EVOLUÇÃO DE SOFTWARE

ANÁLISE DE SENTIMENTOS

GRUPO:

Douglas de Oliveira Deda - 201700053097

Gabriel dos Santos Almeida - 202100011421

Henrick Cardoso dos Santos - 202200060035

Ian da Silva Santos Conceição - 201900123689

Mariana Souza Nunes - 202200059947

Pablo Alves Freire - 202200060080

Rafael Lauton Santos de Oliveira - 202000047753

Yasmim de Andrade Lima - 202000047913

PROJETO SELECIONADO:

ChatTTS

URL DO VÍDEO:

Youtube: https://youtu.be/lOQG-BPdIcU

URL DO PROJETO:

GitHub: https://github.com/faalkor/Evolucao_Software_2025-2_chatTTS

Relação dos participantes

Participantes:

Douglas de Oliveira Deda - 201700053097: 100% de contribuição

Gabriel dos Santos Almeida - 202100011421: 100% de contribuição

Henrick Cardoso dos Santos - 202200060035: 100% de contribuição

Ian da Silva Santos Conceição - 201900123689: 100% de contribuição

Mariana Souza Nunes - 202200059947: 100% de contribuição

Pablo Alves Freire - 202200060080: 100% de contribuição

Rafael Lauton Santos de Oliveira - 202000047753: 100% de contribuição

Yasmim de Andrade Lima - 202000047913: 100% de contribuição

1. Estudo do Projeto ChatTTS

Primeiramente, foi escolhido o projeto ChatTTS, um modelo de Text-to-Speech (TTS) que converte texto em fala em forma de áudio, podendo escolher entre fala feminina ou masculina. O objetivo do grupo é separar no mínimo 100 comments pulls requests desse projeto que está hospedado no GitHub e escolher 3 modelos de linguagem de análise de sentimentos que estão presentes no site Hugging Face para analisar cada um dos pull requests e separar eles em 3 labels que são Negative, Positive e Neutral e ver como está o clima do ambiente do projeto.

Sabendo disso, foi feito um script na linguagem pyhton, cujo o arquivo . py se chama “extract-pr-comments”, para extrair esses pull requests e coloca-los organizados em um arquivo.json chamado “pr_comments_2noise_ChatTTS_closed_nobots_True”. Esse script busca comentários e revisões usando a API do GitHub e coleta título, autor, estado e datas do pull request, comentário gerais (issues/comments), comentários de revisão (pulls/comments).

```python
pr_data = {
    "number": pr_number,
    "title": pr_title,
    "state": pr["state"],
    "user": pr_author,
    "created_at": pr["created_at"],
    "updated_at": pr["updated_at"],
    "closed_at": pr.get("closed_at"),
    "merged_at": pr.get("merged_at"),
    "comments_count": len(all_comments),      Rafael Lauton, 5 days ago • add extracao de
    "issue_comments_count": len(issue_comments),
    "review_comments_count": len(review_comments),
    "comments": [
        {
            "id": comment["id"],
            "user": comment["user"]["login"],
            "body": comment["body"],
            "created_at": comment["created_at"],
            "updated_at": comment["updated_at"],
            "comment_type": "review" if "pull_request_review_id" in comment else "issue"
        }
        for comment in all_comments
    ]
}

all_data["prs"].append(pr_data)
```

Foto de uma parte do script extract-pr-comments.


```
},
"total_prs": 100,
"prs": [
  {
    "number": 946,
    "title": "Visualization of the ChatTTS codebase",
    "state": "closed",
    "user": "ivanmilevtues",
    "created_at": "2025-06-28T19:45:58Z",
    "updated_at": "2025-07-06T15:11:15Z",      You, yesterday • :sparkles:feat: Adicao do modelo Roberta Base
    "closed_at": "2025-07-06T15:11:14Z",
    "merged_at": "2025-07-06T15:11:14Z",
    "comments_count": 3,
    "issue_comments_count": 3,
    "review_comments_count": 0,
    "comments": [
      {
        "id": 3041400027,
        "user": "ivanmilevtues",
        "body": "Thanks for the response!\r\n@fumiama just a quick question you want to add the link to the diagrams into the README?\r\n\r\nAs the
        issue with the structure change, we also have a FREE Github Action which I can setup for you and it can update the diagrams once a week so they
        are always up-to-date :)",
        "created_at": "2025-07-06T12:16:35Z",
        "updated_at": "2025-07-06T12:16:48Z",
        "comment_type": "issue"
```


Foto do .json de pull requests pr_comments_2noise_ChatTTS_closed_nobots_True.

2. Processo de Seleção dos Modelos

1. RoBERTa (Twitter-RoBERTa-base-sentiment-latest)

Descrição: Modelo desenvolvido a partir da arquitetura RoBERTa ou Robustly Optimized BERT Approach (uma variação otimizada do BERT), treinado em um grande conjunto de dados do Twitter para classificação de sentimentos.

Motivo da Escolha:

- É altamente eficaz na detecção de nuances emocionais em textos curtos e informais, semelhantes aos encontrados em interações de pull requests.
- Apresenta excelente desempenho em inglês, especialmente em linguagem com expressões coloquiais e variações sintáticas comuns em comentários de desenvolvedores.
- Modelo robusto e atualizado regularmente pela CardiffNLP, garantindo boa generalização.

