2023年度 修士論文

共感的な応答を生成する対話システム

兵庫県立大学 大学院

情報科学研究科 データ計算科学専攻

IM22D048
名和 夢華

2024年1月19日 提出
指導教員　川嶋宏彰教授

概　要

本研究では，対話の中で共感した反応を返し，話したいことを話しやすいように誘導してくれる対話システムの実現を目指す．ユーザーの感情を踏まえた対話システムを実現する手法として，雑談対話に必要となるようなスキル (共感，知識，話者性) に対応するコーパスでファインチューニングしたモデルが多く提案されている．しかしこのように学習データをフィルタリングし end-to-end で学習すると，事前学習モデルに用いたデータの影響やファインチューニングデータの影響によって想定した対話にならないような課題が挙げられる．

そこで本研究では，二つの段階のフィルタリングにより，ユーザー発話より推定された感情に基づき，システム側が適切な応答を返す枠組みに注目する．まず，ユーザー発話とシステム発話（応答）の適切な組み合わせを,（ユーザー発話の感情ラベル，システム発話の対話行為ラベル）というペアの集合で設定しておく．一段階目は学習時のフィルタリングであり，適切な発話・応答の対応をあらかじめ言語モデルとして獲得する．具体的には，適切な感情ラベル・発話行為ラベルのペア集合であらかじめフィルタリングされた学習データを用いて，事前学習済み対話システムをファインチューニングすることで，適切な応答が生成されやすい言語モデルが学習されると期待できる．二段階目は発話生成時のフィルタリングであり，対話進行時にもユーザーの感情状態を推定し，動的に生成発話を制御することで，対話の文脈に沿った適切な応答を生成できる枠組みとする．具体的には，ユーザー発話から推定されるユーザーの感情に沿った応答を返すために，生成すべき対話行為ラベルを予測し，そのような特徴を入力できるような対話生成モデルを構築する．

実験を通じて，提案手法ではユーザーの感情を踏まえた，共感的な応答文を生成することができた．しかし一方で，相手の感情に共感した応答だけでなく，ユーザーが話しやすくなるような，質問や自己開示が含まれた応答生成を期待していたが，結果としては単調な応答文が生成される結果となった．これらの改善を今後の課題とする．

目 次

図 目 次

表 目 次

1. 序論

1.1 研究背景

近年，日本において，核家族化と高齢化が急速に進行しており，これに伴い独居の高齢者の人数が増加している [7]. 一般に独居での生活では，家族がいる場合に比べて，会話をする機会が減少する．会話する機会が減少すると，認知症を患うリスクが増大することが知られている [8][9] ことから認知機能を維持するために会話の機会を創出することが求められている [10].

対話システムとは，言語を用いて人間とコミュニケーションを行う機械のことであり，例えば AI スピーカやチャットボット等を指し，これらは一問一答ではなく，複数回対話を行う．対話システムを用いることによって，人がデータベースや機器の操作をしたり，対話自体を楽しんだりすることが可能になる．一方で，現在の対話システムの技術では，人間の様にあらゆる状況であらゆる話題の対話が行えるシステムを構築することはできない．そのため，目的に応じて異なるタイプの対話システムを構築することが通常である．対話システムは，タスクの有無・種類，話題の範囲（ドメイン），入出力の 4 つのモダリティ，対話参加者の数の 4 つの観点 [11] から分類でき，システムのタイプに応じて適切なアーキテクチャと要素技術を選択する必要がある．本研究では，タスクの有無・種類に焦点を当て対話システムを構築する．対話の目的には，対話によって遂行すべきタスク (レストランの情報提供や，機器の使い方の説明，インタビュー，説得，交渉，クイズ，ゲーム等) が明確なシステムであるタスク指向型と，タスクが明確ではなく雑談のような会話を行う非タスク指向型に分けられる．非タスク指向型対話システムは目的なく対話を行うのではなく，ユーザーを楽しませる，ユーザーとの信頼関係を築く，ユーザーの趣味・嗜好を知るなどの目的がある．加えて，雑談対話システムは同じような会話をしても変わらずに返答してくれる，人には言いにくいことも気兼ねなく話すことができる等の特徴がある．こうした背景により，雑談対話システムとの会話は時間や場所の制約がないため，会話の機会が少ない人や認知症患者など，様々な人々のいい話し相手になることが期待できる．

1.2 研究目的・アプローチ

本研究では「感情」に着目したシステムを構築する．ユーザーに継続して対話システムと会話したいと思ってもらうためには，ユーザーの「感情」に対してシステム側が「共感」した反応を返すことが非常に重要であると考えられる．そのため本研究では，対話の中で共感した反応を返し，話したいことを話しやすいように誘導してくれる対話システムの実現を目指す．

このような対話システムを実現する手法として，雑談対話に必要となるようなスキル (共感，知識，話者性) に対応するコーパスでファインチューニングしたモデル [12][13][14][6] が挙げられる．しかしこのように学習データをフィルタリングし end-to-end で学習すると事前学習モデルに用いたデータの影響やファインチューニングデータの影響によって想定した対話にならないというような課題が考えられる．

そこで本研究ではアプローチとして，2 段階のフィルタリングにより共感的な対話システムを実現していく．まず，ユーザー発話とシステム発話（応答）の適切な組み合わせを，（ユーザー発話の感情ラベル，システム発話の対話行為ラベル）というペアの集合で設定しておく．一段階目は学習時のフィルタリングであり，適切な発話・応答の対応をあらかじめ言語モデルとして獲得する．二段階目は発話生成時のフィルタリングであり，対話進行時にもユーザーの感情状態を推定し，動的に生成発話を制御することで，対話の文脈に沿った適切な応答を生成できる枠組みとなる．

このように二つの段階でフィルタリングを行うことによって，感情を踏まえた対話が集められた学習データで学習されていることに加え，ユーザーの現状の感情等を交えながら，応答文を生成できるので，end-to-end で対話生成を行うよりもより自身の理想とするような，ユーザーの感情に寄り添った応答文が期待できる．

また，このような対話システムを実現することで，ユーザーの対話欲求を満たし，話し終えた後に少しでもポジティブな感情になって欲しいと考える．

1.3 本研究における貢献

本研究をまとめると，次のような貢献が存在する．

- 学習時と発話生成時の二つの段階でフィルタリングを行うことによって，共感的な応答生成を可能にすることを見出した．

- ４つのパターンのフィルタリング方法の結果を比較することで，各フィルタリングの効果を見出した．

1.4　本論文の構成

本論文の構成は序論を含め全 5 章で構成される．2 章では雑談対話システムに関しての関連研究について述べる．3 章では本研究の提案手法，4 章では評価手法とその結果について，5 章でまとめと今後の課題を述べる．

2. 関連研究

2.1 Transformer

2017 年，Google の研究者たちは，系列モデリングのための新しいニューラルネットワークである Transformer[1] を提案する論文を発表した．Transformer は，注意機構 (Attention) を核にした構造を持っており，従来の RNN や LSTM という手法と比べ，並列処理が容易であったり，長い文章などのデータが入力されても，記憶して最後まで処理が可能であることを特徴としている．図 1 に Transformer の構造を示す.

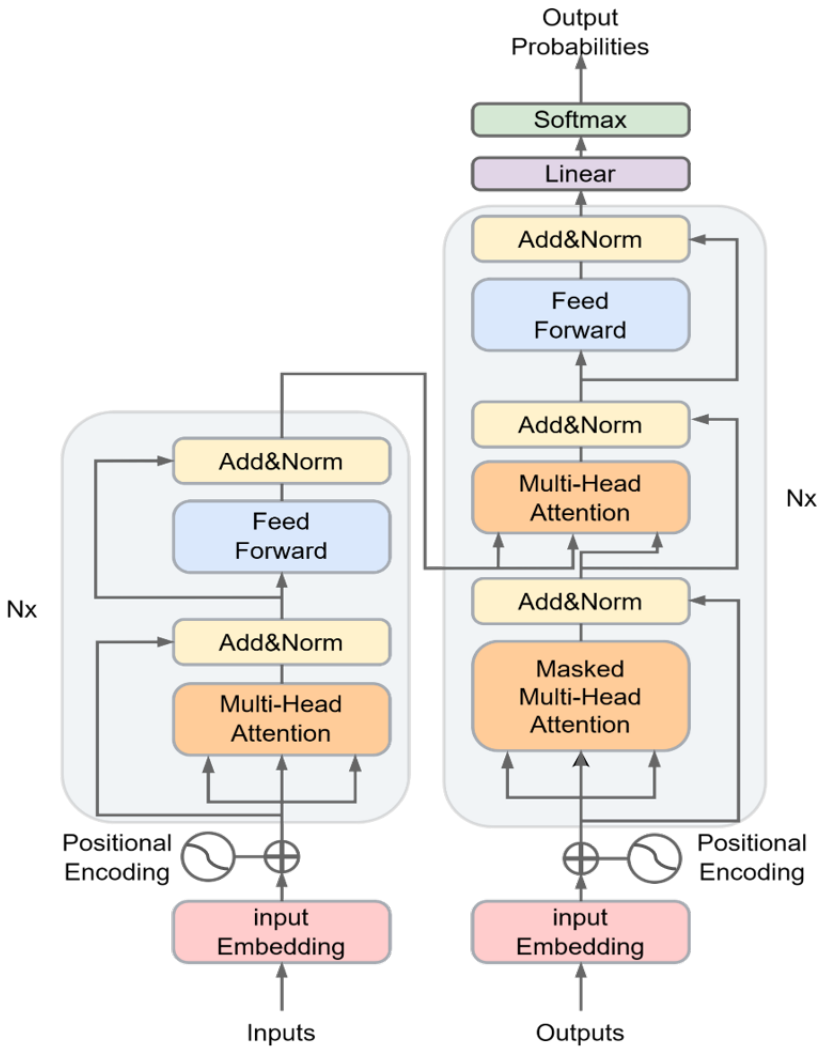


図 1: Transformer のアーキテクチャ（[1] の Figure1 をもとに作成）

2.1.1 エンコーダ・デコーダアーキテクチャ

Transformer はエンコーダ・デコーダアーキテクチャをベースとしている．エンコーダは，入力されたトークン列を埋め込みベクトルの列に変換を行う．そして，エンコーダの隠れ状態を利用し，トークンの出力系列を 1 つずつ繰り返し生成を行うものがデコーダである．入力テキストをトークン化し，Embedding 層でトークン埋め込みに変換を行う．そして，トークン埋め込みは，各トークンの位置情報を含む位置埋め込み (Positional embedding) と組み合わされ，エンコーダに入力される．Transformer のエンコーダ層は，Multi Head Attention 層と，各入力埋め込みに適用される全結合の順伝播層から構成されており，それが複数層積み重なった構造となっている．

また，デコーダはエンコーダと違って,Masked Multi Head Attention 層と Encrder,Decoder Attention 層の 2 つのアテンション層を持つ．Masked Multi Head Attention 層は学習時のデコーダの input は，i 番目のトークンを予測する際は $i-1$ 番目までのみの情報を使うべきなので $i+1$ 番目以降のトークンに対してマスキング処理を行っている．それにより．各時刻で生成するトークンが，過去の出力と現在予測されているトークンだけに基づいていることを保証する．Encrder・Decoder Attention 層は Attention を用いることによって，より注意を払うべき情報を参照しやすいように処理を行う．Attention については次節 2.1.2 でし詳細を述べる. そうすることで 2 つの異なる系列 (例：2 つの言語) からのトークンを関連付ける方法を学習する．

デコーダもエンコーダと同じく上記の処理を複数層に渡って行い最終的な潜在表現を線形層によって語彙サイズに次元を変換後，Softmax 関数で正規化を行うことで，各語彙の生成確率に相当する 0〜1 の数値を獲得する．

