※ 菅原のソフトウェア開発能力を示す参考資料です
　現在の職種が製造業であるため、ソフトウェア開発能力を示す資料があった方が望ましいと感じましたので、勝手ながら添付させていただきました。

テキスト解析を用いた
感情的評価プログラムの開発

菅原　慶真

0. 概要

YouTube のライブチャットへの書き込みを利用し、時間軸方向について感情的評価を行う Python プログラムを開発しました。

このプログラムの開発において、私は以下のような開発を行っています:

・Web からのデータ取得（pytchat を用いたスクレイピング）
・テキストデータからの感情的評価値の取得（pymlask を用いたテキスト解析）
・時間軸方向の評価値算出（（簡単ですが）アルゴリズム開発および実装）
・算出した評価値のグラフ化（pandas を用いた matpltlib へのエクスポート機能の実装）

1. はじめに

御社への応募にあたり、私のプログラミング能力を説明するため、新規にプログラムを作成いたしました。AI による CRM についても言及されておりましたので、Web からのテキストデータ取得、取得したテキストデータの解析、解析結果の可視化に関するものを製作してみました。
このプログラムでは Youtube Live の配信アーカイブからスクレイピングしたチャットのテキストデータを感情的評価として分類して、時間軸方向で集計したデータを60秒ごとに区切り、グラフへプロットしています。

2. 処理について

2.1 Web からのテキストデータ取得について

Youtube Live の配信アーカイブより、チャットデータをスクレイピングしました。これには pytchat を使用しています。pytchat は YouTube チャットを閲覧するための Python ライブラリで、チャットのメッセージテキスト、書き込みをしたユーザー名、スーパーチャットの金額、個別 id 等のデータを取得できます。データの保存のために次に示すプログラムを作成しました。

```
import pytchat
import time

livechat = pytchat.create(video_id = " Youtubeの動画ID ")
filepath = "取得したデータ保存先のパス"

with open(filepath, mode='a') as f:
    while livechat.is_alive():
        chatdata = livechat.get()
        for c in chatdata.items:
            f.write(f"{c.datetime},{c.author.name},{c.message}\n")
        time.sleep(5)
```

本プログラムでは、次項に示す感情的評価値の評価のために、投稿時間、投稿者名、チャットのテキストをスクレイピングしております。ローカル環境での作業性向上とサーバーへの負荷を考慮し、一度CSVとして書き出しています。

2.2 テキストデータからの感情的評価値の取得

取得したチャットテキストから感情的評価値を取得するために、前処理として pandas を使ってデータの整形を行いました。

```
pd.set_option('display.max_rows', None)
filepath = " CSVとしてスクレイピングしたチャットデータのパス "

with open(filepath, mode="r", encoding="UTF-8", errors="",\
          newline="") as f:
    lst = csv.reader(f, delimiter=",")
    df = pd.DataFrame(lst, columns=['datetime', 'author_name',\
                                    'm1', 'm2', 'm3', 'm4'])


df["message"] =
df["m1"]+df["m2"].fillna("")+df["m3"].fillna("")+df["m4"].fillna("")
df.drop(['m1', 'm2', 'm3', 'm4'], axis=1, inplace=True)
```

コメントが一部 delimliter と誤認識されるため、一度データ全体を DataFrame とした後必要な部分を message カラムにまとめています。

文章の感情的評価には pymlask を利用しております。pymlask は文中の単語を感情表現辞典に基づいて、文章の感情(喜、怒、哀、怖、恥、好、厭、昂、安、驚)を推定します。また、推定された感情から文章をポジティブ(positive, mostly_positive)、ニュートラル、ネガティブ(negative, mostly_negative)の三種類に分類します。

```
def try_emotion_perse(m, emoji_prefix=""):
```

```
    """
    動画にカスタム絵文字が使用されている場合,emoji_prefixにカスタム絵文字の接頭辞を渡す
    ex "_kizunaai"
    """
    e = emo(m)
    if emoji_prefix in e['text']: # カスタム絵文字
        emoji_message = re.findall(r':(.*?):', e['text'])
        category = emoji_message[0].replace(emoji_prefix, '')
        orien, repre = custom_emoji[category]['orientation'], custom_emoji[category]['representative']

    else:
        try:
            orien = e['orientation']
        except KeyError:
            return None, None
        ekey = list(e['emotion'].keys())
        repre = ekey[0]

    return orien, repre
```