Normalização, Inicialização e Extração de Dados para usar no modelo:

```python
from transformers import pipeline
from collections import Counter
import json

# === CONFIGURAÇÃO ===
MODEL_NAME = "cardiffnlp/twitter-roberta-base-sentiment-latest"
INPUT_FILE = "pr_comments_2noise_ChatTTS_closed_nobots_True.json"
OUTPUT_FILE = "sentiments_roberta.json"

# === FUNÇÃO DE NORMALIZAÇÃO DE LABEL ===
def normalize_label(label: str) -> str:
    label = label.strip().lower()
    if "neg" in label:
        return "NEGATIVE"
    elif "neu" in label:
        return "NEUTRAL"
    elif "pos" in label:
        return "POSITIVE"
    else:
        return label.upper()

# === CARREGAR MODELO ===
print(f"Carregando modelo {MODEL_NAME}...")
analyzer = pipeline("sentiment-analysis", model=MODEL_NAME)
print("✅ Modelo carregado com sucesso!\n")

# === LER JSON DOS 100 PRs ===
with open(INPUT_FILE, encoding="utf-8") as f:
    data = json.load(f)

# === EXTRAIR TODOS OS COMENTÁRIOS ===
comments = []
for pr in data["prs"]:
    for comment in pr["comments"]:
        body = comment["body"].strip()
        if body:  # ignora comentários vazios
            comments.append({
                "pr_number": pr["number"],
                "user": comment["user"],
                "text": body
            })

print(f"Total de comentários coletados: {len(comments)}\n")
```

Aplicação do modelo nos dados extraídos e coleta de resultados:

```
# === RODAR ANÁLISE DE SENTIMENTOS ===
results = []
for c in comments:
    text = c["text"][:512]
    sentiment = analyzer(text)[0]

    label = normalize_label(sentiment["label"])
    score = round(sentiment["score"], 3)

    results.append({
        "pr_number": c["pr_number"],
        "user": c["user"],
        "text": text,
        "label": label,
        "score": score
    })

# === SALVAR RESULTADOS DETALHADOS ===
with open(OUTPUT_FILE, "w", encoding="utf-8") as f:
    json.dump(results, f, ensure_ascii=False, indent=2)

print(f"✅ Resultados salvos em {OUTPUT_FILE}\n")

# === GERAR RESUMO ESTATÍSTICO ===
counts = Counter([r["label"] for r in results])
total = sum(counts.values())

print("=== 📊 RESUMO DE SENTIMENTOS ===")
for label in ["POSITIVE", "NEUTRAL", "NEGATIVE"]:
    count = counts.get(label, 0)
    print(f"{label:<8}: {count:3} ({count/total:.1%})")

print(f"\nTotal de comentários analisados: {total}")
```

2. BERTweet (finiteautomata/bertweet-base-sentiment-analysis)

Descrição: Baseado no BERT e pré-treinado em bilhões de tweets, o BERTweet é especializado na compreensão de textos curtos e informais com vocabulário técnico misto, emojis e abreviações.

Motivo da Escolha:

- Seu treinamento em dados do Twitter o torna ideal para capturar o estilo de comunicação breve e objetivo frequentemente encontrado em mensagens de pull requests.
- Apresenta desempenho competitivo em tarefas de sentimento no domínio social e técnico.

- Complementa o RoBERTa, permitindo comparar resultados e mitigar vieses de linguagem.

Normalização e Inicialização do Modelo:

```
# Função auxiliar para normalizar labels
def normalize_label(label: str) -> str:
    label = label.strip().lower()
    if "neg" in label:
        return "NEGATIVE"
    elif "neu" in label:
        return "NEUTRAL"
    elif "pos" in label:
        return "POSITIVE"
    else:
        return label.upper()

# Inicialização do modelo
print(f"🚀 Carregando modelo {MODEL_NAME}...")
analyzer = pipeline(
    "sentiment-analysis",
    model=MODEL_NAME,
    tokenizer=MODEL_NAME,
    truncation=True,
    max_length=128
)
print("✅ Modelo carregado com sucesso!\n")

# Leitura do arquivos JSON de PRs
with open(INPUT_FILE, encoding="utf-8") as f:
    data = json.load(f)

comments = []
for pr in data["prs"]:
    for comment in pr["comments"]:
        body = comment["body"].strip()
        if body:  # ignora comentários vazios
            comments.append({
                "pr_number": pr["number"],
                "user": comment["user"],
                "text": body
            })

print(f"💬 Total de comentários coletados: {len(comments)}\n")
```