2.1.2 Scaled Dot-Product Attention

Transformer で使われている注意機構は query,key,value 型の注意機構である．

Transformer で用いられる注意機構の図 2 に示す．Scaled Dot-Product Attention ではそれぞれのトークンはこれらの 3 つのベクトルにより特徴付けられる．ここで，n 個のトークンで構成される文章を処理することについて考える．入力，クエリ，キー，バリュー，出力のそれぞれの (行) ベクトルを縦に結合した行列をそれぞれ X，Q，K，V，A と置く．

$$X = vstack(x_i, x_2, ..., x_n) \tag{1}$$

$$Q = vstack(q_i, q_2, ..., q_n) \tag{2}$$

$$K = vstack(k_i, k_2, ..., k_n) \tag{3}$$

$$V = vstack(v_i, v_2, ..., v_n) \tag{4}$$

$$A = vstack(a_i, a_2, ..., a_n) \tag{5}$$

ここで $vstack$ は行列を縦に結合する関数である．

また，出力に対して行列 W^Q，W^K，W^V で線形変換を行うことにより，クエリ Q，キー K，バリュー V と呼ばれる 3 つの d 次元ベクトルを得られる．

$$Q = XW^Q \tag{6}$$

$$K = XW^K \tag{7}$$

$$V = XW^V \tag{8}$$

Transformer では，クエリとキーのどの程度関連しているかを Scaled Dot-Product と呼ばれる Q，K の内積を $\sqrt{d}$(d はベクトルの次元) で割って得られる重みをスコアとして用いる．Scaled Dot-Product は埋め込み行列積を用いて効率的に計算される．類似しているクエリやキーは内積が大きくなり，そうでない場合は小さくなる．単に内積をとるだけだとベクトルの次元 d が大きくなると最終的にほとんどの重みがほぼ 0 になってしまい学習が進まなくなる．これを回避するために，スコアを計算するときに $\sqrt{d}$ で割る必要がある．また，計算された類似度の割合に応じた重みでバリューのベクトルを配合した出力を返す．出力 A はクエリ Q，キー K，バリュー V の関数として，

$$A = Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d}}\right)V \tag{9}$$

と表現される．

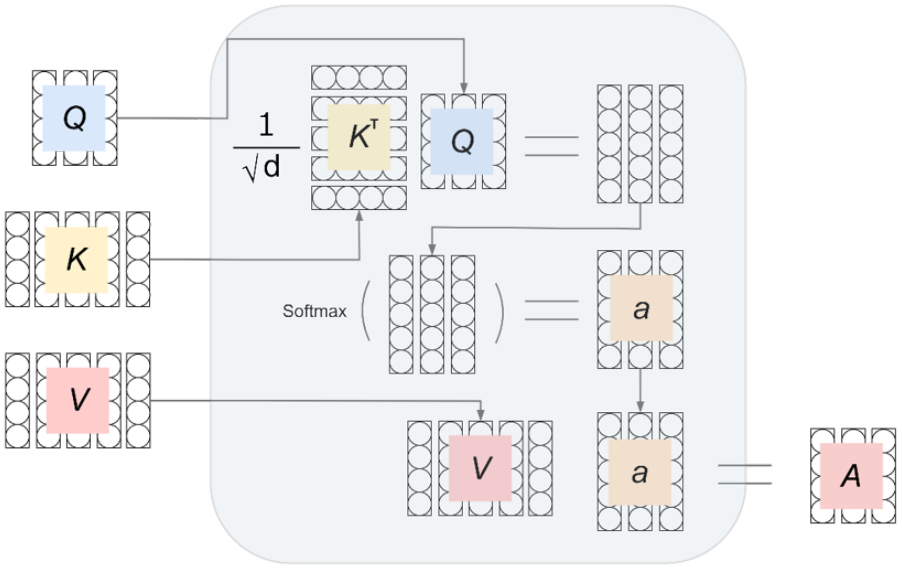


図 2: Transformer で用いられる注意機構の例（[2] の p149 の図 6.3 を元に作成）

2.1.3 Multi-Head Attention

Transformer に用いられている Multi-head Attention は，クエリ，キー，バリューの組を複数用意しておき，それぞれの組に対して Scalsed Dot-Product Attention を適用し，最後に出力を 1 つに集約する方法である．注意機構を 1 つだけ用いた場合，複数の観点で情報を取り出すことが難しい．つまり，Multi-head Attention は，それぞれのヘッドが別々の位置から情報を取得して情報をうまく混ぜ合わせてくれることが期待できる．

Multi-Head Attention を数式で表すと，次のようになる．

$$MultiHeadAttention(Q, K, V) = Concat(head_1, ..., head_h)W^O \tag{10}$$

$$head_i = Attention(QW_i^Q, KW_i^K, VW_i^V) \tag{11}$$

ここで，$i \in (1, ..., head)$，$W_i^Q \in R^{d_k \times d}$，$W_i^K \in R^{d_k \times d}$，$W_i^V \in R^{d_v \times d}$，$W_i^O \in R^{d \times d_v head}$ とする．

2.1.4 学習

自然言語処理の問題を解くニューラルネットワークのモデルは一般的に，モデルに対して入力されるデータとそれに対する望ましい出力の関係をラベル付きデータを用いて学習を行う．しかし，ラベル付けされていないモデルを学習させるために必要なデータの収集にはコストがかかる場合があり，少ないデータで学習したモデルはタスクを処理する性能も低い．そこで，大規模な文章コーパスを用いて汎用的な言語のパターンを学習する事前学習を行っ

たのち，個別のタスクのラベル付きデータを用いてそのタスクに特化させるように学習するファインチューニングといった手法がある．これにより，モデルは特定のタスクに対して最適化され，より高い性能を発揮できるようになる．

2.1.5 BERT

BERT は 2018 年に Google により提案された [4] モデルで，Transformer のエンコーダーを事前学習したものである．BERT は文中で任意の位置の単語をマスクし，そのマスクされた単語を予測するタスクであるマスク付き言語モデルを事前学習に用いることで，文全体を参照する言語モデルを学習した．さらに，BERT では入力文がコーパス中における隣接文同士かどうか識別するタスクである Next Sentence Prediction も事前学習に採用し，文の埋め込み表現を獲得することを目指した．

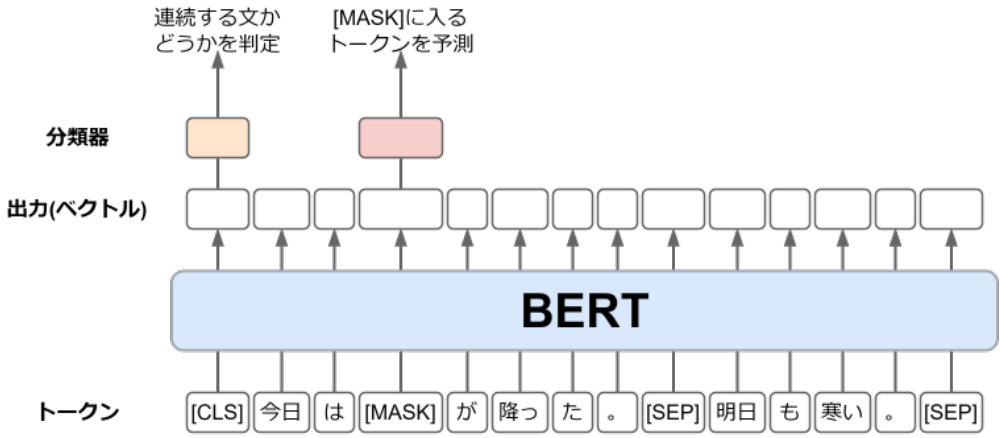
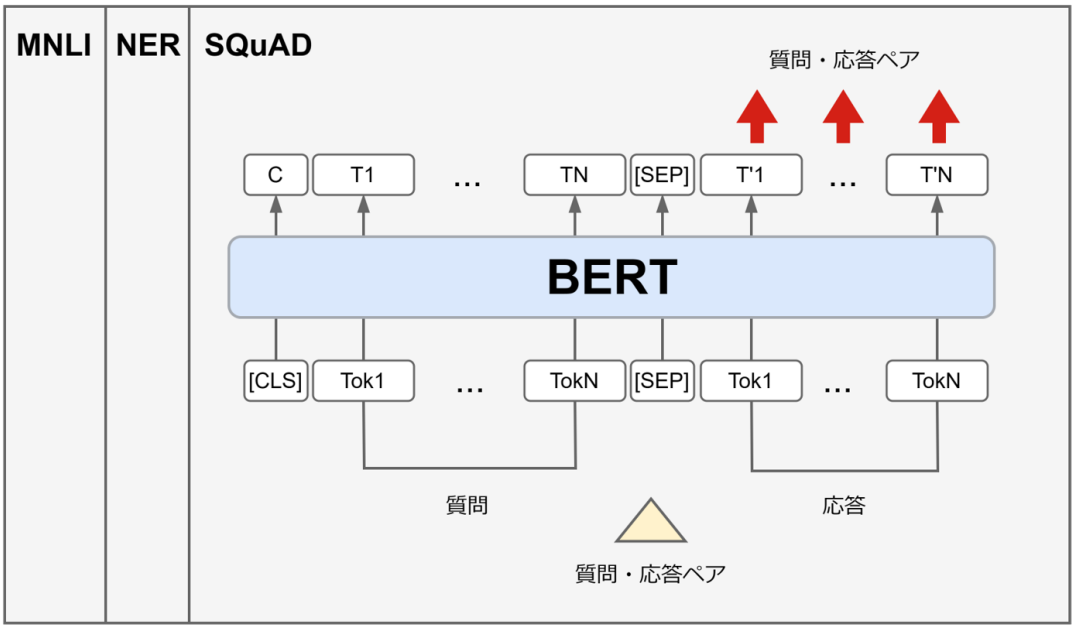


図 3: BERT における事前学習（[3] の p40 の図 3.4 をもとに作成）

BERT のファインチューニングでは BERT のモデルの上にタスクに合わせて設計したニューラルネットワークの層を積み重ね，BERT の内部のパラメータを含め，ネットワーク全体を学習する．BERT は Transformer のエンコーダに基づくアーキテクチャであるため，ファインチューニングのタスクは単語や分類問題が中心となる．事前学習により言語に関する一般的な知識や文脈付き単語埋め込みの合成方法を学習してあるため，ネットワーク全体をゼロから学習するときと比べ，少ない量のラベル付きデータでもファインチューニングが行える．Devlin ら [4] が行った BERT のファインチューニング評価実験では，数千〜数十万件のラベル付きコーパスを用い，様々な自然言語処理タスクにおいて既存研究を大幅に上回る性能を達成した．BERT における事前学習，ファインチューニングの概要を図 3，4 に示す．



SQuAD: クラウドワーカーによって作成された質問と回答が対になったデータセット

MNLI : 自然言語推論のコーパスであり，約43万の前提と仮説のペアから構成されている。前提と仮説の内容上の関係に応じて含意，矛盾，どちらとも言えない，といったラベルが付けられている

NER : 固有値表現抽出

図 4: BERT におけるファインチューニング（[4] の Figure1 を元に作成）

2.1.6 GPT-2

GPT-2 は Transformer のデコーダをアーキテクチャとして採用し，言語モデルとして最大 15.4 億のパラメータ数で訓練したものである [5]，[15]. GPT-2 の構造を図 5 に示す．GPT は Transformer のデコーダにより，入力されたテキストから次の単語を予測するという言語モデルの学習を行う．GPT-2 のテキスト生成の図を図 6 に示す．事前学習に用いる，長さが N 単語のテキストの単語列を $s = x_1, x_2, ..., x_N$ で表す．位置 i の単語 x_i を予測するとき，それよりも k 個前に出現する単語列 $x_{i-k}, x_{i-k+1}, ..., x_{i-1}$ を文脈として用い，言語モデルの負の対数尤度

$$J = -\sum_{i=k+1}^{N} \log P(x_i | x_{i-k}, x_{i-k+1}, ..., x_{i-1}) \tag{12}$$

を最小化するように L 層の Transformer のデコーダ部分を学習する．

また，GPT-2 のファインチューニングではファインチューニングしたい訓練データを生テキストコーパスと見なして言語モデルの最適化を行い，分類モデルの汎化性能，および学習の収束の改善を狙う．GPT はデコーダに基づくアーキテクチャを採用しているため，分類問題だけでなく機械翻訳や要約，対話などの言語生成タスクにも適用できる．本研究では，

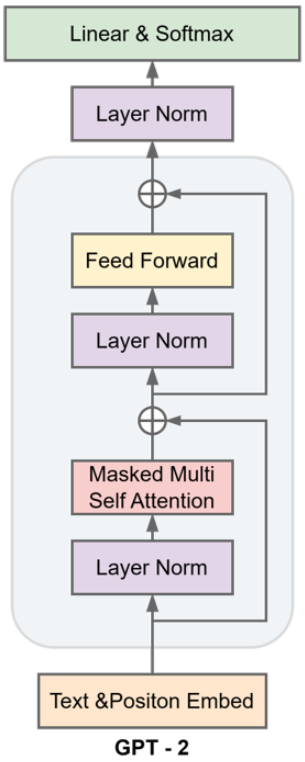


図 5: GPT-2 のアーキテクチャ （[5] の figure1 をもとに作成）

GPT-2 を基に雑談応答生成を学習し，対話システムを構築している．

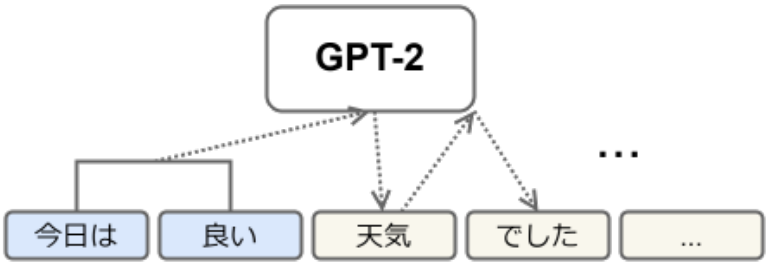


図 6: GPT-2 のテキスト生成イメージ

2.2 ファインチューニング用コーパスに関する研究

対話システムに特徴量を追加するような試みの一つに，Persona-chat[14] というものがある．これは話者の特徴を定めるプロフィール文を 5 文程度をセットとして設定し，各話者が与えられた定められたセットのプロフィール文に従って対話を行うことで，疑似的に様々な話者の対話を収集するコーパスである．また，Empathetic Dialogues[6] は，感情的な状況に