作成した DataFrame は try_emotion_perse() 関数で感情的評価を行います。コメント中のカスタム絵文字については pymlask ではパースできないため、動画に使用されるカスタム絵文字へ自分でラベル付けを行い、感情的評価およびポジティブ/ネガティブ評価を行いました。

```
custom_emoji = {'Hai':{'orientation': 'NEUTRAL', \
                       'representative':'None'},\
                'Doumo':{'orientation': 'NEUTRAL',\
                         'representative':'None'},\
                'Yabami':{'orientation': 'POSITIVE',\
                          'representative':'takaburi'},\
                    .
                     .
```

2.3 時間軸方向の評価値算出

集計した感情的評価結果を計算し、時間軸方向（1分毎）に合計しています。positiveとnegativeにはそれぞれmostly（より強い）が存在するので、得られたデータは(mostly_postive * 10 + positive) - (mostly_negative * 10 + negative)として集計し、これを感情ポイント（emotional_point）として集計しました。

```
emolist = []
for pos, neu, neg, most_pos, most_neg in zip(total_orien['POSITIVE'],
total_orien['NEUTRAL'], total_orien['NEGATIVE'],
total_orien['mostly_POSITIVE'], total_orien['mostly_NEGATIVE']):
    point = (most_pos * 10 + pos) - (most_neg * 10 + neg)
    emolist.append(point)
```

2.4 算出した評価値のグラフ化

時間軸方向で区切った emotional_point を Series へ変換して、折れ線グラフへプロットしました。

```
emotional_point = pd.Series(emolist)

emotional_point.plot()
plt.title('POSITIVE/NEGATIVE評価')
plt.xlabel('再生時間(min)')
plt.ylabel('感情ポイント(point)')
plt.savefig(f'orien_{ch_id}.png')
```

3. 実際の解析結果

3.1 対象とした動画

本プログラムは、kizuna AI [注1)]「キズナアイから大事なお知らせがあります」及び月ノ美兎 [注2)]「公園の地下に巨大神殿があるらしいので行ってみた【にじさんじ/月ノ美兎】」について解析を行いました。

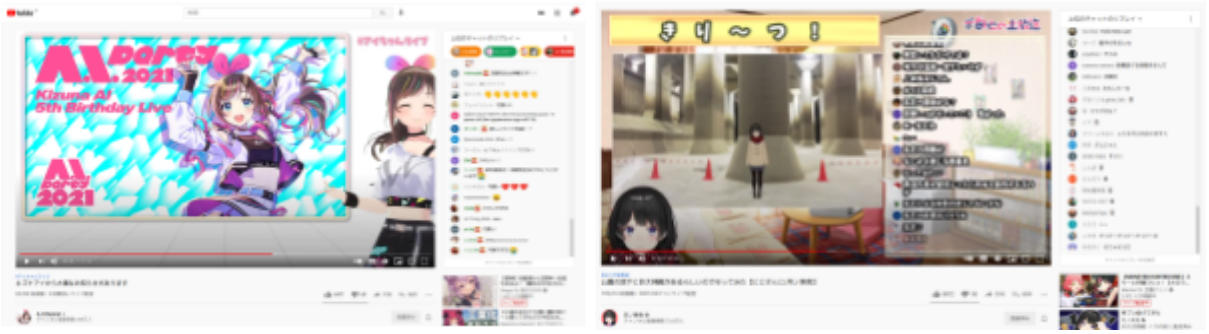

fig.1 使用した動画のキャプチャ

注1）Kizuna AI株式会社に所属している世界初のバーチャルYouTuber。
　　メイン、サブチャンネル合計で登録者数400万人。
注2）いちから株式会社が運営するにじさんじに所属するバーチャルライバー、
　　バーチャルアイドル。チャンネル登録者数 70 万人。

