DEIM2021 F14-4

文の生成要因に着目したライブコメントの類似度計算と意見集約

吉田　司[†,††]　大塚　淳史[†,††]　野本　済央[†,††]　小澤　史朗[†,††]　小橋川　哲[††]

† 日本電信電話株式会社 NTT デジタルツインコンピューティング研究センタ
〒 108–0023 東京都港区芝浦 3 丁目 4-1 グランパークタワー 33F
†† 日本電信電話株式会社 NTT メディアインテリジェンス研究所
〒 239–0847 神奈川県横須賀市光の丘 1-1
E-mail:
†{tsukasa.yoshida.zu,atsushi.otsuka.vs,narichika.nomoto.ds,shiro.ozawa.zs,satoshi.kobashikawa.he}@hco.ntt.co.jp

あらまし　本研究では動画ライブ配信のチャット機能や SNS への実況投稿などに焦点を当て，投稿されるコメント文から配信者へフィードバック可能な情報にリアルタイムで集約・整理することを目的とする．従来のテキスト分類手法では 2 つの文の特徴量ベクトルの類似度計算によりそれらを直接比較して文の意味の類似性を測るが，この場合どのような観点で文が類似しているか不明瞭であり，分類のためにその類似性を意図的に制御することも難しい．そこで本研究では 2 つの文を直接比較するのではなく，文の生成要因となる文を用いて間接的に比較することで類似度を測ることを提案する．間接比較に介在させる間接文を変更することで，主張部が同じ文の類似度や部分的な類似性を持つ文の類似度計算が可能となる．本手法が様々な類似性を計算でき，コメントの分類において有効であることを実験で示す.

キーワード　類似尺度, ライブコメント, 情報検索, 分類

1　はじめに

近年インターネットの拡大により，様々なプラットフォームにおいてチャット投稿やコメント投稿などによる多人数のコミュニケーションが日々大量に行われている．例えば，動画ライブ配信におけるチャット欄やコメント欄への書き込み，コミュニケーションツールにおけるチャット，SNS への投稿テキスト，通販サイトの商品のレビュー文などが挙げられる．これらのテキストから有益な情報や法則性を得る方法がデータマイニングやオピニオンマイニングなどの分野で考案されてきた [1–4].

本論文では動画ライブ配信におけるライブコメントに焦点を当てる．ここでのライブコメントとはライブ配信に関して動画配信サイトのチャット欄に書き込まれるコメントやその他の SNS などで実況投稿されるコメントを指す．ライブコメントとして例えば質問や意見が投稿された場合，配信者がそれに応答することでコミュニケーションの双方向性が向上し，配信の満足度向上が期待できる．しかしながら，ライブコメントが大量に取得可能な場合はコメントを追い切れなくなったり，応答すべきコメントを見つけ出すことが難しくなったりすることも少なくない.

本研究ではこのライブコメントから質問や意見などの情報をリアルタイムで適切に集約・整理することを目的とする．これにより，配信者にフィードバック可能な情報を集めることが目標となる．例えば質問コメントのみを集めて配信者に提示したり，あるコメントに対して同様のコメントを探して同じ意見がどの程度出ているかをまとめたりすることが考えられる．今回のタスクはリアルタイムでの処理が重要となる．また，対象のコメントは文章長が短いため，これに適した分類や検索方法が必要となる.

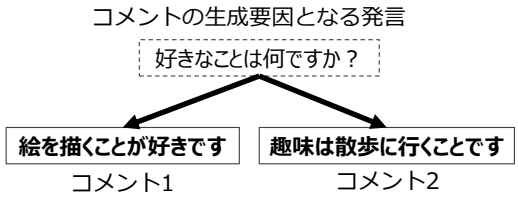

図 1　コメントとその生成要因となる文．本論文では生成要因となる文を用いて間接的なコメント間の類似度を測ることを提案する．生成要因となる文と生成結果のコメントは会話的連続性があることに注意する.

従来のテキストの分類や検索手法ではテキストを直接扱わず，tf-idf などで変換した特徴量ベクトルを用いる [5]．近年では特徴量ベクトルへの変換方法としてニューラルネットワークを用いた word2vec や BERT などの方法も考案されている [6, 7]. しかしながらこれらはテキストを高次元のベクトル空間に埋め込み，それらの差のノルムやコサイン類似度といった量を用いて文の類似度を測るため，どのような観点で文が類似しているか不明瞭である．また，分類したい問題に応じてその類似性の観点を意図的に制御することも難しい.