ついて発話する「話し手」と，それに応答する「聞き手」の 2 者による，共感的な対話を収集したコーパスである．これはクラウドソージングを用いて，32 種類の感情を示す単語について話し手がその感情を抱く状況の説明文と対話を 24850 対話収集している．

このように対話システムに話者のユーザープロフィールや感情を明示的に利用することで，より情報に基づいた対話をすることを試みたものである．

2.3 学習時フィルタリングに関する研究

本間ら [12] は異なる 3 つの方法で作られたファインチューニング用データの効果を比較し，どのファインチューニング用データが高い共感性を持つ応答を生成するか，応答から知覚される対話システムの個性を，設計者が用意した対話例から読み取れる個性に近づけることができるのかの 2 点を同時に実現できるかを実験的に調べたものである．

[12] の実験で使用した，３つの対話データを以下に示す．

1. 対話例ファインチューニング

   設計者が作成した対話例をファインチューニングデータとしたもの．

2. 対話行為フィルタリング

   大規模対話データから所望の対話行為をもつ対話文を抽出したものをファインチューニングデータとしたもの．

3. プロトタイプフィルタリング

   設計者が用意した少数の発話例文 (プロトタイプ発話と呼ぶ) と似た対話行為が現れている対話文をファインチューニングデータとしたもの．

ここで，2 の問題点として細やかな共感性の質的制御ができないこと，3 の問題点として発話例文が少数であると，それを元に抽出された対話データでは，設計者の意図と異なる対話が収集される可能性が高くなる恐れがあると述べられていた．結果としては 1 のファインチューニング手法が対話システムの個性再現に高い効果であることが示された．また，共感性の点では 1 と 2 の 2 つの手法で高い評価が得られ，個性再現と高い共感性の両方を実現できるのは 1 の手法であると示された．2 の対話行為を用いる手法では対話行為推定器の学習データによって「評価者が期待する共感性」とは異なるシステム応答が出力される結果となってしまうという課題が述べられていた．

2.4　生成時フィルタリングに関する研究

現在に至るまで，ユーザーの感情について焦点を当てることと，より良い応答文生成を行う対話システムを作成することは別々の研究として行われており，両方を考慮している研究は非常に少なかった．そこで，Yuhan Liu et al[16] は上記に記した両方を考慮したユーザーの感情を追跡し適切な応答生成を行うこと目的とした対話システムを提案した．まず Emotion State Tracker という部分で話し手の感情を推定し，話し手の感情に基づいて Empathetic Dialogue Policy Predictor という部分で聞き手の感情，聞き手の意図を予測する．これらによって全体の状態を予測し，特徴の抽出を行う．そして予測された感情状態と対話の文脈に基づいて応答文を生成を行う．それによって，相手の感情に基づいた適切な応答文生成が可能になる．

2.4.1　付加情報に関する関連研究

encoder-decoder モデルの応答生成において，encoder へ情報を入力する際，図 1 に示すように，対話の文脈に加え，付加情報を同じテキスト形式で入力することが可能である．そこで，Rashkin[6] は状況文と感情語を入力文に追加することで状況と生成すべき感情が定まるため，発話生成の安定性が高まることが期待できると述べている．

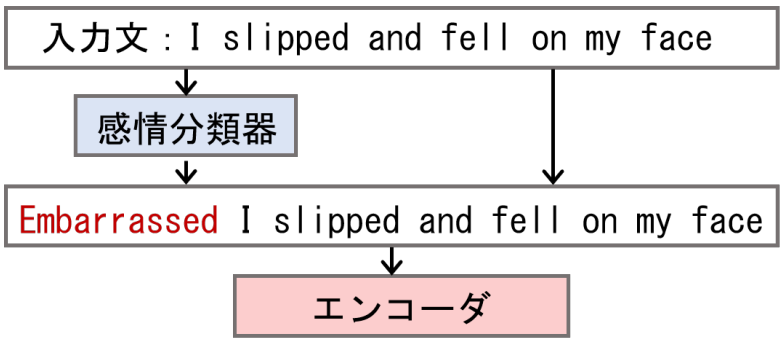


図 7: Encoder へ付加情報（感情語）の追加（[6] の Figure5 を元より作成）

3. 提案手法

本研究では，ユーザー発話より推定された感情に基づき，システム側が適切な応答を返す枠組みに注目する．二つの段階で研究を進める．まず，ユーザー発話とシステム発話応答の適切な組み合わせを,（ユーザー発話の感情ラベル，システム発話の対話行為ラベル）というペアの集合で設定しておく．

提案手法の全体像を図 8 に示す.

一段階目は学習時のフィルタリングであり，適切な発話・応答の対応をあらかじめ言語モデルとして獲得する．具体的には，適切な感情ラベル・発話行為ラベルのペア集合であらかじめフィルタリングされた学習データを用いて，事前学習済み対話システムをファインチューニングすることで，適切な応答が生成されやすい言語モデルが学習されると期待できる．

二段階目は発話生成時のフィルタリングであり，対話進行時にもユーザーの感情状態を推定し，動的に生成発話を制御することで，対話の文脈に沿った適切な応答を生成できる枠組みとする．具体的には，ユーザー発話から推定されるユーザーの感情に沿った応答を返すために，生成すべき対話行為ラベルを予測し，そのような特徴を入力できるような対話生成モデルを構築する．以下の節では，これら各段階の処理を詳しく述べる．

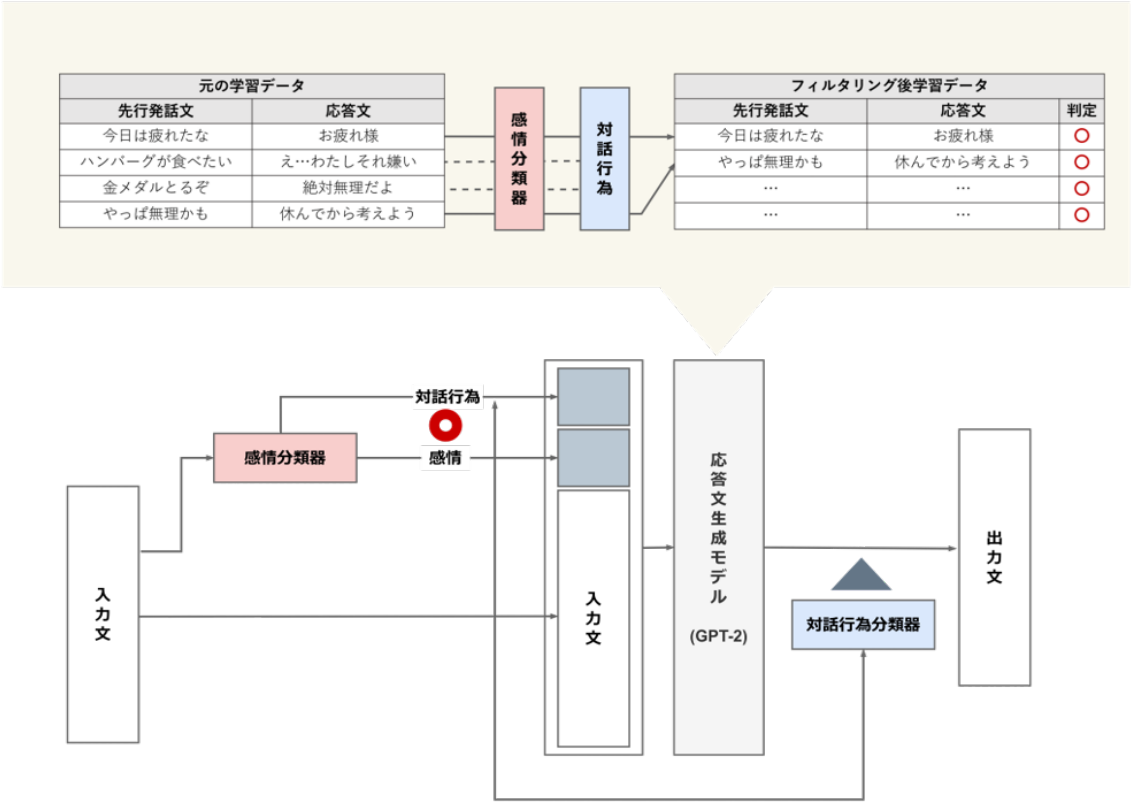


図 8: 本研究の全体像：一段階目に，学習データに対して,「感情ラベル」と「対話行為」の適切な組み合わせのみを学習データとするためにフィルタリングを行う．二段階目に，発話生成時にユーザーの感情状態を推定し生成発話を動的に制御する．

3.1 BERT を用いた感情分類モデルの作成

学習データの対話文に感情ラベルを付与するために入力された文章に対して入力文が何の感情であるかをマルチラベルで推定する感情分類モデルを作成する．

感情分類モデルの作成に用いたデータセット [17] は，日本語の感情分析のための 17000 件のデータセットである．これは，plutchik[18] の 8 感情 (喜び・悲しみ・期待・驚き・怒り・恐れ・嫌悪・信頼) に基づき，50 人の注釈者が自身の過去の SNS の投稿に主観的な感情強度と読み手の客観的な感情強度の両方で 4 段階 (無・弱・中・強) 付与したデータセットである．感情ラベルの例を以下の表 1 に示す．

本研究では，表 1 で網掛けした部分である主観感情のみを学習データとして用いて，感情分類器を作成する．また，マルチラベル分類は文章が複数のカテゴリーに属すことをいい，コンピュータで処理を行うとき，文章が属すカテゴリーを 0 または 1 からなる Multi-hot ベ

表 1: 感情強度ラベルの例

文	タイヤがパンクした... いたずらの可能性が高いんだって...							
感情	喜び	悲しみ	期待	驚き	怒り	恐れ	嫌悪	信頼
主観	0	3	0	1	3	0	0	0
客観 A	0	3	0	3	1	2	1	0
客観 B	0	2	0	2	0	0	0	0
客観 C	0	2	0	2	0	1	1	0

クトルと呼ばれる形式で表現する必要がある．そのため，本研究では，各文に対して感情強度が一番が大きい感情を 1 として他の感情は 0 とした．表 1 で例えると悲しみと怒りが 1 となり喜び・期待・嫌悪・恐れ・信頼は 0 となる．そのように修正した学習データの感情ごと正例，負例の分布を以下の図 9 に示す.

以上のデータを用いて感情分類モデルを事前学習済みの BERT をファインチューニングすることで作成した．マルチラベル分類は，分類スコアに対してシグモイド関数を適用し予測確率に変換する．それぞれのカテゴリーを「選ぶ」か「選ばないか」の二値分類を行うため，マルチラベル分類においてファインチューニング時の損失関数は予測確率と実際に文章がそのカテゴリーに属しているかどうかとの間のバイナリクロスエントロピーを用いる．ラベル数を N，t_n が正解ラベル，y_n が予測値とするとき，損失関数は以下のように記述できる．

$$BCELoss = -\frac{1}{N}\sum_{n=1}^{N} H(t_n, y_n) = -\frac{1}{N}\sum_{n=1}^{N}[t_n \log y_n - (1-t_n)\log(1-y_n)] \qquad (13)$$

今回使用するデータセットでマルチラベル分類を行う場合，図 9 の数値から，どの感情ラベルに対しても正例が少なく，データの不均衡が生じていることが分かる．このように負例に偏ったデータでモデルの学習を行った場合，不均衡データに強い影響を受けてしまうため，自身の目的に対して無意味な指標が最適化されてしまう可能性がある．そのため，損失を計算する際にラベルごとの正例の割合を利用した損失の重みづけを行う．例えば，データセットに 1 つのクラスにおいて正例の数が 100 件，負例の数が 300 件含まれている場合，そのクラスの重みは $\frac{300}{100} = 3$ となる．損失はデータセットに 300 件の正例がまれて含まれているかのように振る舞う．このように割合の少ない正例に対して損失を大きくする処理を行うことで不均衡データに対処する.