3.2 解析結果のプロット

kizuna AI の配信動画の解析結果を以下に記します。

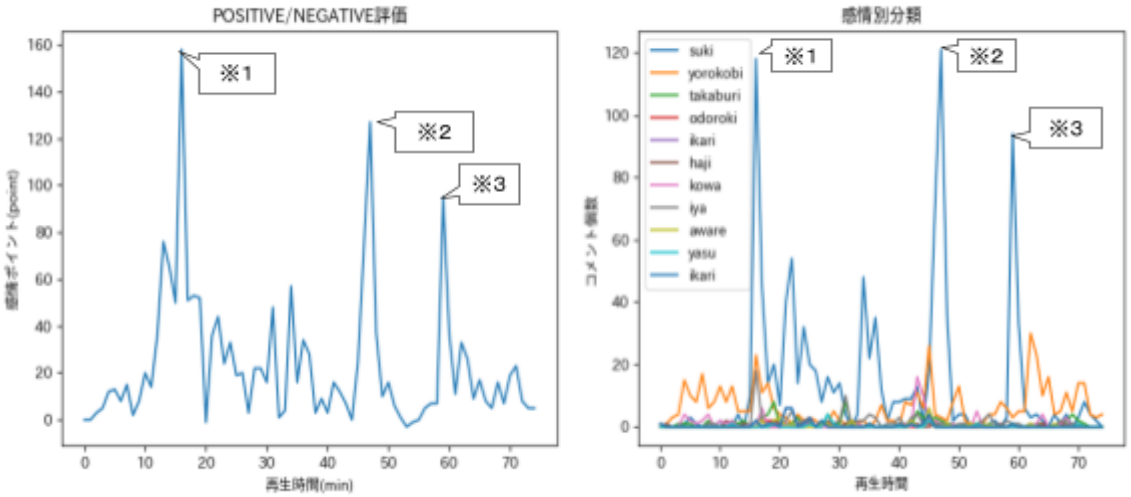

fig.2 kizuna AI 配信動画の感情的評価結果

※1　配信開始より 16 分経過：感情ポイント：+158
　　→ゲームがクリアできたことをリスナーに報告して褒めてもらった。
※2　配信開始より 47 分経過：感情ポイント： +127
　　→ライブについて配信だけでなく、リアル会場での開催もあると報告した。
※3　配信開始より 59 分経過：感情ポイント：+95
　　→ 4 ～ 6 月のライブスケジュールについて報告した。