従来手法の分類結果などを解釈する上で，どのようなコメントが似ていると考えるべきなのか，コメントの類似性とは何かという根本的な問題に直面する．コメントを分類する際にはコメントが生じた背景情報を踏まえるべきであるから，類似の基準の一つとしてコメントが生じた理由の一致が挙げられる．例

えば図 1 に示すように，そのままでは一見関係がないコメントでも，そのコメントが生じる理由が一致している場合は類似していると捉えるべきと考えられる．

そこで本研究ではこの考えを押し広め，文の類似性として生成要因の一致を指標とすることを提案する．従来の類似度と提案する類似度の計算方法の特に大きく異なる点は，2 文を直接比較するか間接比較するかである．すなわち，従来の類似度は 2 つの文を直接的に比較することで意味的類似度を測る．一方で提案手法は図 1 のように生成要因となるような文（間接文）を介在させることで類似度を計算する．このように間接文をつくることでその類似性に解釈性をもたせられ，さらにこの間接文を変更することで類似性の意味を変更できる．これはどのような観点で類似性を測るかを制御できることを意味し，分類の際に役立つ．特に本論文では間接文に疑問文を用い，具体的な質問や抽象的な質問を用いることで類似尺度を制御する．また，間接類似度の計算方法として自然言語処理の機械学習モデルである NSP (Next Sentence Prediction) [7] を用いる．これらを用いて，大量のコメント文の中から入力された疑問文に対して適切な回答となるコメント文を検索しながら，類似コメントをまとめる方法を与える．

本研究の貢献は次の通りである．

- 間接比較を用いた文の類似性の指標を与えた．この類似度は間接文によって類似性を人間が解釈しやすく，間接文を変更することで類似性の意味を変更できる．
- 提案した類似度指標の具体的な計算方法として NSP モデルを挙げ，これを用いた類似コメントの検索アルゴリズムを与えた．
- 間接比較の類似度の有効性を NSP を用いた検索アルゴリズムを用いて検証した．

2 関 連 研 究

2.1 テキストのベクトル表現

単純なドキュメントの埋め込み方法として，BoW (Bag of Words) やその拡張となる tf-idf などがある [5]．これらはベクトルを構成するために事前に出現する単語をすべて知っておく必要がある．ライブコメントではどのような単語が発生するか事前にはわからないため，この埋め込みは適当でない．

ニューラルネットワークを用いた埋め込み手法として word2vec がある [6]．これは単語をベクトルに変換する機構であり，似たような単語を似たベクトルに変換することができる．これを用いて検索クエリに含まれる単語とドキュメントに含まれる単語をベクトル化し，その距離を取ることで言い換えに頑健な検索を行うことができる．しかしながらこの方法は単語をすべてベクトルに変換する必要があり，計算コストが高く，リアルタイムでの検索には適さない．他の単語埋め込みの手法としては Glove [8] や fastText [9], ELMo [10] などがある．

近年，自然言語処理を行うための汎用言語処理モデルとして BERT (Bidirectional Encoder Representations from Transformers) が注目されており，事前学習モデルとして事前に大量のデータを学習しておくことで様々なタスクで活用できることが報告されている [7]．NSP は BERT を使ったモデルの一つであり，BERT の最終層に線形変換のヘッドを付け加え，転移学習することで実現される．BERT は本来の使用方法として転移学習が念頭に置かれているが，BERT の出力をそのまま使用することもでき，BERT の最終層の出力を使うことで入力テキストの全体としてのベクトル表現を得ることができる．

2.2 テキストの意味的情報の利用

分類や検索などの性能を高めるためにはテキストをベクトルに変換する際にテキストの意味的情報をタスクに応じてうまく埋め込む必要がある．Li らは短いテキストの分類を目的とし，BoW の枠組みを拡張した意味的情報を利用できる埋め込み方法を提案している [11]．Kumar らはマイクロブログの投稿記事のクラスタリングを行うために，LDA (Latent Dirichlet Allocation) を拡張し，意味的情報を用いたオンライン学習型のクラスタリングを提案している [12]．He らは文章で述べられていない情報，特に皮肉表現などを解釈するためのニューラルネットによる埋め込みモデルを提案している [13]．

