

作業内容の実況に対する応援コメントの自動生成によるタスク遂行促進手法の提案 A Task Execution Promotion Method by Automatic Generation of Supportive Comments for Live Commentary of Work Contents

堀川 達平¹⁾北山 大輔¹⁾

Tappei Horikawa Daisuke Kitayama

1 はじめに

我々は様々なタスクを抱えて生活しているが、時には集中力の低下やモチベーションの喪失により、作業を円滑に進められない場合がある。そのような中、近年 YouTube などの動画配信プラットフォームにおいて、作業配信と呼ばれる自身の作業状況を実況しつつリスナーとコメントのやり取りを行い作業を進めていく配信が行われている。

この作業配信は、作業内容を実況するという自己説明による課題解決促進効果と、応援やアドバイスといったリスナーからのリアクションによる発話促進効果という2つの効果の相乗効果によりタスク遂行が促進されていると考えることができる。他者からのリアクションが存在する場合自己説明の理解が促進されるという研究結果があることから、作業配信の効果が伺える [1]。

既存の発話を促進させタスク遂行を促進する試みでは、相槌や質問を行うキャラクタ「ベアプロくん」を画面に表示するという研究がある [2]。この研究ではシステムの利用が発話促進効果を与えることを示唆したが、遂行促進効果については不明であった。「ベアプロくん」はキャラクタを画面上に表示し音声刺激を与えるというシステムであったが、リスナーを自身とは異なる人物という意味でキャラクタと捉えれば、作業配信は、キャラクタを画面上に表示し、テキストによる視覚刺激を与えるものであると考えることができる。

このような利点のある作業配信だが、これを行うには自身の行動をインターネット上に配信する必要がある。機密性が高い作業を行うことができない。また、ある程度の規模のリスナーを獲得する必要がある、実施の際の障壁となっている。

そこで本研究では、ユーザの発話や操作に反応し、作業の応援やアドバイスを行うコメントを自動生成する手法を提案する。

2 提案手法

本研究の提案手法は、ユーザが発話した作業内容の実況に基づき、大規模言語モデルを活用してコメントを生成し、それを表示することにより実現される。音声認識には Google Speech-to-Text API¹⁾ を、コメント生成には OpenAI Chat Completions API²⁾ を通じて gpt3.5-turbo を利用する。

Chat Completions API では大規模言語モデルと自然言語でのやりとりが可能となっており、モデルのふるまいを指定する system プロンプト、ユーザからの入力を表す user プロンプト、モデルの出力を表す assistant プロンプトでやりとりが行われる。

生成に使用したプロンプトは以下の通りである。これを system プロンプトとし、user プロンプトに音声の

文字起こしを行ったテキストを直接入力することで、assistant が json 形式で複数のコメントを返すことが可能となっている。

作業を行う生放送の文字起こしに対し、応援やアドバイスを行うコメントを複数生成してください。コメントはなるべく短く、簡潔であることが望ましいです。以下の形式で出力してください。

```
{
  "comments": [
  ]
}
```

Speech-to-Text API は音声片からテキストを自動的に認識し、最良の予測結果を返す。そのため、API より予測結果が返されたタイミングで user プロンプトの入力が行われ、コメント生成が実行される。ここで、user プロンプトとそれに対する assistant プロンプトの履歴を同時に入力すると、モデルが発話の文脈を認識することが可能となる。なお、本稿では文脈の認識を考慮していない。

本提案では、無発話時に発言を促すコメントを生成する試みも行っている。発話の有無を判断する無音検出にはゼロ交差数を用いており、録音機器より取得された 100ms 相当の音声信号 1 チャンクにおいて、ゼロ交差数が信号点数の 15% 未満であれば無音であると判定している。この無音と判定されたチャンクが 100 個連続した場合、つまり発話が 10 秒間検出されなかった場合に発言を促すコメント生成を行う。生成の際は特別なプロンプトは使用せず、通常の文字起こし結果に代わって無言を表す「……」を user プロンプトとすることで、発話のきっかけとなるコメントが生成されることを狙う。

作業時の画面構成を図 1 に示す。図のように、画像右側のチャットウィンドウに思考発話の内容と生成されたコメントが表示される。本システムでは発話内容の認識結果を緑色、コメントを白色として表示を行っている。このとき、1 コメントに対し、平均 2.5s の指数分布に従う遅延を入れることで、実際のコメント欄のようなコメントタイミングのランダム性を模倣することを狙う。この遅延は、Chat Completions API の応答時間が長いため、それを隠匿し自然さを出す効果も狙っている。なお、後述の動作確認においては、平均して total tokens 199.73 の応答を 6.26s の遅延で受信した。

3 利用例と考察

実際にシステムを利用し動作確認を行う。同時に、発話量と作業時間を計測し、その結果をもとに提案システムにより発話思考における発話量が増えるか、作業時間が短縮されるかといったことを考察する。

3.1 環境

被験者は著者一人で、被験者の持つ環境でシステムを動作させる。システムは Python で作成されたため、実行には Python インタプリタの導入を要する。また、音

