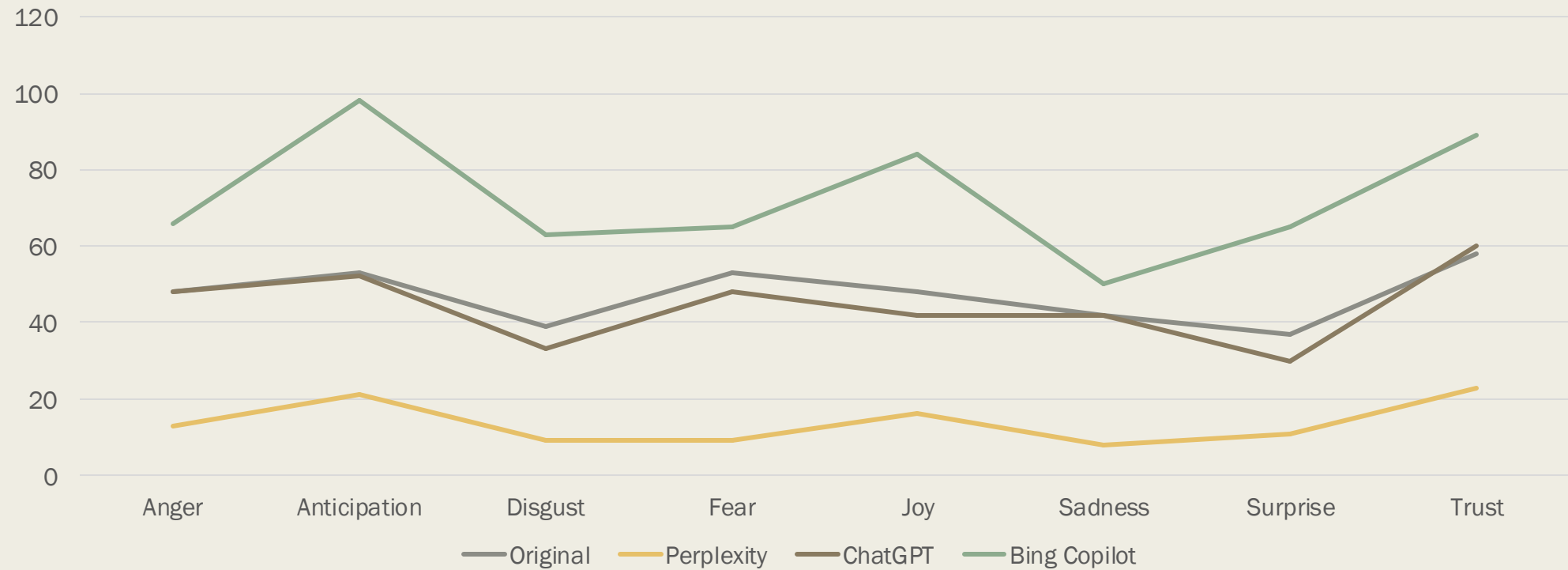
Polarity: 0.1142433572174951, Subjectivity: 0.46528045230631404

Emotion Analysis Scores (NRC lexicon):