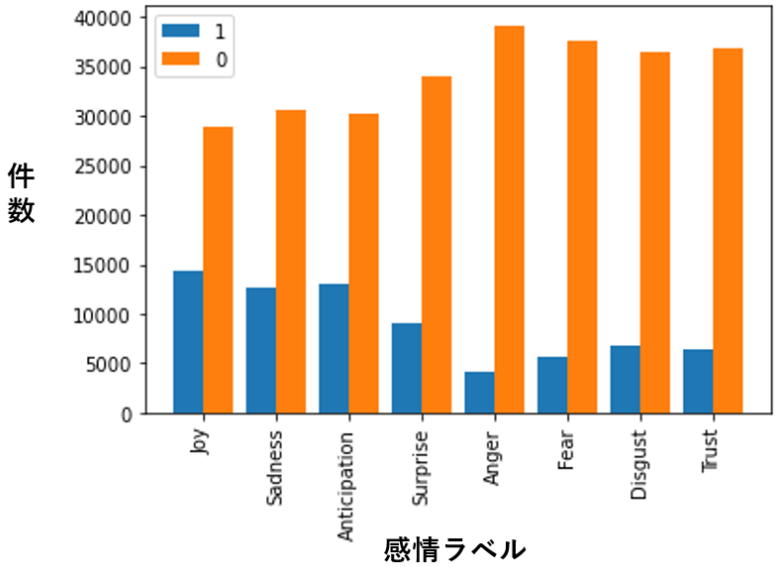


図 9: 感情ラベルごとの正例，負例の分布

3.2 BERT を用いた対話行為分類モデルの作成

対話行為とは，ある会話の一文の内容を話者の意図がどのような種類であるかを分類したものである [19][20]. 雑談対話の内容は多岐にわたるためユーザーからの発話の内容が難しく，雑談対話における応答生成は困難なでタスクである．そこで，対話文に対して，対話行為推定を行うことによって，対話内容の理解や自然な応答生成が期待できる．具体的な，対話システムにおける対話行為情報の利用目的としては，ユーザーの意図の理解，システムの応答文生成における条件，システムの対話制御などが挙げられる．例えば，ユーザーの発話を分析し，対話行為にクラス分けすることは，ユーザーの意図理解の 1 つと見なせる．ユーザーが挨拶をしているか，何を質問しているのかなどをシステムが理解することで，その後の対話の展開を予測しやすくなることが考えられる．本研究では，上記で記したような目的を果たす対話行為を感情と組み合わせて用いることでより共感的な応答生成を目指す.

表 2: JAIST タグ付き自由対話コーパスの一例

話者	発話文	対話行為	共感
F011	どうですか．あの，大学は.	Wh 質問	その他
F089	あっ，もう，な，少しずつ慣れてきたような感じなんですけど，でも，まだまだわからないことだらけで.	応答 (平叙)	その他

本研究では対話行為タグがあらかじめ付与された学習データを用いる．用いたデータセットは JAIST タグ付き自由対話コーパス [21] である．例を表 4 に示す．このデータセットは 10 代から 90 代の二人の雑談会話である名大対話コーパスの一部の対話に対して対話行為タグ・共感タグを付与したものである．対話数は 97，発話数は 92020 である．表 3 に各対話行為，共感の概要とコーパスにおける分布数を示す.

表 3: 各対話行為・共感の概要と分布数

対話行為	概要	分布数
自己開示	自身の考えの表明や事実の列挙など	53701
YN 質問	「はい」「いいえ」などで答えられる質問	6430
Wh 質問	平叙文で答える必要がある質問	3950
YN 応答	質問に対する短い肯定または否定	2130
Wh 応答	質問に対する平叙文出の応答	7508
あいづち	話の続きを促す短い発話	9216
フィラー	発話の合間に挟みこむ短い発話	4405
確認	自分の考えが正しいかを相手に確認	3940
要求	具体的な行動を指示もしくは依頼	751
共感	相手の発話に対して共感や賛意を示す	1067
非共感	相手の発話に対して非共感や反感を示す	222
その他	共感・非共感いずれでもない	90731

次節 3.3.1，3.4 で, 感情ラベルと対話行為ラベルの組み合わせを決定するために，対話行為推定が必要となるため，以上のデータセットを用いて事前学習済み BERT をファインチューニングし対話行為推定器を作成した．入力は対話文と応答文の 2 つの文であり，出力は 2 つの文から考えられる応答文の対話行為ラベルである.

ここで，損失関数は，予測スコアを Softmax 関数で各ラベルの予測確率に変換し，それと実際のラベルとの間のクロスエントロピーを用いる．そしてここで計算された損失を最小化するように，パラメータが更新される．2 つの確率分布 $P(x_i)$: 正解データ分布，$Q(x_i)$: 予測モデル分布に対してクロスエントロピーは以下で定義される.

$$CELoss = -\sum_{i} P(x_i) \log Q(x_i) \tag{14}$$

3.3 学習時のフィルタリング

学習時のフィルタリング，すなわち，ユーザーの感情に寄り添った共感性を学習データとして得る方法について図 10 の左上を用いて説明を行う．図 10 のように学習データ $D_{all} = \{P^{(i)}, R^{(i)}\}_i$ として先行発話文 $P^{(i)}$ と応答文 $R^{(i)}$ があるとする．そして，感情分類器を用いて先行発話文に感情ラベル $Emotion =$ { 喜び, 悲しみ, 期待, 驚き, 怒り, 恐れ, 嫌悪, 信頼 } のいずれかをを付与する．$P^{(i)}$ に付与した後の先行発話は $P^{(i)}_{em} \in Emotion$ と表す．また，応答文には対話行為ラベル集合 $DialogueAct =$ { 自己開示，YN 質問，Wh 質問,YN 応答，Wh 応答，あいづち，フィラー，確認，要求，共感 } のいずれかが予め付与された学習データを用いる．応答文 $R^{(i)}$ に付与した対話行為ラベルをの応答文は $R^{(i)}_{da} \in DialogueAct$ と表す.

例として，図 10 のように，学習データの中で「今日は疲れたな」という文章に対して,「お疲れ様」というような応答文,「ハンバーグが食べたい」という文章に対して「え... わたしそれ嫌い」というような応答文があるとする．ここでは,「今日は疲れたな」には「悲しみ」,「ハンバーグが食べたい」には「期待」の感情ラベルが節 3.1 感情分類器によって付与される．また,「お疲れ様」には「自己開示」,「え... わたしそれ嫌い」には「非共感」の対話行為ラベルがタグ付けされているとする．そして，タグ付けされている対話行為に対して推定された感情ラベルとあらかじめ定めた適切なペア (感情ラベルと応答文ラベルの組み合わせ) のみを学習データとして対話生成モデルに学習させる．感情ラベルと対話行為ラベルの組み合わせ以下で表現できる．組み合わせの決定の詳細については次節 3.3.1 で述べる．

$Pair =$ {(喜び，Wh 質問，共感) ,(悲しみ，自己開示，あいづち，共感)
(期待，YN 質問，共感),(驚き，YN 質問，共感),
(怒り，自己開示，Wh 質問，あいづち，共感),
(恐れ，YN 質問，Wh 質問，あいづち，共感),
(嫌悪，自己開示，あいづち，共感), (信頼，YN 質問，共感)}

ここでは「今日は疲れたな」という「悲しみ」の感情ラベルの文に対して，肯定や慰めが含まれる「自己開示」の対話行為である「お疲れ様」の応答文とは適切なペアとなり,「ハンバーグが食べたい」という「期待」の感情ラベルの文に対して，相手の気持ちを否定する「非共感」の対話行為である「え... わたしそれ嫌い」の応答文とは不適切なペアとなる．このようにして得られた学習データを用いてモデルに訓練することによってフィルタリングを

行わなかった学習データを用いたときと比べ，より共感的な応答生成が期待できる．フィルタリング後のデータは

$$D_{filterted} = \{P^{(i)}, R^{(i)} | (P_{em}^{(i)}, R_{di}^{(i)}) \in Pair\} \tag{15}$$

と示される．

図 10: 学習時のフィルタリング: 元の対話行為タグ付き学習データに対して，先行発話に感情分類器を通して感情ラベル，応答文に対話行為ラベルの付与を行う．

3.3.1 感情ラベルと対話行為ラベルの組み合わせについて

杉山ら [13] は，Empathetic Dialogues の日本語版コーパスを作成した．感情を示す 32 種類の感情を示す単語 (英語) を英語の意味や使われ方を考慮して日本語に翻訳し，それを基に日本語話者が状況文よび対話をクラウドワーカーが作成する．クラウドソージングを用いて 1 人の作業者が翻訳された感情のリストを参照し，その感情に沿った 1〜3 文の状況文と，その状況下で対話する「話し手」と「聞き手」の 2 者による 4 から 8 発話のテキスト対話を作成したもので，収集した対話は計 20000 対話，59997 発話対である．Empathetic Dialogues の日本語版コーパスの例を 4 に示す．

表 4: Japanese Empathetic Dialogues の一例

話者	発話文	感情	状況文
A	寝ているときに首がもぞもぞしたので，起きて見てみたらムカデがいたの！	驚く	…
B	きゃー，それはびっくりだね.	驚く	…

このコーパスは共感的な対話となるように作成されていることから，このコーパスから得られる情報を用いて，感情ラベルと対話行為ラベルの組み合わせの指標にする．3.2 節で作成した対話行為推定器を用いて，表 4 の話者 B の発話に対して，対話行為ラベルを付与する．そうすることによってこのコーパスでは，どの感情の時にどの対話行為で応答しているのかの割合を知ることができる．以下の表 5 がその割合を示したものである．

この表 5 からどの感情でも「質問」,「Wh 質問」と「自己開示」の割合が高くなることが見て取れる．しかし，ネガティブな感情は「自己開示」の確率がポジティブな感情と比べて高くポジティブな感情はネガティブな感情と比べ「YN 質問」,「Wh 質問の確率が高いことがわかる．以上のような結果を基に感情と対話行為の組み合わせを決定した．決定した組み合わせを以下の表 6 に示す．決定した組み合わせのみの対話を学習データとして扱う．

表 5: Japanese Empathetic Dialogues において感情ラベルと対話行為ラベルのペアの割合

	驚き (%)	怒り (%)	悲しみ (%)	喜び (%)	期待 (%)	信頼 (%)	嫌悪 (%)	恐れ (%)
Wh 応答	4.7	5.6	1.6	4	4.6	2.6	3.2	3.5
Wh 質問	19.4	16.4	16.2	18.9	13.3	11.7	14.6	17.3
YN 質問	31.8	21.9	20.8	25.4	29.8	29.9	17.9	29.3
自己開示	36.5	49.4	51.2	45	42.4	45.1	57.6	42.6
あいづち	4.8	4.6	7.8	5.6	8	9.1	4.8	4.6
要求	0	0	0	0	0	0	0	0
フィラー	0	0.3	0	0	0	0	0	0
確認	2.2	0.6	2.2	1.1	1.3	1.4	1.3	2.2
YN 応答	0.6	0.8	0.2	0	0.6	0.2	0.6	0.5

表 6: 感情ラベルと対話行為ラベルのペア

驚き	怒り	悲しみ	喜び	期待	信頼	嫌悪	恐れ
	自己開示	自己開示				自己開示	Wh 質問
YN 質問	Wh 質問	あいづち	Wh 質問	YN 質問	YN 質問	あいづち	NY 質問
共感	あいづち	共感	共感	共感	共感	共感	あいづち
	共感						共感

3.3.2 応答文生成モデル

発話生成時のフィルタリングを行わないモデルの場合，応答生成モデルには，rinna 社 [22] の japanese-gpt2-small を事前学習済みモデルとして，節 3.2 の JAIST タグ付きコーパス [21] を用いてファインチューニングしたものを用いた．また，応答生成モデルの最大入力長を超えた入力は先頭から切り捨て，モデルに入力した．共感的な応答に対するクロスエントロピーが最小化されるように微調整する．共感的な応答を y とすると，損失関数は次のように記述される．(ただし y のトークン数を $|y|$ 個，$y_{1:t-1} = y_1, y_2, ..., y_{t-1}$ とする．)

$$L_{model} = -\sum_{t=1}^{|y|} \log P(y_t | y_{1:t-1}) \tag{16}$$

Encoder の入力形式は発話の先端を示すための $\langle s \rangle$ トークン，先行発話文と応答文の境目を区別するための $\langle SEP \rangle$ トークン，対話の終端を示す $\langle /s \rangle$ トークンを付与した．

3.4 発話生成時のフィルタリング

発話生成時のフィルタリング，すなわち，学習データのフィルタリングだけでは，事前学習データなどの影響により想定外の応答がでてくる可能性があり，それを避ける発話，及び，ユーザーの感情に寄り添った共感的な応答を得る方法について図 11 を用いて説明を行う．

二段階目の発話生成時のフィルタリングは，対話進行時にも節 3.1 で作成した感情分類器で，ユーザーの感情状態を推定し，動的に生成発話を制御することで，対話の文脈に沿った適切な応答を生成する．具体的には，ユーザー発話から推定されるユーザーの感情に沿った応答を返すために，生成すべき対話行為ラベルを予測し，そのような特徴を対話生成モデル入力する．その後，生成モデルから出力された文を節 3.2 で作成した対話行為分類器を用いて対話行為推定を行い，生成モデルに入力した対話行為と一致する文を出力文とするようなモデルを構築する．

発話生成時のフィルタリングを行う場合の応答生成モデルには，節 3.3.2 の時と同じ事前学習済みモデルおよびファインチューニングデータを用いてファインチューニングしたものを用いた．共感的な応答に対するクロスエントロピーが最小化されるように微調整する．ここで，発話生成時の応答文生成では，感情と，対話行為を入力文に追加するため，共感的な応答を y，感情分類器を通して推定された感情ラベルを v_{em}，感情ラベルから予測される対話行為ラベル v_{da} とする．すると，損失関数は次のように記述される．(ただし y のトークン数を $|y|$ 個，$y_{1:t-1} = y_1, y_2, ..., y_{t-1}$ とする．)

$$L_{model} = -\sum_{t=1}^{|y|} \log P(y_t|v, y_{1:t-1}) \tag{17}$$

Encoder の入力形式は発話の先端を示すための $\langle s \rangle$ トークン，感情ラベルと先行発話文と対話行為ラベルと応答文の境目を区別するために $\langle SEP \rangle$ トークン，対話の終端を示す $\langle /s \rangle$ トークンを付与した．