Análise dos dados e Resultados:

```
# Análise de sentimentos
results = []
for i, c in enumerate(comments, 1):
    text = c["text"]
    try:
        sentiment = analyzer(text, truncation=True, max_length=128)[0]
        label = normalize_label(sentiment["label"])
        score = round(sentiment["score"], 3)

        results.append({
            "pr_number": c["pr_number"],
            "user": c["user"],
            "text": text,
            "label": label,
            "score": score
        })

        if i % 20 == 0:
            print(f"🔎 Processados {i}/{len(comments)} comentários...")

    except Exception as e:
        print(f"⚠️ Erro ao processar comentário do PR #{c['pr_number']}: {e}")

# Salvando resultados
with open(OUTPUT_FILE, "w", encoding="utf-8") as f:
    json.dump(results, f, ensure_ascii=False, indent=2)

print(f"💾 Resultados salvos em {OUTPUT_FILE}\n")

# Gerando resumo dos sentimentos
counts = Counter([r["label"] for r in results])
total = sum(counts.values())

print("--- 📊 Resultados ---")
for label in ["POSITIVE", "NEUTRAL", "NEGATIVE"]:
    count = counts.get(label, 0)
    print(f"{label:<8}: {count:3} ({count/total:.1%})")

print(f"\nTotal de comentários analisados: {total}")
```

3. Multilingual Sentiment Analysis (tabularisai/multilingual-sentiment-analysis)

Descrição: Modelo multilíngue treinado em dados de múltiplos idiomas, capaz de realizar análise de sentimento em mais de 20 línguas.

Motivo da Escolha:

- Garante cobertura para pull requests e comentários escritos em diferentes idiomas, refletindo a natureza global da comunidade de desenvolvimento.
- Atua como suporte aos outros modelos, ampliando a consistência dos resultados em contextos multilíngues.

Normalização e Inicialização do Modelo:

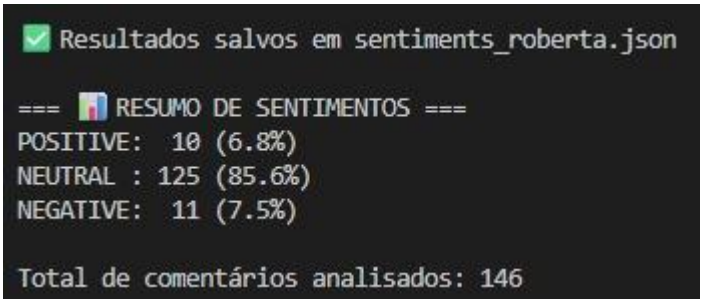
```python
from transformers import pipeline
from collections import Counter
import json

# === CONFIGURAÇÃO ===
MODEL_NAME = "tabularisai/multilingual-sentiment-analysis"
INPUT_FILE = "pr_comments_2noise_ChatTTS_closed_nobots_True.json"
OUTPUT_FILE = "sentiments_multilingual.json"

# === FUNÇÃO DE NORMALIZAÇÃO DE LABEL ===
def normalize_label(label: str) -> str:
    label = label.strip().lower()
    if "neg" in label:
        return "NEGATIVE"
    elif "neu" in label:
        return "NEUTRAL"
    elif "pos" in label:
        return "POSITIVE"
    else:
        return label.upper()

# === CARREGAR MODELO ===
print(f"Carregando modelo {MODEL_NAME}...")
analyzer = pipeline("sentiment-analysis", model=MODEL_NAME)
print("✅ Modelo carregado com sucesso!\n")

# === LER JSON DOS 100 PRs ===
with open(INPUT_FILE, encoding="utf-8") as f:
    data = json.load(f)

# === EXTRAIR TODOS OS COMENTÁRIOS ===
comments = []
for pr in data["prs"]:
    for comment in pr["comments"]:
        body = comment["body"].strip()
        if body:  # ignora comentários vazios
            comments.append({
                "pr_number": pr["number"],
                "user": comment["user"],
                "text": body
            })

print(f"Total de comentários coletados: {len(comments)}\n")
```

Análise e Resultados:

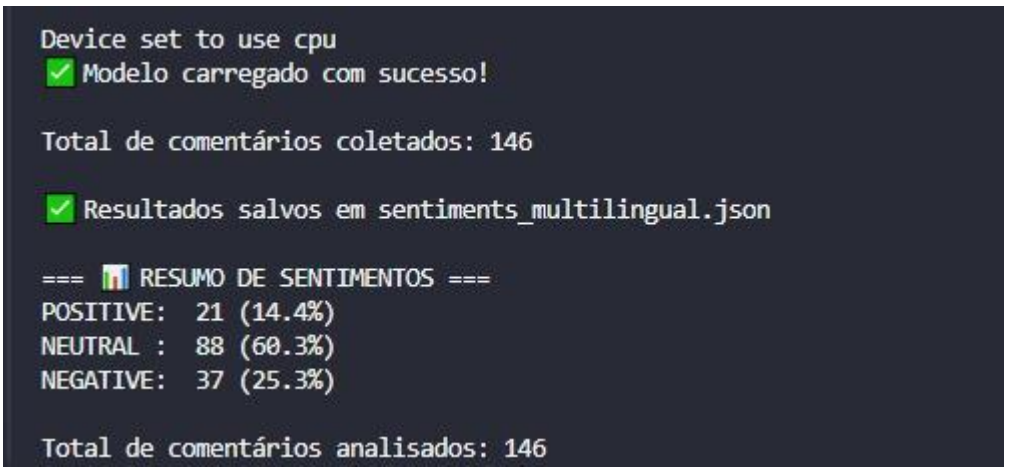
```python
# === RODAR ANÁLISE DE SENTIMENTOS ===
results = []
for c in comments:
    text = c["text"][:512]
    sentiment = analyzer(text)[0]

    label = normalize_label(sentiment["label"])
    score = round(sentiment["score"], 3)

    results.append({
        "pr_number": c["pr_number"],
        "user": c["user"],
        "text": text,
        "label": label,
        "score": score
    })

# === SALVAR RESULTADOS DETALHADOS ===
with open(OUTPUT_FILE, "w", encoding="utf-8") as f:
    json.dump(results, f, ensure_ascii=False, indent=2)

print(f"✅ Resultados salvos em {OUTPUT_FILE}\n")

# === GERAR RESUMO ESTATÍSTICO ===
counts = Counter([r["label"] for r in results])
total = sum(counts.values())

print("=== 📊 RESUMO DE SENTIMENTOS ===")
for label in ["POSITIVE", "NEUTRAL", "NEGATIVE"]:
    count = counts.get(label, 0)
    print(f"{label:<8}: {count:3} ({count/total:.1%})")

print(f"\nTotal de comentários analisados: {total}")
```