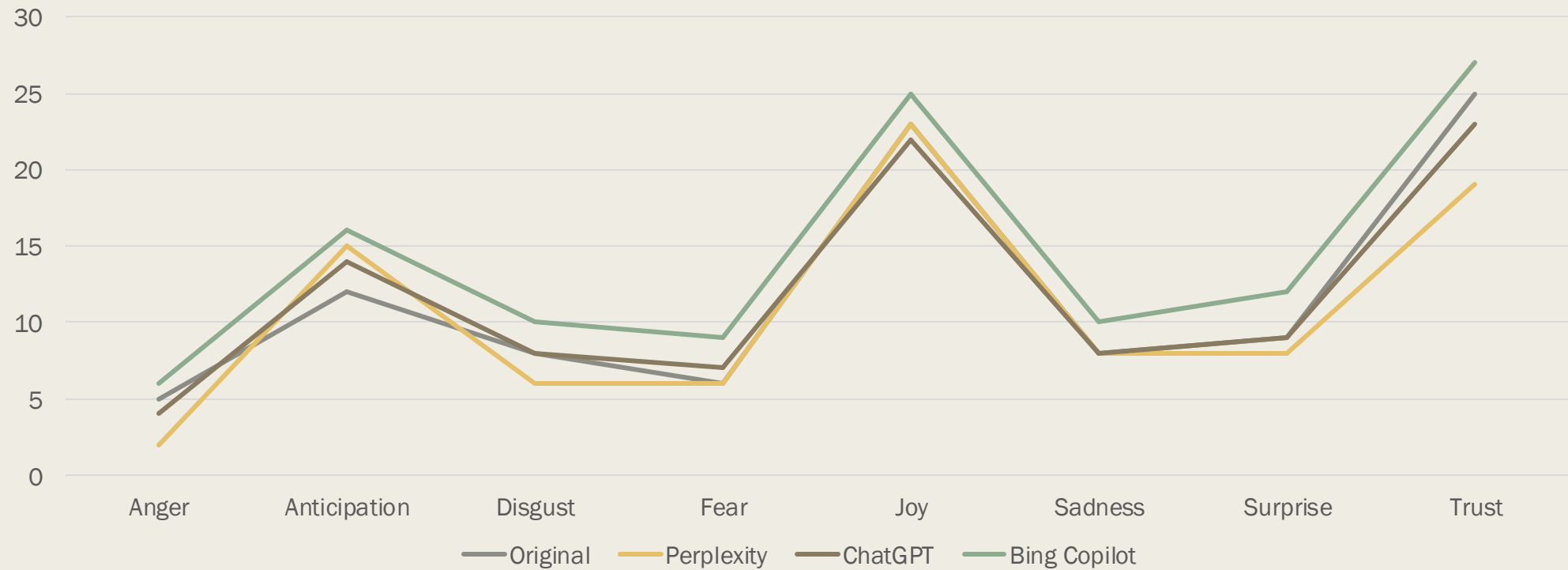
{'anger': 66, 'anticipation': 98, 'disgust': 63, 'fear': 65, 'joy': 84, 'negative': 101, 'positive': 135, 'sadness': 50, 'surprise': 65, 'trust': 89}