ここで，図 11 の様に応答文生成モデルから生成された出力文に対して，対話行為分類器を導入し，入力した対話行為と同じ対話行為と分類できるものを出力文とするが，10 回，分類を行っても入力した対話行為と同じ対話行為に分類できない場合,「共感」か「あいづち」と分類されたものを出力文とする．また，30 回分類を行っても「入力した対話行為と同じ対話行為ラベル」,「共感」,「あいづち」に分類されない場合，31 回目の生成文を出力文とする．

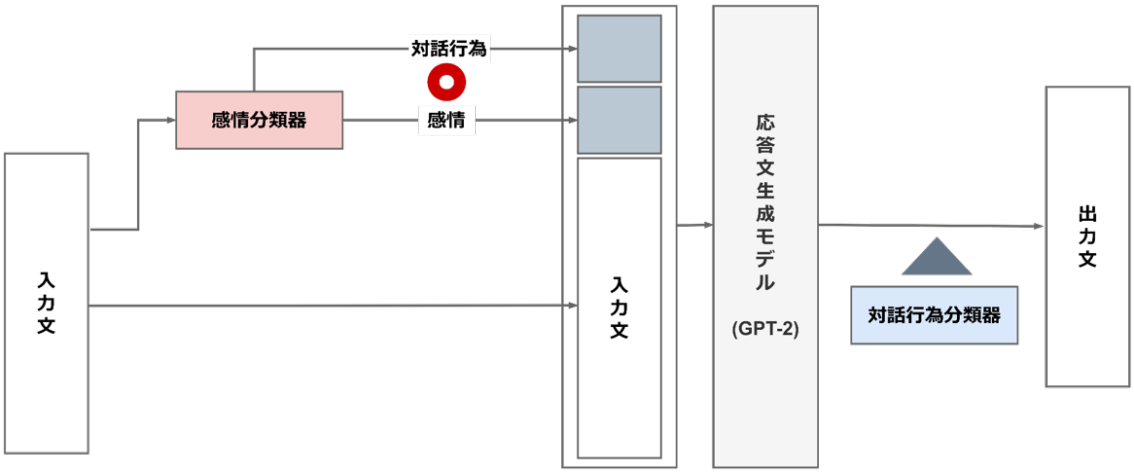


図 11: 発話生成時のフィルタリング: 入力文に対して感情分類器を通して感情ラベルを付与し，あらかじめ定めた感情ラベルと対話行為ラベルの組み合わせを入力文と結合して GPT-2 に入力する

4. 評価

本章では節 4.1 で節 3.1 で作成した感情分類器の評価，節 4.2 で節 4.2 で作成した対話行為分類器の評価，4.3 で提案手法の対話生成結果について評価を行う．実装では，プログラミング言語として Python および transformers 4.30.2，機械学習フレームワークとして PyTorch を利用した．

4.1 感情分類モデルの評価

3.1 節の方法で作成した感情分類モデルについて，ファインチューニングで用いたデータ数は 43200 件の投稿であり，そのうち 1200 件が検証データ，2000 件がテストデータである．バッチサイズは 8, 学習率は 2×10^{-5}，最適化は AdamW [23] として，3 エポックの early-stopping を適用する．

損失の重み付けを行った場合と行わなかった場合の感情分類モデルの比較評価を以下の表 7，8 に示す．表 7 が重みなしの各感情ラベルの精度，表 8 が重みありの各感情ラベルの精度である．また，各感情ラベルの混同行列を図 12 から図 19 に示す.

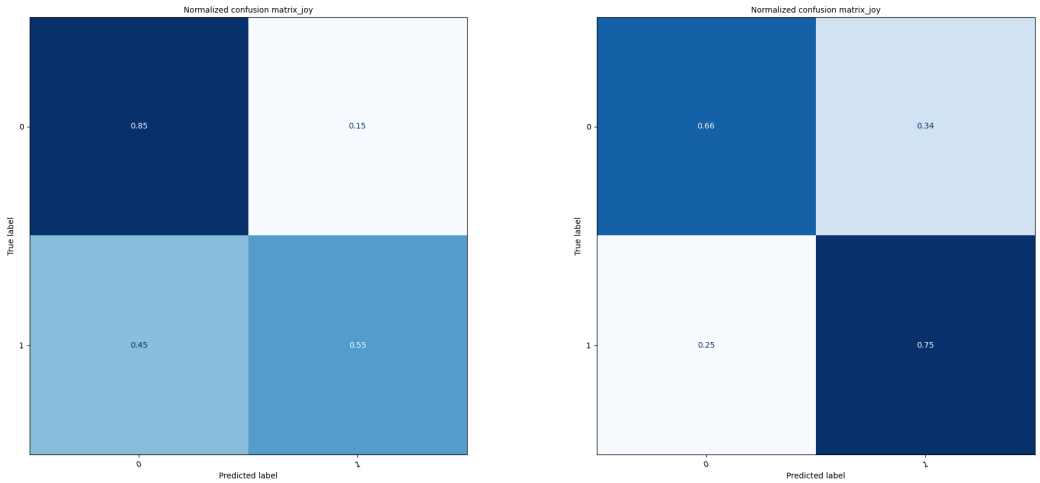
表の marco-F1 の数値から，全ての感情ラベルにおいて重みをつけることによって精度が大幅に高くなっていることがわかる．さらに，各ラベルの混同行列をみてみると，大半の感情ラベルにおいて，重みを用いない感情分類モデルでは，高い確率で，データの不均衡の影響で，1 と分類するべきところを 0 と誤分類してしまっていた．しかし，損失関数に対して重みづけを行うことによって，1 と分類する場合に 0 と誤分類されることが改善されたことがわかる．重みづけを行うことで感情分類モデルの精度が大幅に改善されたが，やはり，正例のデータが少ない感情ラベル (怒りや恐れ) ほど，精度が低くなっており誤分類しやすくなると考えられる．

表 7: 重み付けなしの各感情ラベルの精度

感情ラベル	F1	precision	recall
喜び	0.609	0.680	0.551
悲しみ	0.485	0.556	0.429
期待	0.478	0.624	0.387
驚き	0.269	0.631	0.171
怒り	0.02	0.750	0.0100
恐れ	0.085	0.778	0.0447
嫌悪	0.118	0.676	0.0656
信頼	0.104	0.447	0.0590

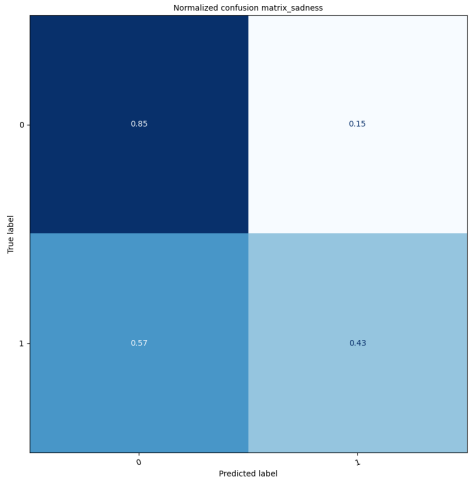
表 8: 重み付けありの各感情ラベルの精度

感情ラベル	F1	precision	recall
喜び	0.644	0.564	0.750
悲しみ	0.542	0.481	0.620
期待	0.593	0.504	0.720
驚き	0.438	0.38	0.516
怒り	0.376	0.385	0.367
恐れ	0.319	0.295	0.348
嫌悪	0.423	0.378	0.378
信頼	0.294	0.192	0.632