3. Resultados e Comparações

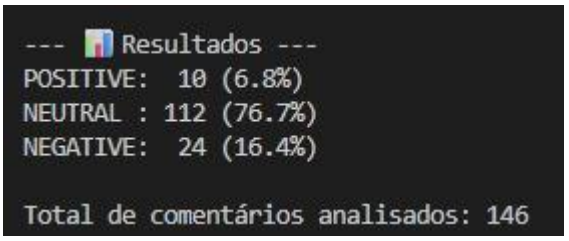
Após configurar e rodar os códigos de cada modelo, analisamos os 3 .json gerados e os seus resumos quantificados da análise de sentimentos:



Esse resultado acima é do RoBERTa base sentiment. Ele trouxe a análise com um número de neutros elevados para classificar os pull resquest.



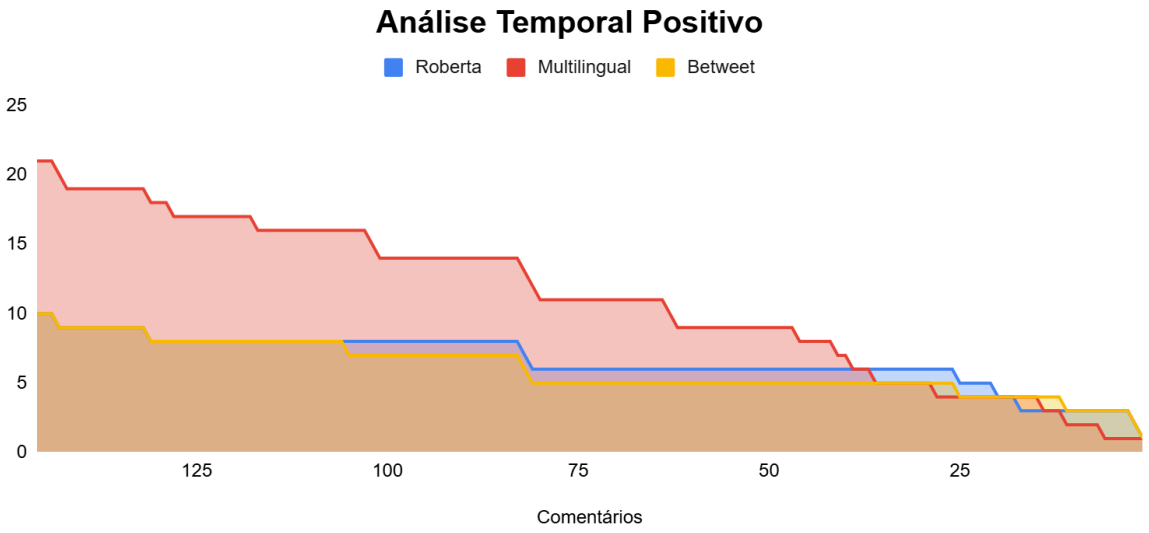
Esse resultado acima é do Multilingual Sentiment Analysis. Assim, trazendo uma perspectiva de comparação entre os três modelos, ele foi o modelo mais discrepante, trazendo elevada taxa de Positivos e Negativos, sendo que, na maioria dos casos, os comentários deveriam ter sido analisados como Neutros, já que não continham sentimentos expressos.