月ノ美兎の配信動画の解析結果を以下に記します。

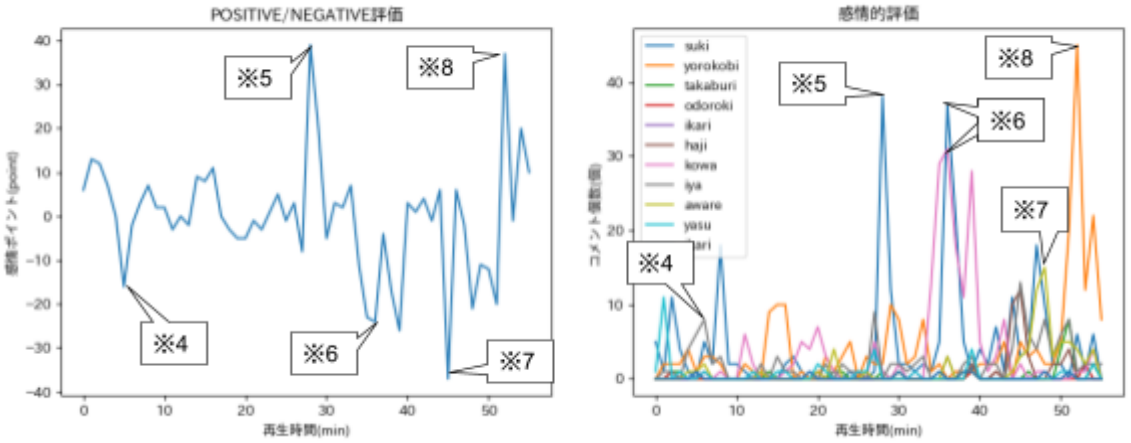

fig.3 月ノ美兎 配信動画の感情的評価結果

※4　配信開始より5分経過：感情ポイント -16
　　→ティーカップのソーサーを醤油皿代わりに使用したことをリスナーに諫められた。
※5　配信開始より28分経過：感情ポイント+39
　　→柱の前で記念撮影した写真を配信に出してリスナーから褒められた。
※6　配信開始より36分経過：感情ポイント -24
　　→あまり共感されない、自分が怖いものの話をした。
※7　配信開始45分より経過：感情ポイント -37
　　→位置情報ゲームアプリ「ジオキャッシング」の体験レポ。
　　ゲーム内のコンテナが見つからなくて恥ずかしい思いをしたという話をした。
※8　配信開始より52分経過：感情ポイント＋37
　　→ジオキャッシュのコンテナを最終的に発見できた話をした。

4. 結果と考察

4.1 結果

kizuna AI の解析結果について感情的評価を行った結果、概ね「suki」、「yorokobi」であり、それ以外の感情はほぼ見られません。また、POSITIVE / NEGATIVE 分類のグラフからもネガティブな意見はほぼ無く概ねポジティブなコメントが多いことが集計結果から読み取れます。月ノ美兎の感情的評価については「suki」や「yorokobi」といったポジティブな感情が見られる一方で、「iya」や「kowa」、「aware」等の kizuna AI の解析結果では見られない様々な感情のピークが見られました。それに伴い、POSITIVE / NEGATIVE 分類のグラフにも激しい変動が見られました。

4.2 考察

実際に自分で解析した両動画を初めから視聴し、ピークとなる部分を確認してみたところ実際に盛り上がっている部分とグラフのピークが一致していると感じられました。kizuna AI の解析に使用した動画についてはバースデイライブの情報初出配信であり、本プログラムで解析できたコメントは「かわいい！」「最高！」等のほぼポジティブな文章でした。そのため、テスト用の動画としては不十分であった可能性があります。

追加で解析した月ノ美兎の動画については POSITIVE / NEGATIVE 分類のグラフ激しい変動が見られました。例えば配信開始から5分経過したところ等です。(fig.3 ※4) 作成したプログラムでは、ここで感情の評価値が初めてマイナスになります。