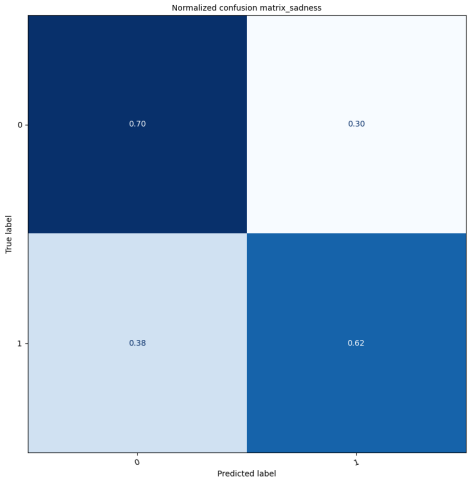


(a) 重みなし　　(b) 重みあり

図 12: 感情ラベル「喜び」の混同行列

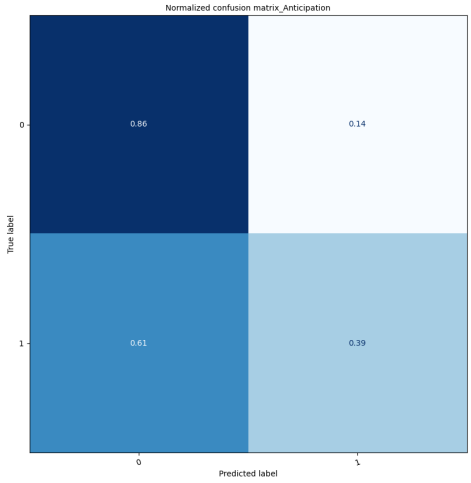


(a) 重みなし

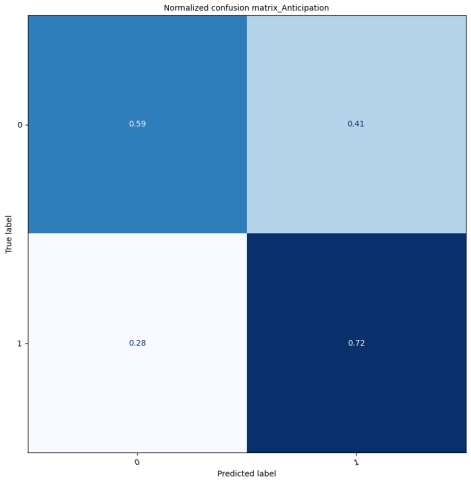


(b) 重みあり

図 13: 感情ラベル「悲しみ」の混同行列

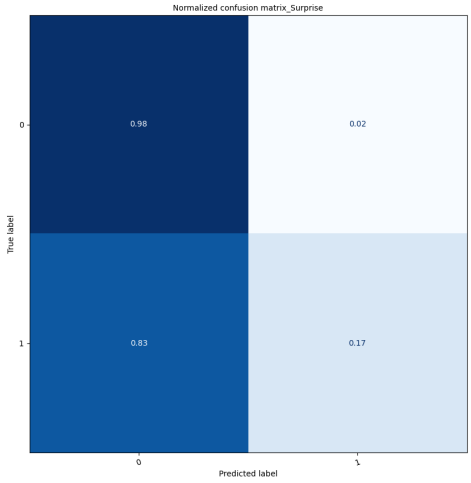


(a) 重みなし

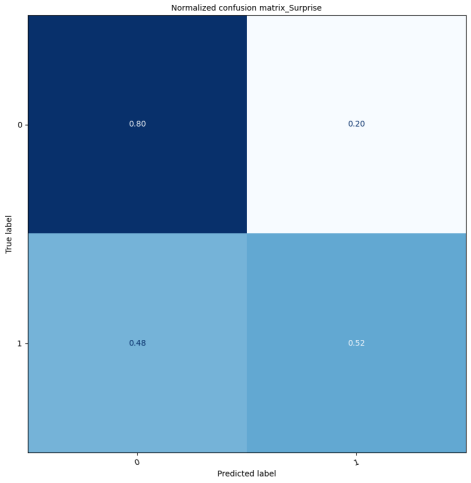


(b) 重みあり

図 14: 感情ラベル「期待」の混同行列

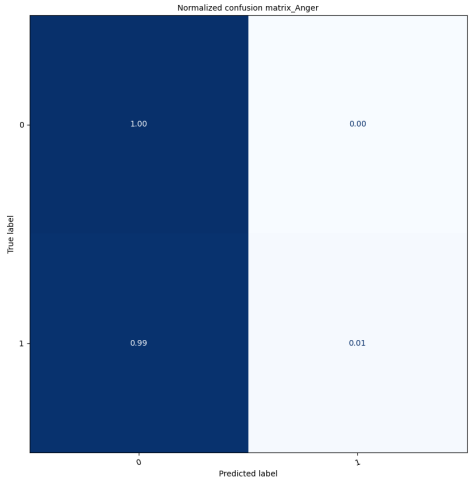


(a) 重みなし

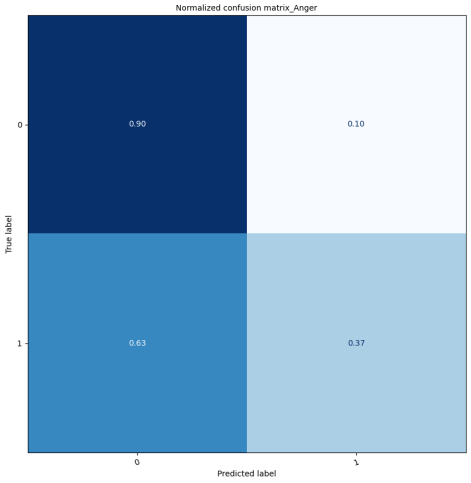


(b) 重みあり

図 15: 感情ラベル「驚き」の混同行列

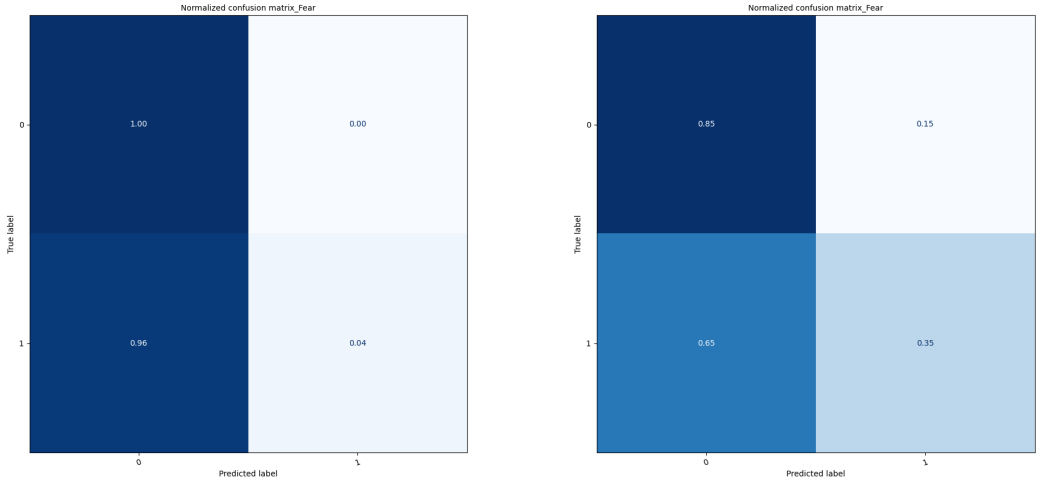


(a) 重みなし



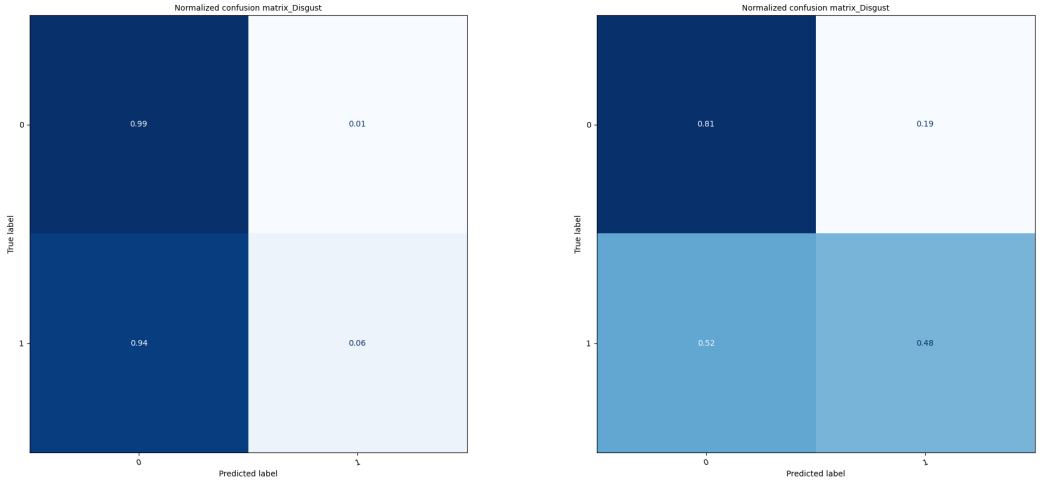
(b) 重みあり

図 16: 感情ラベル「怒り」の混同行列



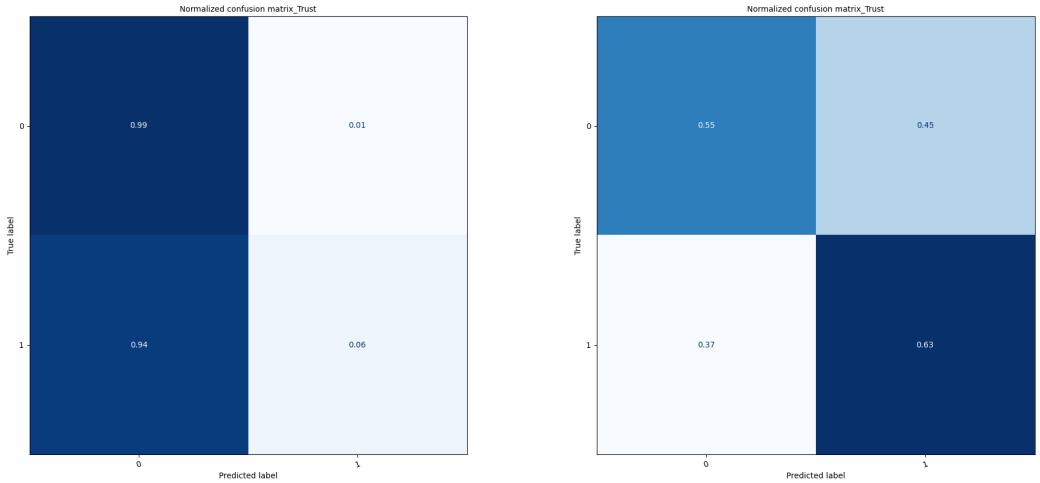
(a) 重みなし　　(b) 重みあり

図 17: 感情ラベル「恐れ」の混同行列



(a) 重みなし　　(b) 重みあり

図 18: 感情ラベル「嫌悪」の混同行列



(a) 重みなし　　(b) 重みあり

図 19: 感情ラベル「信頼」の混同行列

4.2 対話行為分類モデルの評価

節 3.3.1 で用いるために 3.2 節の方法で作成した対話行為推定モデルについて，ファインチューニングで用いたデータ数は 20664 発話であり，2583 件が検証データ，2584 件がテストデータである．ラベルは，自己開示，YN 質問，Wh 質問, YN 応答，Wh 応答，あいづち，フィラー，確認，要求の 9 種類である．バッチサイズは 8, 学習率は 6×10^{-5}，最適化は AdamW[23] として，3 エポックの early-stopping を適用する．

9 クラスの対話行為推定モデルの精度を表 9 に示す．表 9 だけでは各ラベルの精度が分からないので，加えて対話行為ラベルの混同行列を図 20 に示す．

表 9 から全体の精度としては，0.75 の正確さで予測できているが，図 20 で各ラベルごとに見てみると，予測精度にばらつきがあることがわかる．9 種類中 6 種類が 0.6 以上の再現率で予測できているが,「フィラー」の場合は半分近い割合で「あいづち」に誤分類されている．これは「フィラー」と「あいづち」は意味合いが似ている場合があるため，誤分類されやすいと考えられる．また,「確認」では「自己開示」「YN 質問」の二つに 6 割以上誤分類されており,「要求」においても「自己開示」に誤分類されやすいことがわかった.「自己開示」は表 3 のとおり,「挨拶」,「謝罪」なども含まれており，比較的幅広い意味をもつため，誤って分類されやすいと考えられる．

表 9: 9 クラス対話行為推定モデルの精度

macro-F1	accuracy	precision	recall
0.622	0.75	0.62	0.647

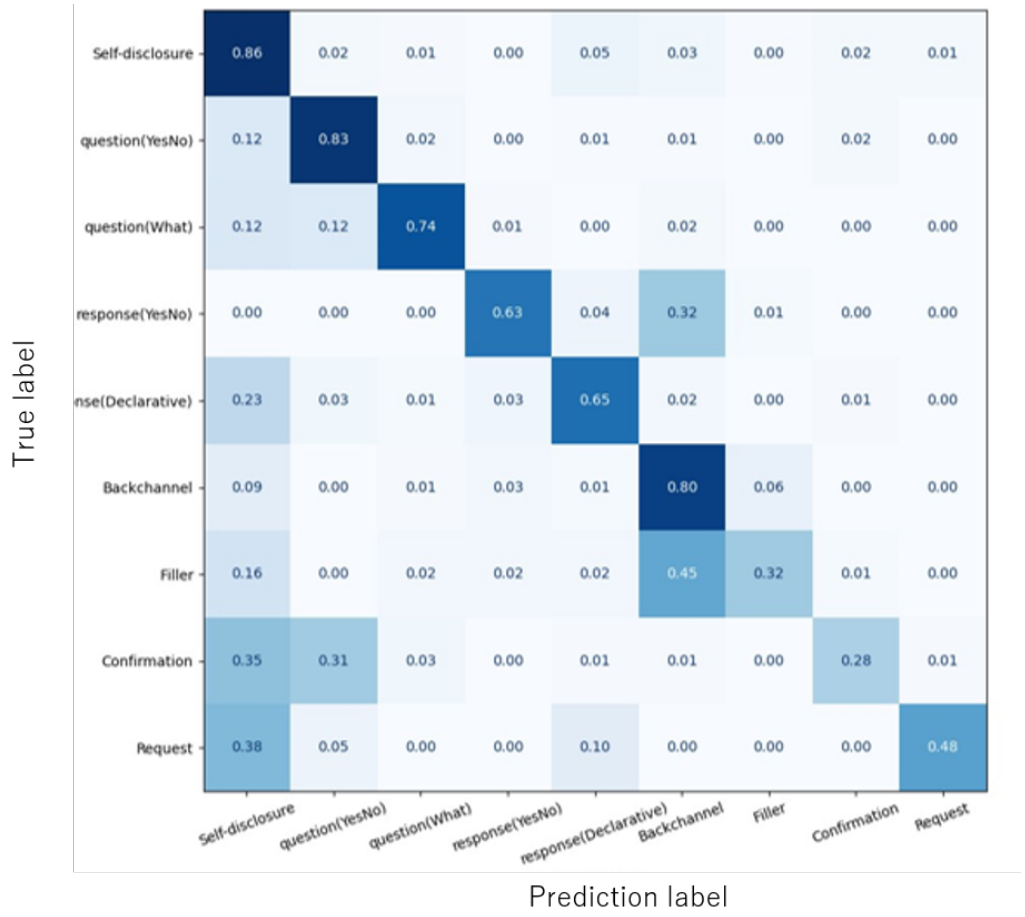


図 20: 9 クラスの対話行為ラベルの混同行列

加えて，節 3.4 で用いるために，同じく節 3.2 の方法で作成した対話行為推定モデルについて，ファインチューニングで用いたデータ数は 20881 発話であり，2610 件が検証データ，2611 件がテストデータである．ラベルは，自己開示，YN 質問，Wh 質問, YN 応答，Wh 応答，あいづち，フィラー，確認，要求，共感の 10 種類である．バッチサイズは 8, 学習率は 2×10^{-5}，最適化は AdamW[23] として，3 エポックの early-stopping を適用する．

10 クラスの対話行為推定モデルの精度を表 10 に示す．表 10 だけでは各ラベルの精度が分からないので，加えて対話行為ラベルの混同行列を図 21 に示す．

表 10: 10 クラスの対話行為推定モデルの精度

macro-F1	accuracy	precision	recall
0.610	0.762	0.655	0.591

図 21: 10 クラスの対話行為ラベルの混同行列

表 9 から全体の精度としては，0.76 の正確さで予測できているが，図 21 で各ラベルごとに見てみると，先ほどの対話行為推定器より，予測精度にばらつきがあることがわかる．先ほどと比べ，クラス数が増え，加えて，データのばらつきの影響によって誤分類されると考えられる．そのため，クラスのデータ数の逆数を重みとして掛けて，損失への寄与率をクラスのデータ数に反比例するように調節する．