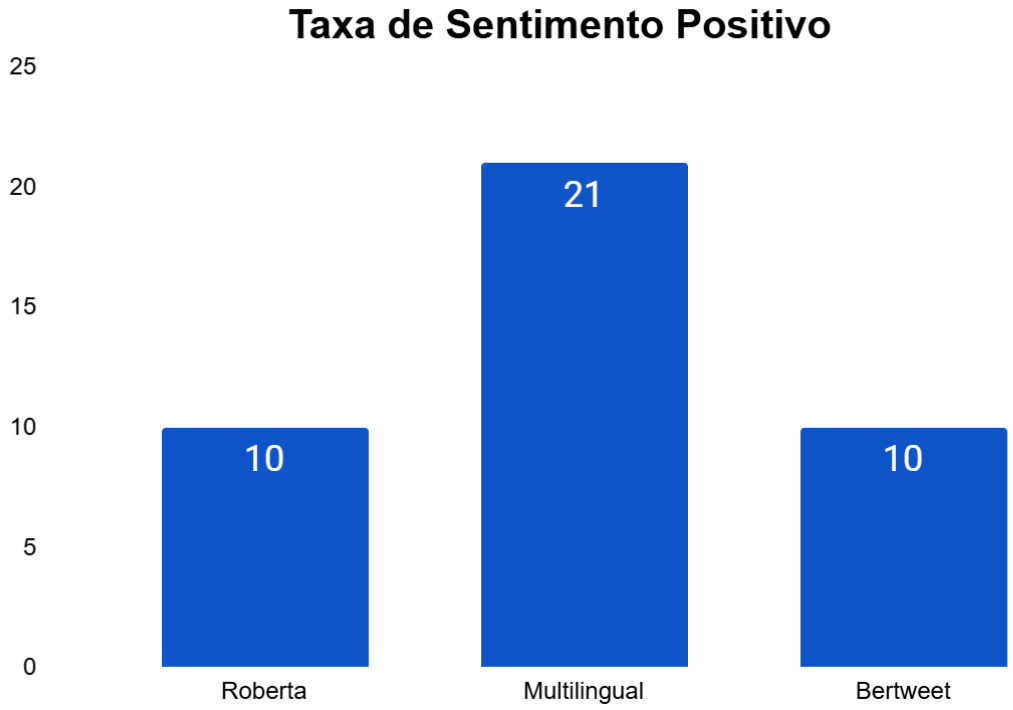


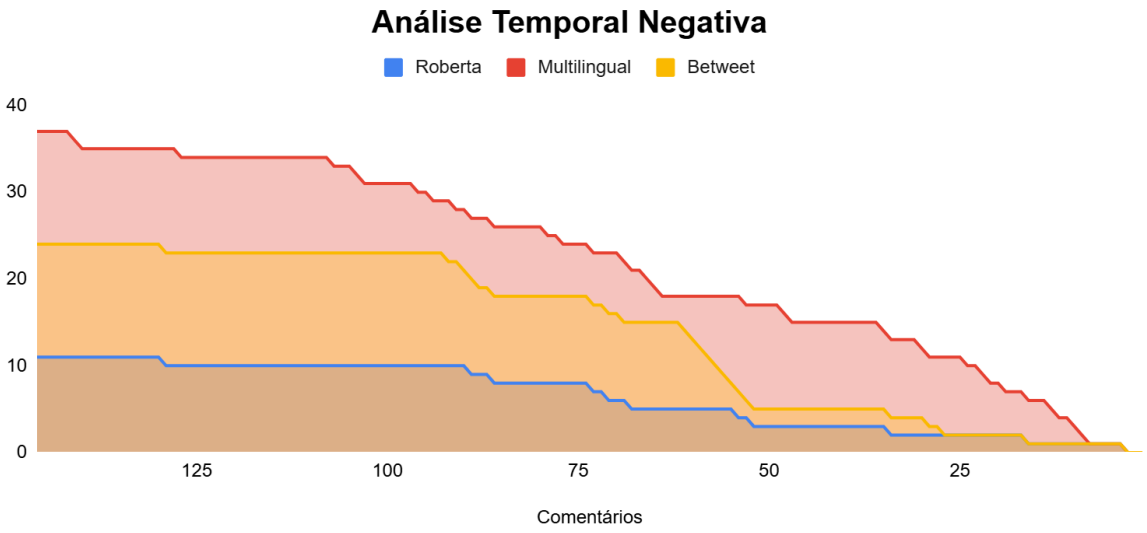
Esse resultado acima é do BERTweet Sentimental Analyse. Por fim, foi testado o BERTweet que é comparado ao resultado do RoBERTa, apenas tendo um número menor de neutrals e compensado no aumento dos pull requests negativo.

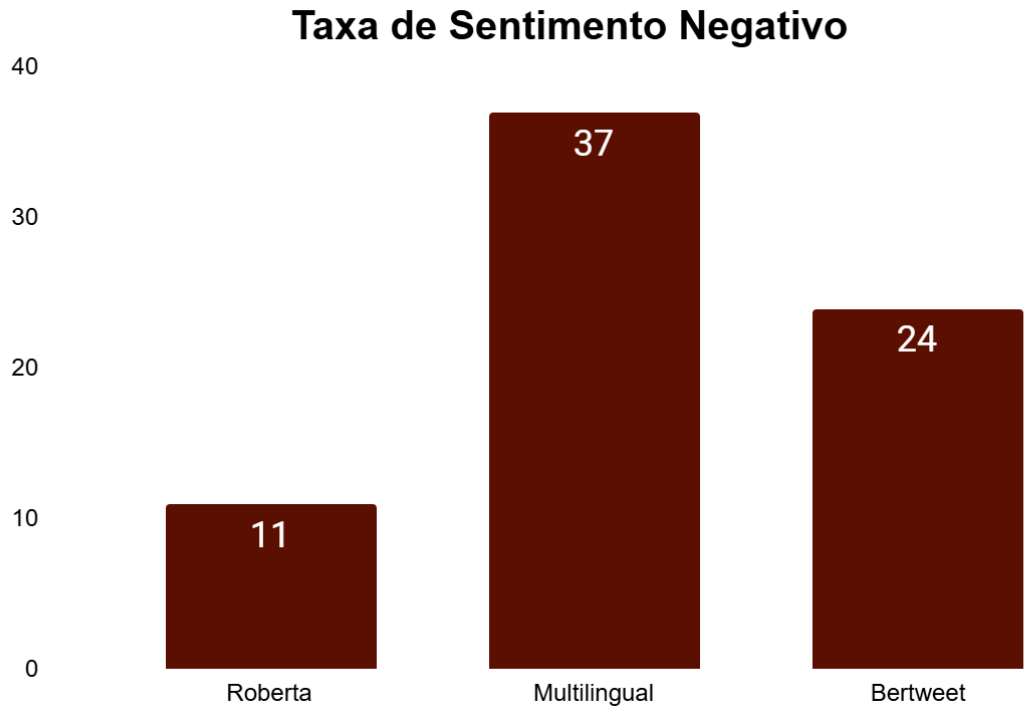
Podemos observar abaixo os dados analisados através de gráficos e tabelas referentes às 3 comparações informadas acima.