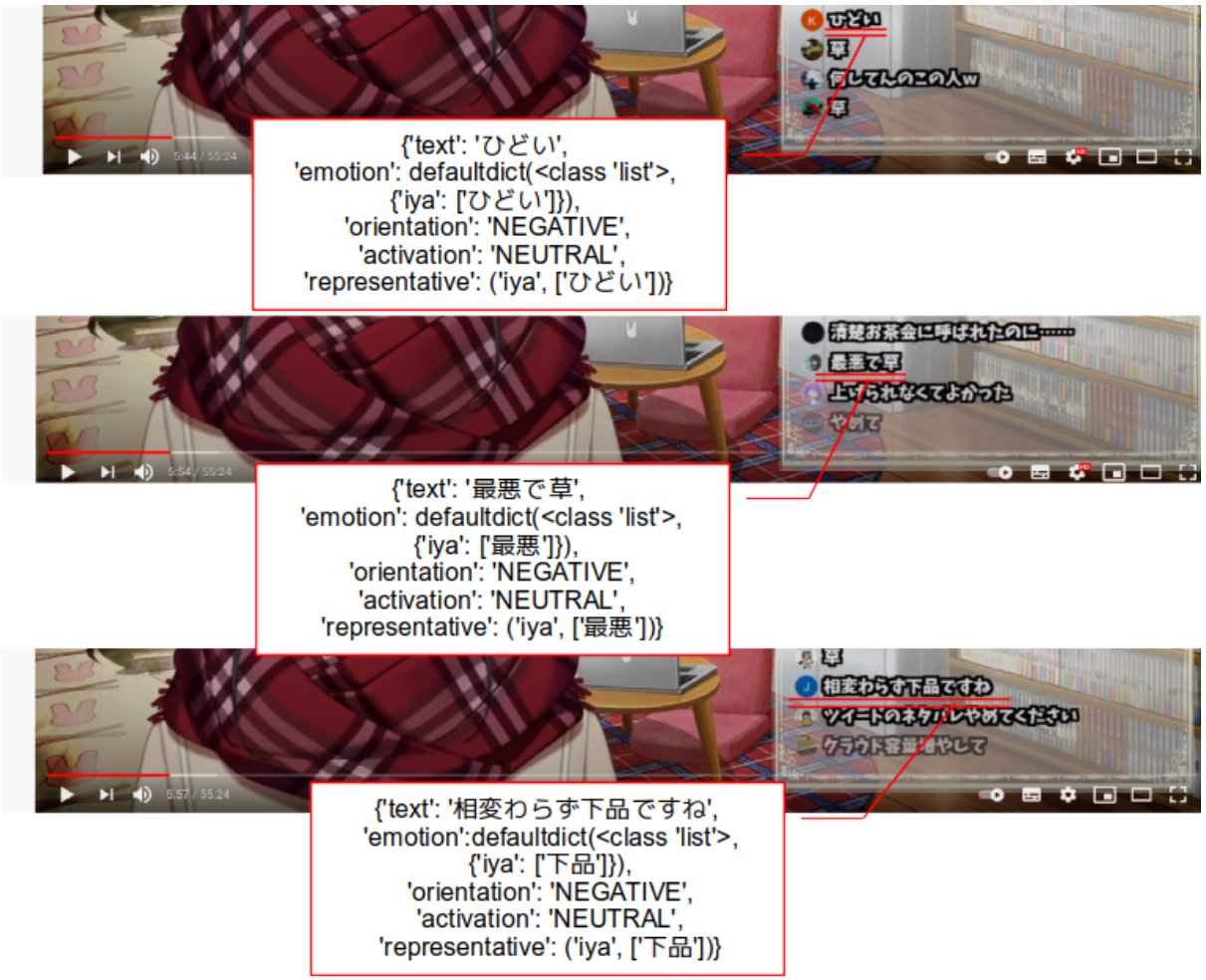

fig.4 配信開始5分時点において NEGATIVE と判定されたコメントの例

これについては配信内で、「ティーカップのソーサーを醤油皿代わりに使用したことをツイートしようとした際に、icloudの容量がいっぱいになってしまい、写真の画像が上げられなくてツイートできなかった」というエピソードを話した時間と一致しており、リスナーからネガティブな反応があったためであると考えられます。

また、配信開始28分時点(fig.3 ※5)では、動画内で感情の評価値が最もプラスになっています。これは、柱の前で記念撮影した写真を配信に出した際にリスナーからポジティブな反応があったためであると考えられます。コメントの一部を分類した結果を次へ記します。