1) 工学院大学大学院工学研究科情報学専攻

1) <https://cloud.google.com/speech-to-text>

2) <https://platform.openai.com/docs/api-reference/chat/create>

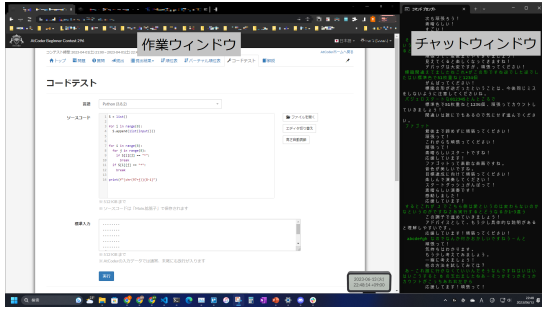


図 1 システム起動時のスクリーンショット

表 1 実行結果

コメント	発話あり [s]	作業時間 [s]	発話率 [%]
あり	643.69	959.61	67.08
なし	472.18	618.35	76.36

声認識を行うため、システムを実行するコンピュータにはマイクが搭載されている必要がある。

対象とするタスクはプログラミングとし、様々な難易度が存在する AtCoder³⁾より問題を選択することとした。今回は被験者のプログラミングレベルに合わせ、B 問題を選択した。

提案システムによる影響を調査するため、コメント生成機能のありなしで各 1 問ずつ実施する。生成ありは AtCoder Beginner Contest(ABC) 296 より、生成なしは ABC 297 より出題した。

3.2 結果と考察

コメント生成のありなしで発話した時間、作業時間がどう変化したかを比較する。結果を表 1 に示す。表内の発話率は発話時間を作業時間で割り百分率にした値である。この結果からは提案手法の発話促進効果、作業促進効果は確認できない。

このような結果になった要因として、プログラミング問題の差が挙げられる。今回は問題レベルこそ統一しているものの、ケアレスミスの発生などの影響で常に同等の作業時間で終わられるとは限らない。そのため、今回の結果のみで結論を求めることは早計と考えられ、実施する問題の数を増やした上で平均化を行い、よりもっともらしい値を用いて議論を行う必要がある。

ここで、発話履歴を確認すると、発話の内容は、問題文の読み上げ、書こうとしているコードの内容やアルゴリズムなどの実装に関する報告、エラーに対するリアクション、といったものが主となっており、問題によって大きな差は見られなかった。しかし、コメント生成ありの履歴の一部には、「私も挑戦したことがある」というコメントに対し「解いたことある人とかいるんですね」といった、コメントに対するリアクションが 1,2 回程度含まれていた。体感としては生成コメントを見て積極的に発話したとは感じておらず、現状のシステムには発話促進効果がないと考えているが、「応援しています！頑張ってください！」といった応援コメントを見て、嬉しさや少しではあるがモチベーションの向上を感じたのもまた事実である。これらより、効果を明確にするためにコメントの生成手法を改善する必要があると考えている。

本システムを利用した上で気付いた点として、コメン

3) <https://atcoder.jp>

表 2 文字起こしとコメント生成の結果

API による文字起こし	角ますが七 HB 2 みみたいな感じで玉がついてるって感じなんです
生成コメント	なるほど、玉がついていると握りやすそうですね。角ますに玉がついているのは初めて知りました。HB2 という鉛筆と同じような表現方法ですね。握りやすさが向上すると作業もはかどりそうですね。
著者による文字起こし	各マスが んー あー AI B2 みみたいな感じで名前がついてるって感じなんです

トが見辛いという問題がある。これはフォントサイズやコメント同士のスペース、発話内容の文字起こしとの混同が要因として挙げられる。これらは Terminal よりも適切な、専用の UI を構築することで改善する可能性がある。

これとは異なる問題点として、発話内容の文字起こしが不正確で、適切なコメントが生成されていないという点がある。表 2 に Speech-to-Text API の文字起こし結果、コメント生成結果、並びに著者による文字起こし結果を示す。この結果から、より適切なコメントを得るために、文字起こしの精度が悪化している原因を調査し、より正確な結果を取得するための対策が必要である。

4 実験計画

ここでは、複数人の被験者を用意して行う実験の計画を説明する。対象とするタスクは利用例と同様にプログラミングとし、AtCoder より適切な難易度の問題を選択し、被験者に解答を求める。実験の流れは次のようになる。

まず、発話思考、並びに競技プログラミング形式に対する練習と、プログラミングスキルの確認を兼ねて、AtCoder の ARC(AtCoder Regular Contest) 問題を、最も簡単な A 問題から順に解いてもらう。この時、被験者が難しいと判断した場合、回答を中断することができる。なお、A 問題で中断してしまった場合は、より簡単なコンテストである ABC(AtCoder Beginner Contest) の問題を A 問題から順に解いてもらう。

この結果から問題レベルを決め、本番として同レベルの問題を、コメント生成ありで 3 問、コメント生成なしで 3 問の合計 6 問を解いてもらう。ここで、生成ありなしの適用順はランダムとし、発話量と作業時間を計測する。この結果を比較することで、提案手法が発話思考における発話量を増やすか、作業時間が短縮されるかを調査する。

5 おわりに

本論文では、応援コメントの自動生成によるタスク遂行促進手法の提案を行った。今後は、動作確認によって得られた問題点を改善した上で、複数名の被験者に協力を依頼して実験を行い、遂行促進効果のさらなる検証を進める予定である。

謝辞

本研究の一部は、2023 年度科研費基盤研究 (C)(課題番号: 21K12147) によるものです。ここに記して謝意を表すものとします。

参考文献

- 伊藤貴昭, 垣花真一郎. 説明はなげ話者自身の理解を促すか. 教育心理学研究, Vol. 57, No. 1, pp. 86–98, 2009.
- 飯田拓人, 小林稔. 発話思考を促進するインタラクティブキャラクタの提案. ワークショップ 2019 (GN Workshop 2019) 論文集, 第 2019 巻, pp. 73–78, nov 2019.