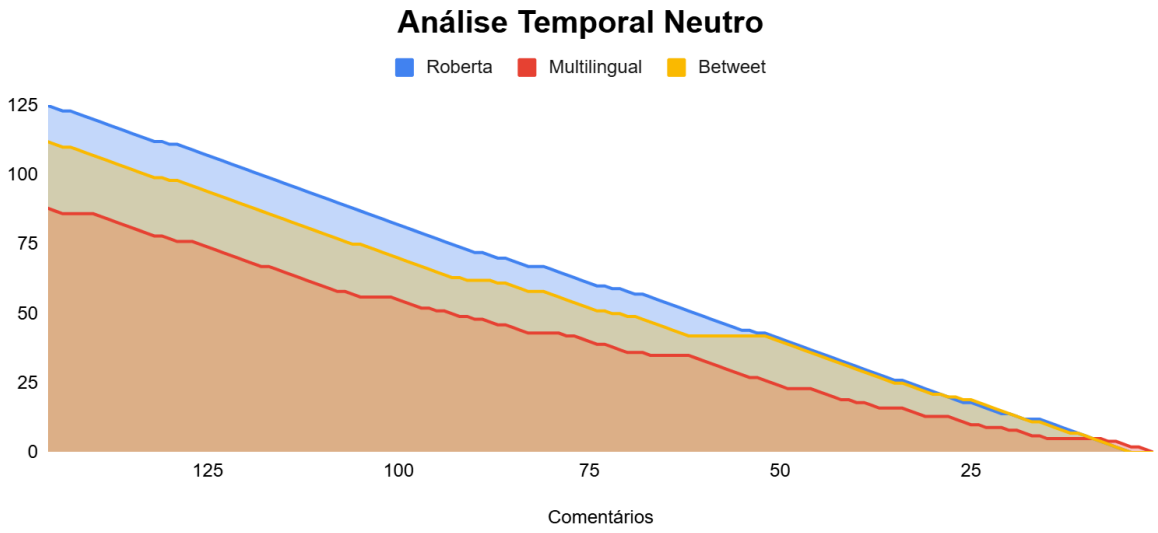
É apresentada uma análise temporal de sentimentos positivos, negativos e neutros referentes aos Commits.