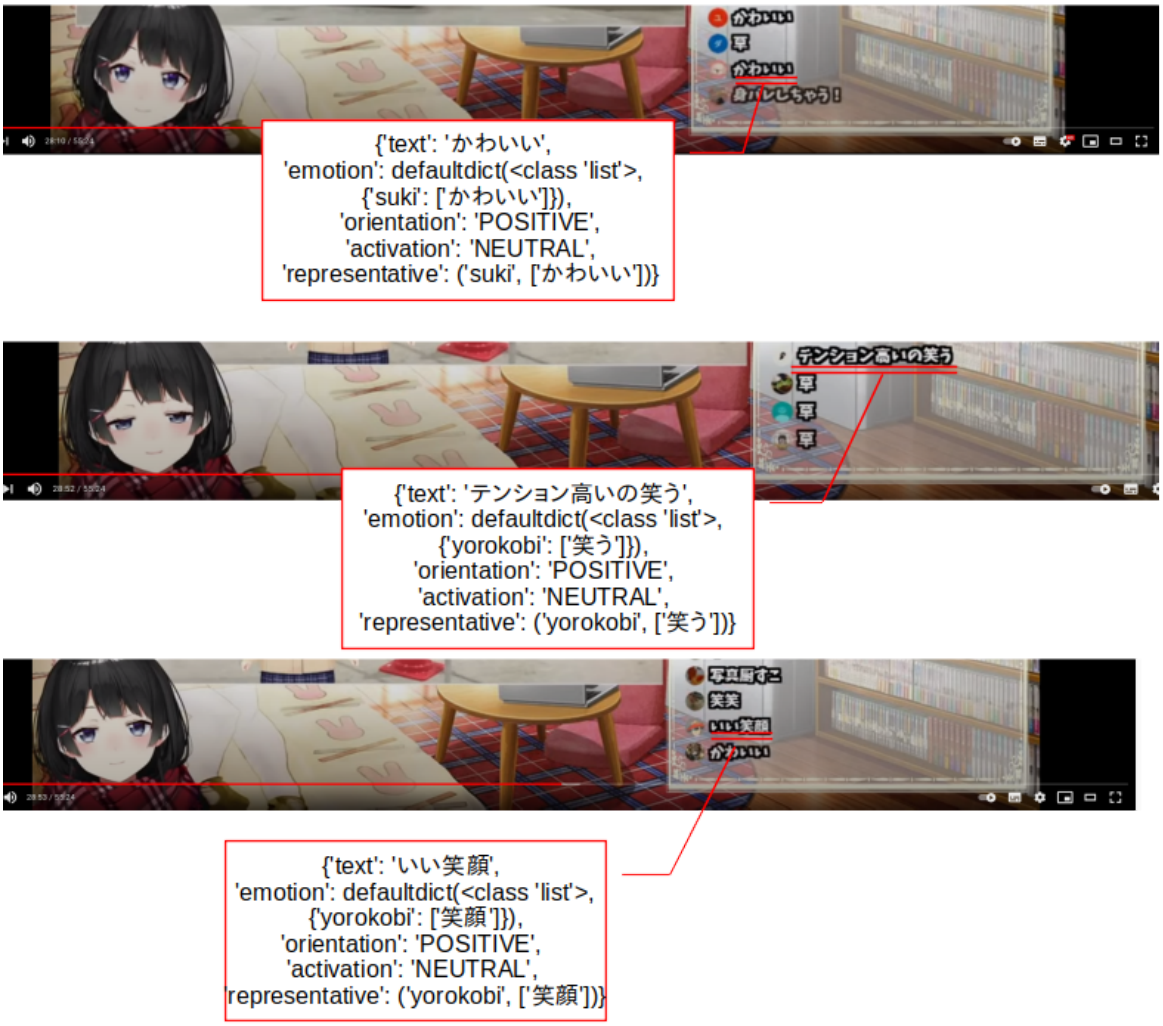

fig 5 配信開始28分時点で POSITIVE と判定されたコメントの例

このことからリスナーの感情をで読み取ることができていると判断します。

5. まとめ

ML-Ask での感情分類では、「かわいい！」や「楽しかった」といった単語は正しくPOSITIVE として分類される一方、解析アルゴリズムのベースとなっている感情表現辞書に 登録されていない「すごい！」や「えらい！」等のポジティブに感じられる単語でも None (感情なし)として分類されてしまいました。fasttext を使用すると作成した感情辞書(極性辞書)を使用できるようなので、解析精度を向上させるために今後fasttextの使用も検討したいと考えます。

6.製作したプログラムのソースコード

・当資料にて説明した自作プログラム

github リポジトリ https://github.com/raveman179/youtubelive_emotional_analysis

・フォルダ構成

/dataplot : チャットのメッセージテキスト解析結果のプロット。

/livechat_log : 解析に使用したYouTube Live の配信アーカイブよりテキスト形式で
スクレイピングしたライブチャットの生データ。

get_youtube_livechat.py
: pytchat を利用して、動画IDを渡すとライブチャットから、チャットの
投稿時間、投稿者、メッセージテキストを取得して、テキストファイル
に書き出すプログラム。

emotional_analyse.py
: get_youtube_livechatで作成したテキストファイルをpandasで整形後、
メッセージテキストをML-Askで感情的表現として解析。
その後、得られたデータを時間軸方向で集計して、matplotlibで折れ線
グラフとしてプロットするプログラム。

参考文献・資料

・当資料にて使用した動画

kizuna AI https://www.youtube.com/watch?v=zCK_490ryjg
月ノ美兎 https://www.youtube.com/watch?v=I1wqOXlXHp8

・ライブラリについて

pytchat https://github.com/taizan-hokuto/pytchat/wiki/Home_jp
pymlask https://github.com/ikegami-yukino/pymlask