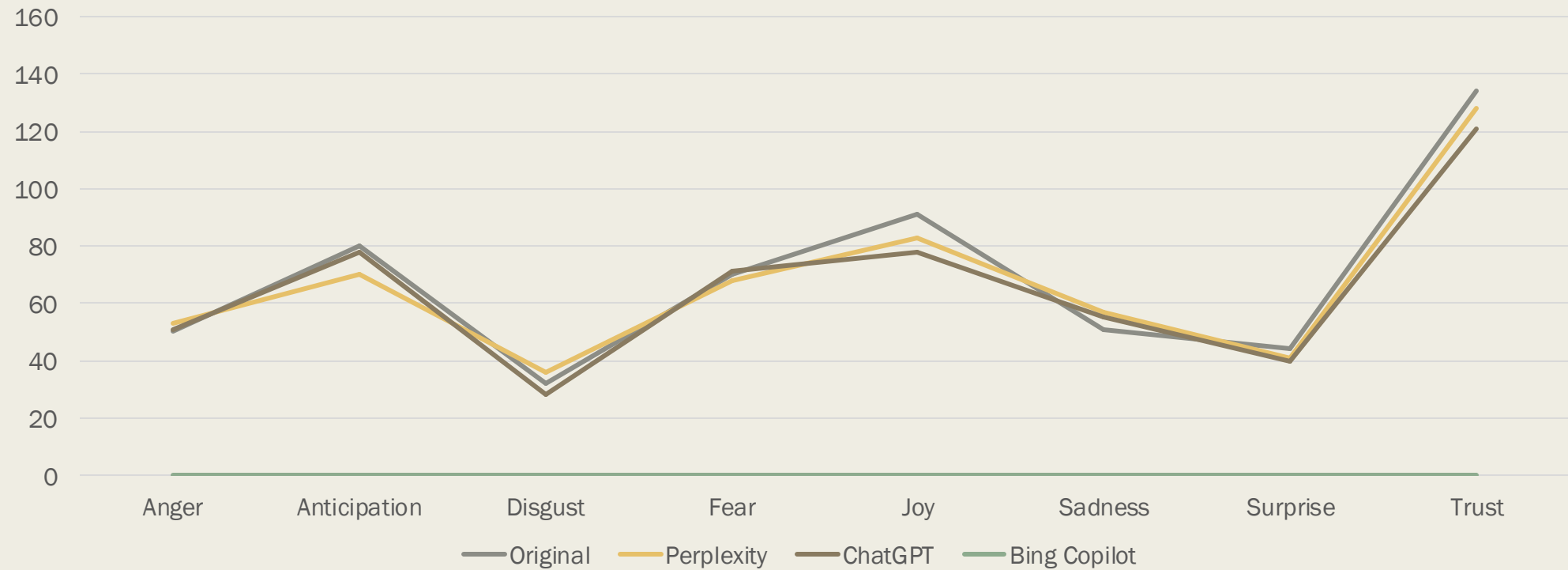
Comparison - Shylock



Comparison - Jessica



Comparison - Portia



Improvements observed : The Merchant of Venice

■ Shylock:

- *Old Approach: Overall Sentiment Score: -0.21*
- *New Approach: Polarity: 0.1142433572174951, Subjectivity: 0.46528045230631404*

■ Jessica:

- *Old Approach: Overall Sentiment Score: 0.15*
- *New Approach: Polarity: 0.05725216450216452, Subjectivity: 0.5902059884559884*

■ Portia:

- *Old Approach: Overall Sentiment Score: 0.18*
- *New Approach: Polarity: 0.1403703133552272, Subjectivity: 0.5110041050903121*

Areas of improvement

- Use of a vast and highly sophisticated Lexicon: NRC Word-Emotion Association Lexicon (aka EmoLex).
- Shift to Polarity and Subjectivity scores from Sentiment Score with detailed analyses of over 10 emotions.
- Comparison and normalization by utilizing multiple LLMs for translation from Shakespeare's English to Modern English.
- Future scope of implementation of Relation Extraction.

Literature survey

■ Relation Extraction | Natural Language Processing

by Chandresh Maurya MAR, 2021

<https://www.youtube.com/watch?v=t0R3NdT1vGY>

Through this talk we learn about the concepts of Relation Extraction. Relation Extraction is a process of identifying and extracting the connections between different things or entities mentioned in a text. This task involves analyzing the text and identifying the entities involved, such as people, places, or things, and then determining how they are related to each other. The goal of relation extraction is to extract these structured relationships from unstructured text data, which can be valuable for a wide range of applications.

One of the key challenges associated with relation extraction is determining the correct relationships between entities. This can be especially difficult when dealing with complex or ambiguous text. Additionally, different types of relations may require different extraction techniques, and the quality of the extracted relations can be affected by factors such as the quality of the input data and the specific extraction algorithm used.

Literature survey

■ TextBlob : Python library for Natural Language Processing

<https://towardsdatascience.com/my-absolute-go-to-for-sentiment-analysis-textblob-3ac3a11d524>

TextBlob, a Python library for Natural Language Processing (NLP), utilizes the Natural Language Toolkit (NLTK) for various tasks, providing a straightforward yet powerful approach to textual data analysis. In sentiment analysis, it employs a lexicon-based approach where pre-defined dictionaries classify words as negative or positive. Sentiment is determined by the semantic orientation and intensity of these words in a sentence, calculated through operations like averaging.

TextBlob outputs polarity and subjectivity scores, with polarity ranging from -1 to 1 and subjectivity from 0 to 1. Polarity denotes sentiment orientation, with negation words reversing it. Subjectivity quantifies personal opinion versus factual information, and TextBlob introduces an intensity parameter to assess word modification. Notably, the library considers semantic labels for nuanced analysis, making it a versatile tool for sentiment and opinion analysis in textual data.