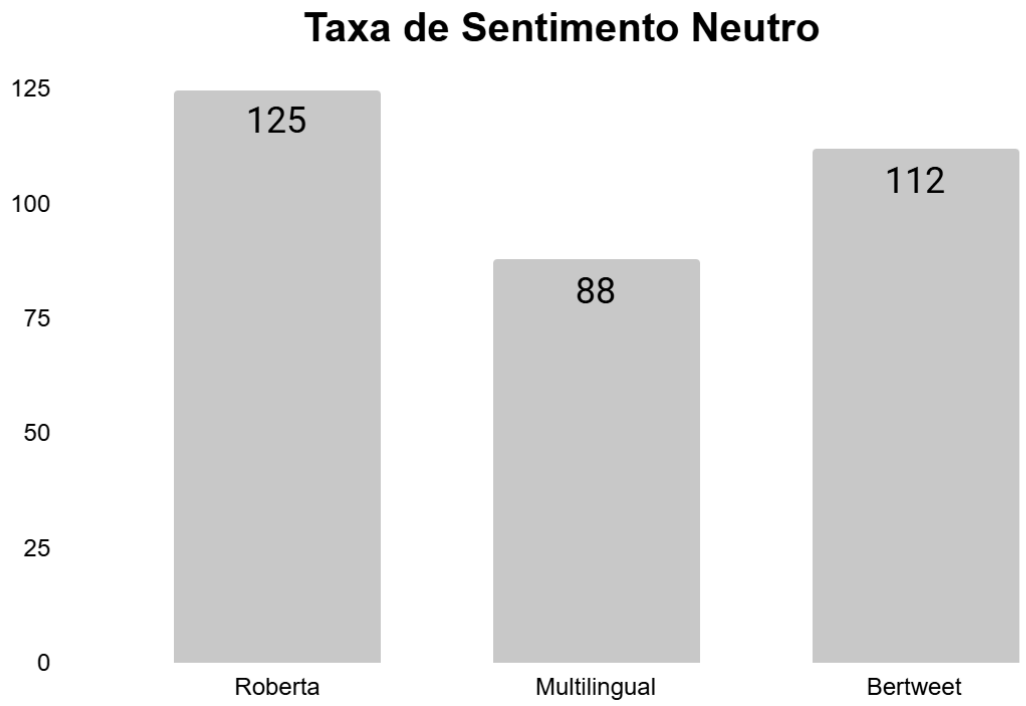

Análise Temporal Positivo
Roberta
Multilingual
Betweet
25
20
15
10
5
0
125
100
75
50
25
Comentários


Taxa de Sentimento Positivo
25
20
15
10
5
0
10
21
10
Roberta
Multilingual
Bertweet


Análise Temporal Negativa
Roberta
Multilingual
Betweet
40
30
20
10
0
125
100
75
50
25
Comentários


Taxa de Sentimento Negativo
40
30
20
10
0
11
37
24
Roberta
Multilingual
Bertweet


Análise Temporal Neutro
Roberta
Multilingual
Betweet
125
100
75
50
25
0
125
100
75
50
25
Comentários

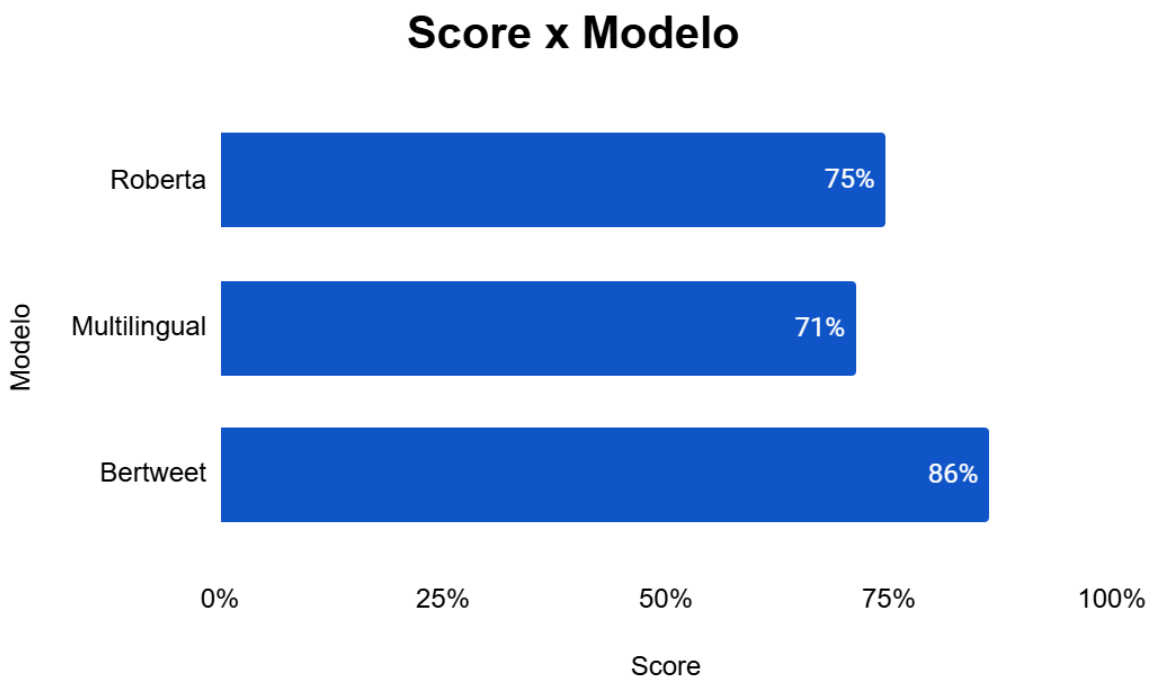

Taxa de Sentimento Neutro
125
100
75
50
25
0
125
88
112
Roberta
Multilingual
Bertweet

Abaixo podemos observar a análise de score

Análise de Score			
	Positivo	**Negativo**	**Neutro**
Roberta	80,93%	64,68%	75,03%
Multilingual	63,91%	63,72%	76,22%
Bertweet	85,23%	79,13%	87,85%

Dados sobre a análise

Qtd. Usuários	Qtd. PR	Qtd. Comentários
26	100	146



Abaixo podemos ver a quantidade de Comentários dos usuários referente ao modelo **Roberta:**

Usuário	Negativo	Neutro	Positivo	Total
AdamMayor2018		1		1
ai408		1		1
ain-soph	2	7	1	10
BBC-Esq	1	3		4
charSLee013			1	1
CJY1018		1		1
fengyizhu		5		5
fumiama	8	77	2	87
Gloridust		1	1	2
IrisSally		1		1
ivanmilevtues			2	2
JaysonAlbert		1		1
LLongIsland		1		1
MengqingCao		1		1
niuzheng168		2		2
Ox0400		4		4
pengooseDev			1	1
quanqigu		1		1
shen-shanshan		2	2	4
weedge		2		2
wenyangchou		1		1
xiaohua-drive		2		2
ylzz1997		1		1
yueguobin		2		2
ZaymeShaw		6		6
ZillaRU		2		2
Total	**11**	**125**	**10**	**146**