各クラスのデータの数を以下グラフを図 22 のに示す．

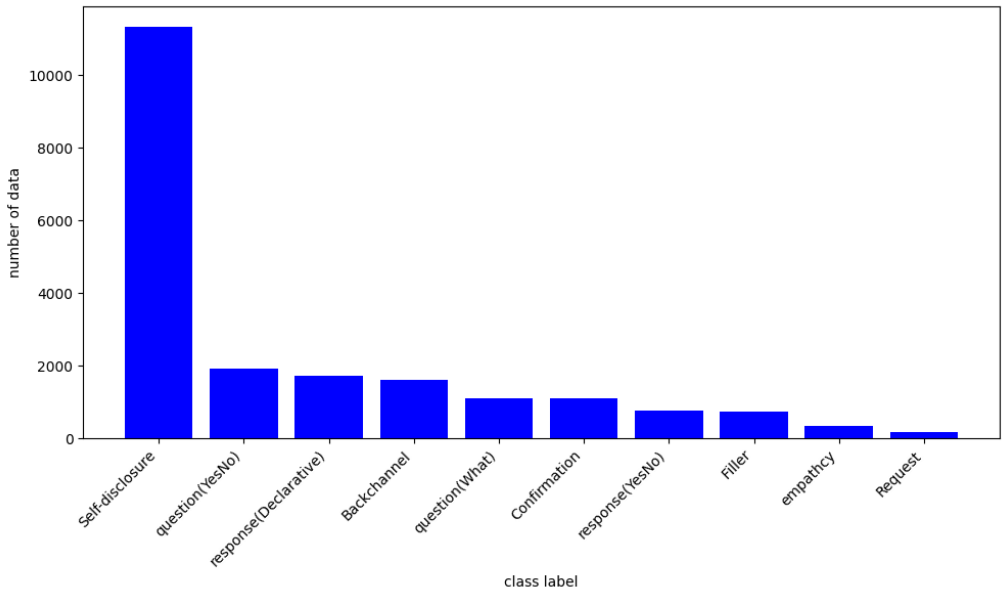


図 22: 対話行為ラベルごとの分布

調整後の 10 種類の対話行為分類器の精度を表 11 に示す．加えて対話行為ラベルの混同行列を図 23 に示す．

表 11: 10 クラスの重みありの対話行為推定モデルの精度

macro-F1	accuracy	precision	recall
0.580	0.66	0.540	0.676

表 11 から見ると，全体の精度は下がったように見えるが，図 23 を見ると「自己開示」と「あいづち」以外の 8 クラスは精度が上がっていることが分かる．こちらの重み付き対話行為分類器を節 3.4 で用いる．

図 23: 10 クラスの重みありの対話行為ラベルの混同行列

4.3 対話生成結果

本節では以下に示す４つのパターンでモデルの比較評価を行う．本研究では，応答生成モデルには日本語版 GPT-2[22] を使用し，節 3.2 の JAIST タグ付き自由対話コーパス [21] でファインチューニングしたものを用いた．

モデル 1: 学習時フィルタリングなし，発話生成時フィルタリングなしモデル　図 24 のように，学習データに対してフィルタリングを行わず，そのまま学習させ，発話生成時にもフィルタリングを行わないモデルである．ファインチューニングで用いたデータ数は 26090 対話であり，訓練データ，検証データ，テストデータへの分割は 8:1.5:0.5 で行った．モデルは 3 エポックで学習させた．応答生成時にはグリードサーチと Top-K サンプリング $(k = 10)$[24]，Top-p サンプリング $(p = 0.9)$[24] を用いて応答生成を行った．また，同一 trigram を複数回生成しないように制約をかけることで，同じ文字列の繰り返しを抑制した．

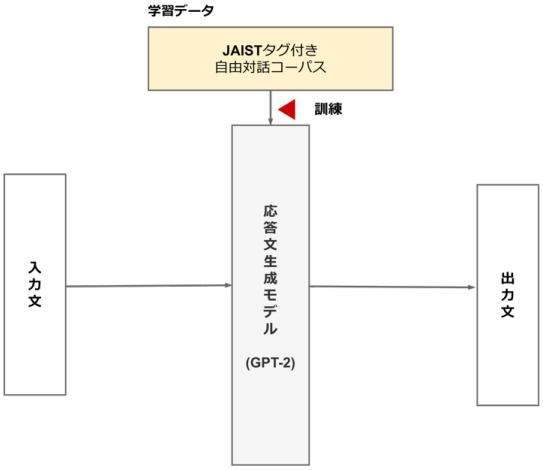


図 24: 学習時フィルタリングなし，発話生成時フィルタリングなしのモデル

モデル 2: 学習時フィルタリングあり，発話生成時フィルタリングなしモデル　図 25 のように，学習データに対して節 3.3 の手法でフィルタリングを行い，発話生成時にはフィルタリングを行わないモデルである．ファインチューニングで用いたデータ数は 4857 対話であり，訓練データ，検証データ，テストデータへの分割は 8:1.5:0.5 で行った．モデルは 3 エポックで学習させた．応答生成時にはグリードサーチと Top-K サンプリング $(k = 10)$[24]，Top-p サンプリング $(p = 0.9)$[24] を用いて応答生成を行った．また，同一 trigram を複数回生成しないように制約をかけることで，同じ文字列の繰り返しを抑制した．

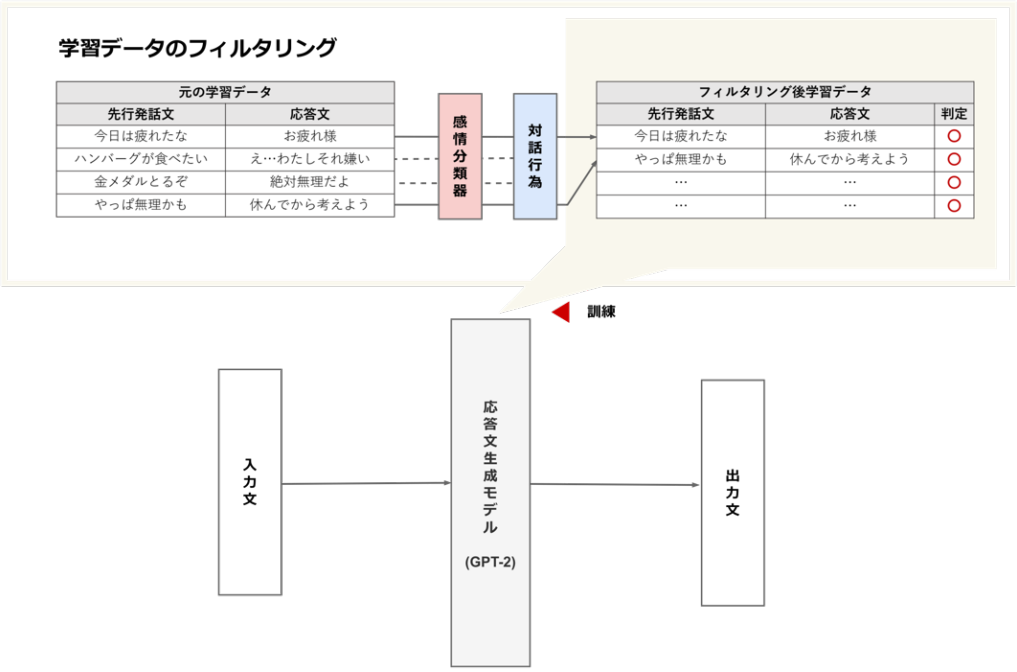


図 25: 学習時フィルタリングあり，発話生成時フィルタリングなしのモデル

モデル 3: 学習時フィルタリングなし，発話生成時フィルタリングありモデル　図 26 のように，学習データに対してフィルタリングを行わず，発話生成時に節 3.4 の手法でフィルタリングを行ったモデルである．ファインチューニングで用いたデータ数は 26090 対話であり，訓練データ，検証データ，テストデータへの分割は 8:1.5:0.5 で行った．モデルは 3 エポックで学習させた．応答生成時にはグリードサーチと Top-K サンプリング $(k = 10)$[24]，Top-p サンプリング $(p = 0.9)$[24] を用いて応答生成を行った．また，同一 trigram を複数回生成しないように制約をかけることで，同じ文字列の繰り返しを抑制した．

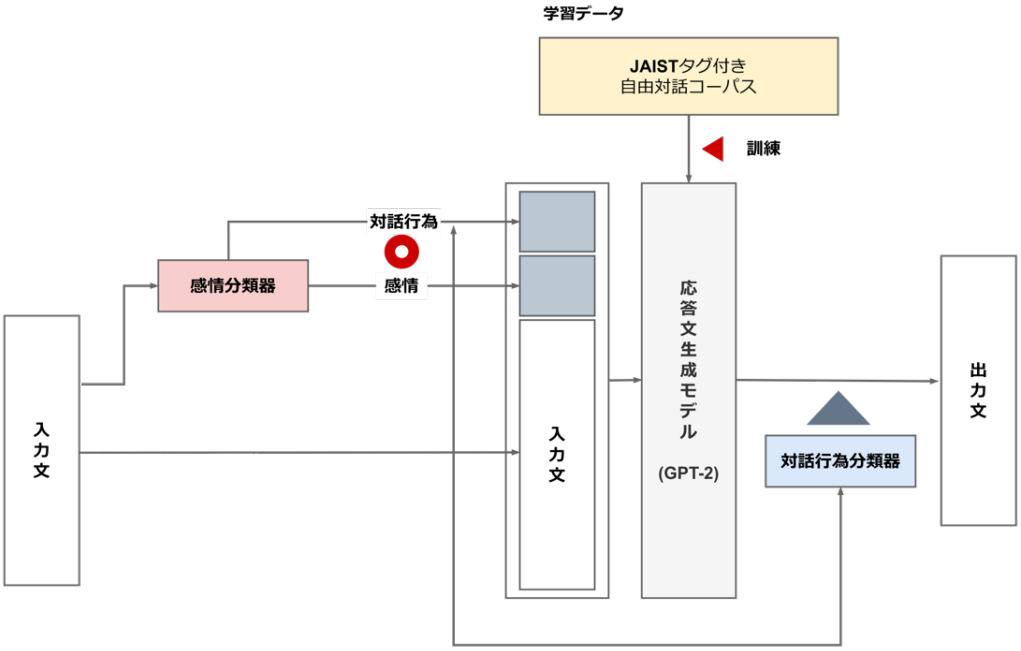


図 26: 学習時フィルタリングなし，発話生成時フィルタリングありのモデル

モデル 4: 学習時フィルタリングあり，発話生成時フィルタリングありモデル　図 8 に示すように，学習データに対して節 3.3 の手法でフィルタリングを行い，発話生成時に節 3.4 の手法でフィルタリングを行ったモデルである．ファインチューニングで用いたデータ数は 4857 対話であり，訓練データ，検証データ，テストデータへの分割は 8:1.5:0.5 で行った．モデルは 3 エポックで学習させた．応答生成時にはグリードサーチと Top-K サンプリング $(k = 10)$[24], Top-p サンプリング $(p = 0.9)$[24] を用いて応答生成を行った．また，同一 trigram を複数回生成しないように制約をかけることで，同じ文字列の繰り返しを抑制した．

4.3.1 評価基準

節 4.3 の 1 から 4 のモデルに対して以下の基準を設けて著者自身と，協力者 2 名，計 3 名で評価を行う．協力者 2 名は各々のフィルタリング方法のテストデータをランダムに 100 個抜き出し，計 400 個の対話に対して評価を行った．各モデルのテストデータに対して応答生成を行い，各評価基準について当てはまる場合を 1，当てはまらない場合を 0 と評価した．

- 流暢性：文として不自然ではないか
- 関連性：文脈と矛盾しない応答か
- 共感性：共感的な応答であるか

表 12 に著者自身で評価した場合の結果を示す．加えて，表 13 に協力者 1，表 14 に協力者 2 が評価した場合の結果を示す．結果の傾向として，モデル 1，すなわち学習時フィルタリングなし，発話生成時フィルタリングなしのモデルとモデル 2，すなわち学習時フィルタリングあり，発話生成時フィルタリングなしのモデルを比較するとモデル 2 が「流暢性」,「関連性」,「共感性」の全てにおいて高い値を示した．これにより，ユーザーの感情に寄り添った共感性を学習データとして得ることで，精度の高い応答文が得られることがわかる．そして，モデル 1 とモデル 3，すなわち学習時フィルタリングなし，発話生成時フィルタリングありのモデルを比較するとモデル 3 が全ての評価基準において，高い値を示した．これにより，発話ごとにユーザーの感情状態を推定し，そのユーザの感情状態から，システムがどのような応答をするべきか対話行為として与え，動的に生成発話を制御することで，対話の文脈に沿った共感的な応答文を生成すること可能になることが結果として得られた．モデル 2 とモデル 3 を比較すると，モデル 2 が全ての評価基準において，高い値を示した．これは，発話生成時に応答を制御するより，学習時のフィルタリングによって，自身の理想とする学習データを作成し，それを使用するほうが，適切な応答文を生成できることが考えられる．また，提案手法においては全評価者において，他のどのモデルよりも「流暢性」,「関連性」,「共感性」の全てにおいて高い値を示した．学習時と発話生成時の両方でフィルタリングを行うことで，より共感的な応答文を生成できることがわかる．

表 12: 主観評価の結果 (各評価基準ごとに 1 となった応答文の数をデータ数で割り，それを確率として表現)

モデル	データ数	流暢性	関連性	共感性
モデル 1	779	0.426	0.293	0.129
モデル 2	332	0.705	0.512	0.223
モデル 3	719	0.490	0.453	0.163
モデル 4(提案手法)	336	0.789	0.655	0.235

表 13: 協力者 1 の評価結果 (各評価基準ごとに 1 となった応答文の数をデータ数で割り，それを確率として表現)

モデル	データ数	流暢性	関連性	共感性
モデル 1	100	0.63	0.48	0.20
モデル 2	100	0.75	0.59	0.35
モデル 3	100	0.54	0.51	0.41
モデル 4(提案手法)	100	0.76	0.70	0.57

表 14: 協力者 2 の評価結果 (各評価基準ごとに 1 となった応答文の数をデータ数で割り，それを確率として表現)