Literature survey

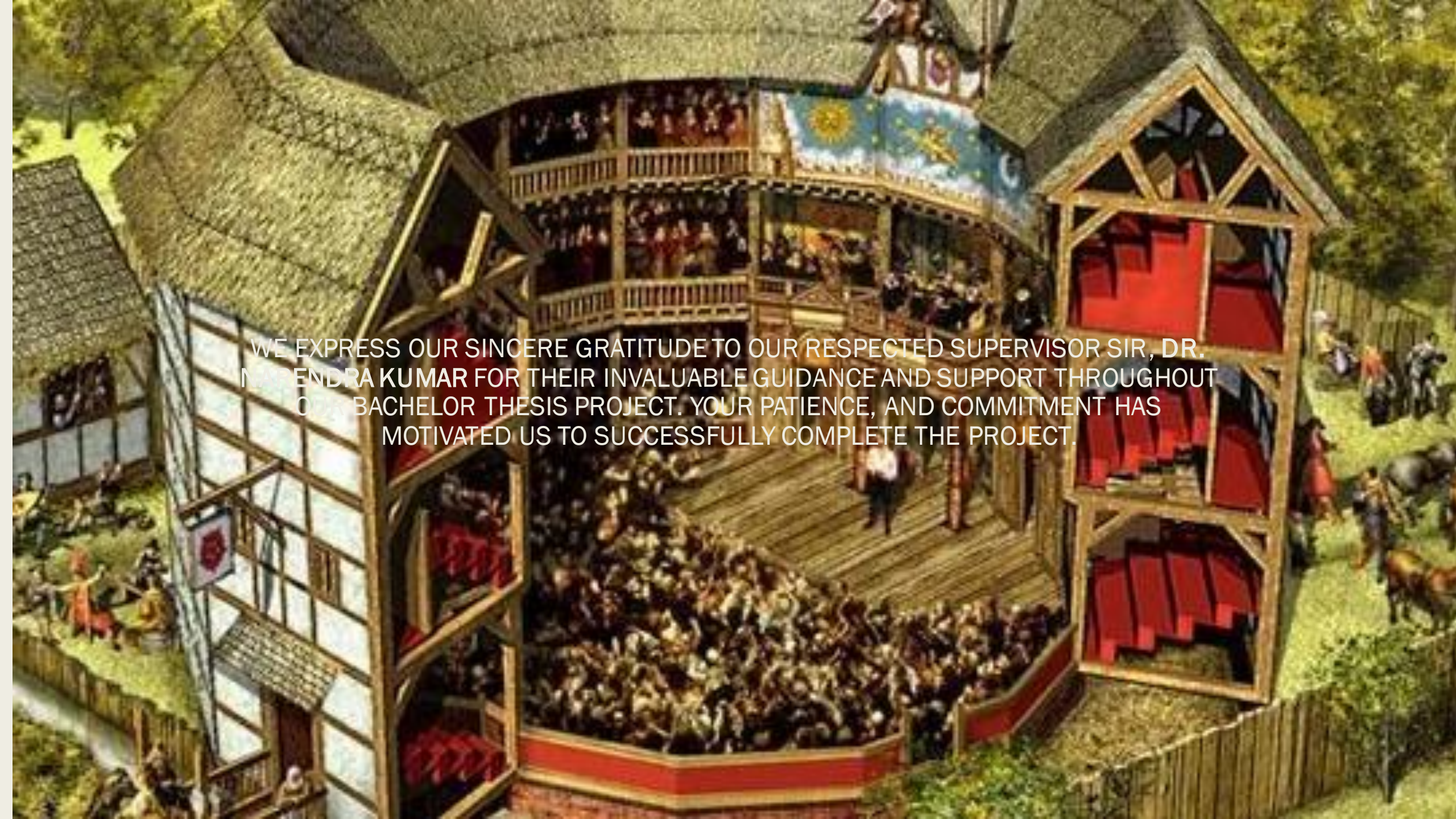
■ NRC Word-Emotion Association Lexicon (aka EmoLex)

Created By: Dr. Saif M. Mohammad, Dr. Peter Turney

<https://saifmohammad.com/WebPages/NRC-Emotion-Lexicon.htm>

The NRC Emotion Lexicon is a resource developed by Saif Mohammad that provides a list of English words associated with eight basic emotions: anger, fear, anticipation, trust, surprise, sadness, joy, and disgust. Each word in the lexicon is annotated with scores indicating the extent to which it is associated with each emotion.

The lexicon is designed to be a valuable tool for researchers and practitioners in natural language processing and related fields, enabling them to analyze and understand the emotional content of text. It can be used for sentiment analysis, emotion detection, and other applications where capturing emotional nuances in language is important.



WE EXPRESS OUR SINCERE GRATITUDE TO OUR RESPECTED SUPERVISOR SIR, DR. NAGENDRA KUMAR FOR THEIR INVALUABLE GUIDANCE AND SUPPORT THROUGHOUT OUR BACHELOR THESIS PROJECT. YOUR PATIENCE, AND COMMITMENT HAS MOTIVATED US TO SUCCESSFULLY COMPLETE THE PROJECT.