Abaixo podemos ver a quantidade de Comentários dos usuários referente ao modelo **Multilingual:**

Usuário	Negativo	Neutro	Positivo	Total
AdamMayor2018	1			1
ai408	1			1
ain-soph	2	6	2	10
BBC-Esq	3	1		4
charSLee013			1	1
CJY1018		1		1
fengyizhu		3	2	5
fumiama	24	56	7	87
Gloridust			2	2
IrisSally			1	1
ivanmilevtues		1	1	2
JaysonAlbert			1	1
LLongIsland			1	1
MengqingCao		1		1
niuzheng168		2		2
Ox0400		4		4
pengooseDev		1		1
quanqigu		1		1
shen-shanshan	1	2	1	4
weedge		2		2
wenyangchou		1		1
xiaohua-drive	1		1	2
ylzz1997		1		1
yueguobin	1	1		2
ZaymeShaw	2	3	1	6
ZillaRU	1	1		2
Total	**37**	**88**	**21**	**146**

Abaixo podemos ver a quantidade de Comentários dos usuários referente ao modelo **Bertweet:**

Usuário	Negativo	Neutro	Positivo	Total
AdamMayor2018		1		1
ai408		1		1
ain-soph	2	7	1	10
BBC-Esq	1	3		4
charSLee013			1	1
CJY1018		1		1
fengyizhu		5		5
fumiama	18	66	3	87
Gloridust		1	1	2
IrisSally		1		1
ivanmilevtues			2	2
JaysonAlbert			1	1
LLongIsland		1		1
MengqingCao	1			1
niuzheng168		2		2
Ox0400		4		4
pengooseDev			1	1
quanqigu		1		1
shen-shanshan		4		4
weedge	1	1		2
wenyangchou	1			1
xiaohua-drive		2		2
ylzz1997		1		1
yueguobin		2		2
ZaymeShaw		6		6
ZillaRU		2		2
Total	**24**	**112**	**10**	**146**

Efetividade dos Modelos de Sentimento

Tendo em vista os resultados obtidos, nota-se que o modelo BERTweet foi o mais efetivo na análise de sentimentos, uma vez que ele demonstrou uma maior sensibilidade e capacidade de identificar tanto sentimentos positivos quanto negativos presentes nos comentários das pull requests. Além disso, ele apresentou o maior score médio em comparação aos outros dois modelos, indicando uma maior confiança nas classificações.

Em contrapartida, no modelo RoBERTa houve uma predominância de classificações neutras. Isso se deve ao fato de que, devido ao seu treinamento em linguagem emocional, como tweets, o modelo tende a interpretar como neutros os textos que são mais objetivos ou técnicos e que não apresentam expressões explícitas de sentimentos, possuindo, assim, dificuldade em detectar sentimentos positivos ou negativos.

Já o Multilingual Sentiment Analysis apresentou resultados mais equilibrados entre as classes, porém seu desempenho geral foi inferior, atuando melhor como apoio para garantir consistência em comentários multilíngues.

Impacto na evolução do Projeto ChatTTS

De modo geral, houve uma predominância de sentimentos neutros e positivos nas pull requests, o que demonstra que o projeto mantém um ambiente colaborativo e respeitoso entre os desenvolvedores, indicando que há uma comunicação saudável entre os membros da comunidade, o que colabora para o progresso contínuo do projeto. Além disso, a baixa presença de comentários negativos mostra que as interações ocorrem de forma construtiva, favorecendo a resolução de problemas e a evolução do projeto. Nesse cenário, percebe-se que o projeto vem evoluindo ao longo do tempo de maneira estável e organizada e com um engajamento constante da comunidade.

4. Conclusão

A execução deste estudo permitiu compreender de forma prática como modelos de linguagem podem ser aplicados para análise de sentimentos em projetos reais de software. A partir da coleta e organização dos 146 comentários extraídos de 100 *pull requests* do repositório ChatTTS, foi possível observar o comportamento da comunidade de desenvolvedores e o clima colaborativo do projeto.

Os três modelos testados apresentaram resultados consistentes, mas com variações importantes: o RoBERTa mostrou maior tendência a classificar comentários como neutros, o BERTweet teve uma sensibilidade maior a críticas e emoções negativas, enquanto o Multilingual Sentiment Analysis demonstrou equilíbrio e boa adaptação a comentários em diferentes idiomas. Essa diversidade de resultados reforça a importância de combinar múltiplos modelos para uma análise mais confiável e imparcial.

De modo geral, a maioria dos comentários analisados apresentou um tom neutro ou positivo, indicando que o projeto ChatTTS mantém um ambiente colaborativo e respeitoso entre seus contribuidores. A aplicação prática dos modelos da plataforma Hugging Face também evidenciou a relevância das técnicas de *Natural Language Processing (NLP)* no contexto da Engenharia de Software, especialmente para compreender dinâmicas de equipes e a evolução social de projetos open source.