モデル	データ数	流暢性	関連性	共感性
モデル 1	100	0.70	0.50	0.36
モデル 2	100	0.63	0.52	0.44
モデル 3	100	0.48	0.44	0.40
モデル 4(提案手法)	100	0.80	0.67	0.45

またここで，協力者 2 名の一致率ならびにカッパ係数を表 15，16 に示す。表 16 を見ると，関連性においては他の項目と比べ高い一致度になっていることがわかる。共感性においては他の項目と比べて低い一致度になっており，やはり，共感的な応答かどうかの評価をすると個人差が出てしまうことがわかる。

表 15: 協力者 2 の一致度

モデル	流暢性	関連性	共感性
モデル 1	0.73	0.74	0.72
モデル 2	0.80	0.75	0.59
モデル 3	0.84	0.81	0.75
モデル 4(提案手法)	0.82	0.75	0.66

表 16: 協力者 2 のカッパ係数

モデル	流暢性	関連性	共感性
モデル 1	0.39	0.48	0.33
モデル 2	0.54	0.50	0.15
モデル 3	0.68	0.62	0.48
モデル 4(提案手法)	0.48	0.42	0.33

4.3.2 生成例

表 17 に具体的な生成応答の例を示す．ここでは節 4.3 のモデル 1，モデル 2，モデル 3，提案手法であるモデル 4 の生成結果と，参照応答を掲載している．

表 17 の生成例からフィルタリングを行っていないモデル 1 は，文脈に沿った応答文が生成できておらず，共感的な応答とはいえないことがわかる．学習データのフィルタリングを行ったモデル 2 はモデル 1 と比べると，共感的な応答が生成できているが，ところどころ文脈に沿った応答ができていないことがわかる．提案手法では，どの例においても先行発話の感情を踏まえた共感的な応答が出来ていることがわかる．例えば，先行発話の「この年になって，遅ればせながらピアノを習い始めました!」に対して提案手法であるモデル 4 の応答文では「あーすごいねぇ」と返答しており，ユーザの「喜び」の感情を踏まえた共感的な応答だととらえることができる．また,「飼ってた犬，ついに死んじゃった」という先行発話に対しては,「かわいそうにねぇと思って」と応答文を生成しており，ユーザの「悲しみ」の感情と同じ感情での返答ができていることがわかる．このように，提案手法であるモデル 4 ではユーザの感情に共感できるような応答生成文が多くみられる．しかしながら，モデル 1 やモデル 2 と比べ，発話生成時のフィルタリングを行った，モデル 3 や提案手法の応答文は発話が短いことがわかる．テストデータの結果を見ても「そうなんだ」や「すごい」等の短調な文が高頻度で出現しており，応答文の多様性があるかという観点に関してはあまりよくない結果となった．

表 17: 対話生成例

先行発話	この年になって，遅ればせながらピアノを習い始めました！
参照応答	すてき！レッスンはどんな感じ？
モデル 1	すごい．なんかちょっと何か違う気がするんだけどねー
モデル 2	ねー．なんかもうちょっと早く教えてくれればよかったの
モデル 3	あー．すごいねえ
モデル 4(提案手法)	あー．すごいねえ
先行発話	飼ってた犬，ついに死んじゃった．
参照応答	それは悲しいね．
モデル 1	そうよねーやっぱり動物好きだったらわかるわけどさあ
モデル 2	いやーほんとにすごいショックだったわねえ (涙) もう
モデル 3	あーそっかそうか
モデル 4(提案手法)	かわいそうにねえと思って
先行発話	昨夜とっても恐ろしい夢を見たの．なんと歯が全部抜ける夢！
参照応答	それは，とても恐ろしい夢だね！起きたとき，焦ったでしょ？
モデル 1	あっ，そっかーと思ってさあちょっと
モデル 2	うーん，やっぱりねえ
モデル 3	ね
モデル 4(提案手法)	すごい．
先行発話	そういえば聞いた？あの子プロポーズされたらしいよ．
参照応答	ええっ，そんな気全然なかったよね．
モデル 1	えー，もうちょっと待っててくださいねとか言ってたら
モデル 2	うん，でも私も今はさーなんか結婚しない
モデル 3	うん，わかるーね
モデル 4(提案手法)	あー，すごいね

5. 結論

5.1 まとめ

本論文では，対話の中で共感した反応を返し，話したいことを話しやすいように誘導してくれる応答文を生成するために，二つの段階で研究を進めた．一段階目の学習時のフィルタリングでは，適切な発話・応答の対応をあらかじめ言語モデルとして獲得し，二段階目の発話生成時のフィルタリングでは，対話進行時にもユーザーの感情状態を推定し，動的に生成発話を制御することで，対話の文脈に沿った適切な応答を生成できる対話システムを提案した．提案手法を用いた場合，既存手法でよくみられる，学習時のフィルタリングのみのモデルで応答文生成を行ったときと比べ，ユーザーの感情を踏まえた応答文生成が可能になることを確認した．一方で，相手の感情に共感した応答だけでなく，ユーザーが話しやすくなるような，質問や自己開示が含まれた応答生成を期待していたが，結果としては単調な応答文が生成される傾向がみられた．

5.2 今後の課題

結果を踏まえて，今後取り組むべき課題として以下のような項目がある．

学習時のフィルタリング　本研究では学習時のフィルタリングを行うことで，ユーザーの感情に寄り添った共感性を学習データとして得ることを可能にした．しかし，学習時のフィルタリングを行うと，各感情ラベルと対話行為ラベルのペアのデータの数に偏りが出てしまい，データが少ないペアの応答文を生成させたい場合，想定した応答文が生成できない場合が考えられる．そのため，データが少ないペアにはデータを追加する等の方法で各ペアのデータ数を均一にする必要がある．また，本研究で用いた対話行為タグ付きデータセットの「自己開示」ラベルは，挨拶や謝罪も含まれており，幅広い意味を持っている．そのため，生成すべき応答文を「自己開示」と指定した場合，事実を述べるのか，自身の考えを述べるのか，挨拶をするのかと，どのような文章を生成するべきなのか曖昧になってしまう．そこで，より詳細な対話行為ラベルがタグ付けされたデータセットを用いることによって，生成すべき応答文が明確になるのではないかと考えられる．

発話生成時のフィルタリング　相手に話したいと思ってもらえるような対話システムを実現するには，相手の感情に共感した応答だけでなく，ユーザーが話しやすくなるような，質問や自身の意見が含まれた応答生成が必要である．しかし，発話生成時のフィルタリングを行った場合，対話行為に「質問」と指定しても，質問文が生成されず，単調な応答文が生成される傾向がみられた．なぜそのような問題が生じるのか原因を突き止める必要がある．

感情分類モデルの分類精度　本研究の感情分類モデルは，入力された文章に対して入力文が何の感情であるかをマルチラベルで推定を行った．感情分類モデルに用いたデータセットはデータの不均衡が生じており，それを対処するために，ラベルごと正例の割合を利用した損失の重みづけを行った．重みづけを行うことによって，重みづけを行わない場合を比べ，精度は大幅に改善したが，やはり，正例が少ない感情ラベルほど精度が低くなっており誤分類する割合は高い．また,「怒り」は第二感情と呼ばれ,「怒り」の前に必ず「悔しさ」や「悲しみ」などの第一感情が存在する．そのような第一感情を踏まえることができる，感情分類器を作成することによって，よりユーザーの感情状態を正確に推定することで共感的な応答文の生成につながるのではないかと考える．

生成された応答文に対しての評価　本論文では，生成された応答文に対して，BLEU や BERT-Score のような自動評価と，多数のユーザーによる人手評価が行えていない．今後，対話システムの性能を示すために，数値化された指標を提供し，客観的にモデルを評価できる自動評価と，主観的な要素や文脈を考慮できる人で評価行う必要がある．

謝辞

本研究に取り組むにあたり，多くの方のご支援を賜りました．周囲の方々のご協力のおかげで，修士課程の 2 年間を実りあるものにすることができました．まず初めに，川嶋宏彰教授には研究テーマや提案手法，論文の執筆などを含め，様々なご指摘，ご助言をいただきました．心から深く感謝を申し上げます．研究室の皆様には，日頃から様々な形でお世話になりました．本当にありがとうございました．

参考文献

[1] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 2017.

[2] 荒瀬由紀, 岡崎直観, 鈴木潤, 鶴岡慶雅, 宮尾祐介. IT Text 自然言語処理の基礎. オーム社, 2022.

[3] 近江崇宏, 金田健太郎, 森長誠, 江間見亜利. BERT による自然言語処理入門: Transformers を使った実践プログラミング. ストックマーク株式会社, 2021.

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[5] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. 2018.

[6] Hannah Rashkin, Eric Michael Smith, Margaret Li, and Y-Lan Boureau. Towards empathetic open-domain conversation models: A new benchmark and dataset. In Anna Korhonen, David Traum, and Lluís Màrquez, editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 5370–5381, 2019.

[7] 内閣府. 令和元年版高齢社会白書 (全体版). 地域別にみた高齢化, https://www8. cao. go. jp/kourei/white paper/w−2019/zenbun/pdf/1s1s_04. pdf, 参照 (2020−06−30), 2019.

[8] Laura Fratiglioni, Hui-Xin Wang, Kjerstin Ericsson, Margaret Maytan, and Bengt Winblad. Influence of social network on occurrence of dementia: a community-based longitudinal study. *The lancet*, Vol. 355, No. 9212, pp. 1315–1319, 2000.

[9] Jane S Saczynski, Lisa A Pfeifer, Kamal Masaki, Esther SC Korf, Danielle Laurin, Lon White, and Lenore J Launer. The effect of social engagement on incident dementia: the honolulu-asia aging study. *American journal of epidemiology*, Vol. 163, No. 5, pp. 433–440, 2006.

[10] 杉山弘晃, 中村賢治, 原田欣宏, 大口達也, 堀口美奈子, 佐川祥吾. 認知症患者を対象とした対話エージェントとのしりとりゲームによる認知機能維持効果の検証. 人工知能学会

全国大会論文集 第 34 回 (2020), pp. 1D4GS1302–1D4GS1302. 一般社団法人 人工知能学会, 2020.

[11] 中野幹生. 身近になった対話システム: 1. 対話システムを知ろう-自然言語による機械と人間とのコミュニケーション-(epub 版). 情報処理, Vol. 62, No. 10, pp. e1–e6, 2021.

[12] 本間健, 陰山宗一, 石田真捺, 内田尚和, 森一, 岩山真, 十河泰弘. 感情的応答を表出する end-to-end 対話システムの学習方法の検討. 人工知能学会全国大会論文集 第 36 回 (2022), pp. 3Yin253–3Yin253. 一般社団法人 人工知能学会, 2022.

[13] 杉山弘晃, 成松宏美, 水上雅博, 有本庸浩, 千葉祐弥, 目黒豊美, 中嶋秀治. Transformer encoder-decoder モデルベース雑談対話システムの学習方法に対する主観評価の変動分析. 人工知能学会全国大会論文集 第 35 回 (2021), pp. 4E1OS11a03–4E1OS11a03. 一般社団法人 人工知能学会, 2021.

[14] Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. Personalizing dialogue agents: I have a dog, do you have pets too? In Iryna Gurevych and Yusuke Miyao, editors, *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2204–2213, Melbourne, Australia, July 2018. Association for Computational Linguistics.

[15] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, Vol. 1, No. 8, p. 9, 2019.

[16] Yuhan Liu, Jun Gao, Jiachen Du, Lanjun Zhou, and Ruifeng Xu. Empathetic response generation with state management, 2022.

[17] 梶原智之. Wrime: 主観と客観の感情強度を付与した日本語データセット. 自然言語処理, Vol. 28, No. 3, pp. 907–912, 2021.

[18] Robert Plutchik. A general psychoevolutionary theory of emotion. In *Theories of emotion*, pp. 3–33. Elsevier, 1980.

[19] 泉春乃, 加藤昇平. Attention 機構による低頻度な対話行為の特徴を捉えた対話行為推定. 人工知能学会研究会資料 言語・音声理解と対話処理研究会 87 回 (2019/12), p. 28. 一般社団法人 人工知能学会, 2019.

[20] 目黒豊美, 東中竜一郎, 堂坂浩二, 南泰浩ほか. 聞き役対話の分析および分析に基づいた対話制御部の構築. 情報処理学会論文誌, Vol. 53, No. 12, pp. 2787–2801, 2012.

[21] 福岡知隆, 白井清昭. 対話行為に固有の特徴を考慮した自由対話システムにおける対話行為推定. 自然言語処理, Vol. 24, No. 4, pp. 523–547, 2017.

[22] 趙天雨, 沢田慶. 日本語自然言語処理における事前学習モデルの公開. 人工知能学会研究会資料 言語・音声理解と対話処理研究会 93 回 (2021/11), pp. 169–170. 一般社団法人人工知能学会, 2021.

[23] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[24] Lewis Tunstall, Leandro von Werra, Thomas Wolf, 中山光樹. 機械学習エンジニアのための Transformers ―最先端の自然言語処理ライブラリによるモデル開発. オライリージャパン, 2022.