2.3 データマイニング

Mukherjee らは旅行のレビュー文を対象として，複数のレビュー文を個人に合わせた形で要約する方法を提案している [2]．Tekumalla らは Twitter から COVID-19 に関して有効性があるかもしれない薬に関して情報収集することを目的とし，事前に病名や薬に関する用語のミススペルを作り，それを検索に用いることで使用可能なデータが増えることを明らかにしている [3]．Maio らは事前学習モデルの学習データが大量に必要であることから，オピニオンマイニングを行うための学習データを大量に生成する方法を提案している [4]．

3 提 案 手 法

従来の文の類似性は直接比較を用いて測られていた．本研究では文の類似性を間接比較を用いて測ることを提案する．従来手法と提案手法の文の類似度計算を模式的にまとめたのが図 2 である．ここでは間接比較の類似性の考え方とその特徴について述べる．

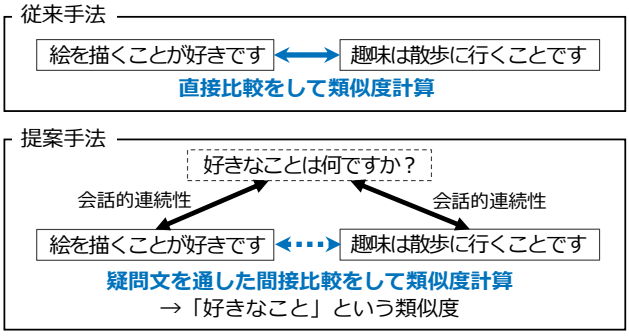

図 2　文の直接比較と間接比較による類似度計算

3.1 直接比較の類似尺度

提案手法の間接比較の説明の前に，まず直接比較の考え方を説明する．これは提案手法の間接比較を理解するのに役立つ．

文を比較する際に現在主流となっている方法はテキストをベクトルに変換し，そのベクトルの差のノルムやコサイン類似度などを計算することで類似度を測る方法である．例えば，ベクトルの差のノルムを用いる場合を考える．文 1, 文 2 のベクトル表現をそれぞれ $\boldsymbol{v}_1, \boldsymbol{v_2}$ としたとき，その類似度 s_D は

$$s_D(\text{文 } 1, \text{文 } 2) = -\|\boldsymbol{v}_1 - \boldsymbol{v}_2\|$$

で与えられる．ここで $\|\cdot\|$ は L_2 ノルムである．ノルムにマイナス符号がついているのは距離を類似度の指標として変換するためであり，本質的ではないことに注意する．また，別の s_D の取り方としてノルム値が閾値 θ_D 以下のときに類似性があるものとして 1, それ以外は 0 を返す関数

$$s_D(\text{文 } 1, \text{文 } 2) = \begin{cases} 1 & (\|\boldsymbol{v}_1 - \boldsymbol{v}_2\| < \theta_D) \\ 0 & (\text{other}) \end{cases} \tag{1}$$

として定義することが考えられる．この定義は分類や検索を行う際に使う指標である．

このとき，文 1, 文 2 の比較を行う際にそれ以外の文は存在しない．ここからこの類似度は直接比較を行っていると考えられる．直接比較の問題点はその類似度をノルムやコサイン類似度といった量を用いて測るため，どのような観点で文が類似しているか不明瞭なことである．また，分類したい問題に応じてその類似性の観点を意図的に制御することも難しい．さらに，文の word2vec や BERT を用いたベクトル表現はノルムやコサイン類似度を求めるために最適化された表現ではないため，その利用には疑問が残る．つまり，学習モデル (word2vec や BERT) の学習目的とモデルの使用目的が一致しておらず，その点で最適性がない．

3.2 間接比較の類似尺度

前述の直接比較の類似尺度の問題点を解決するために，本研究では間接比較の類似尺度を提案する．1 章の図 1 でも説明したとおり，コメントを分類する際にはコメントが生じた背景情報を踏まえるべきである．そこで文の生成要因を考えることで 2 つの文の類似性を測ることができると考えられる．これが生成要因を間に挟んだ間接比較である．この考え方は “文が使われた背景状況によって文の意味は変わるため，背景状況を用いることで意味的類似性を測ることができる” という思想に基づいている．

文の生成要因となる事象は多くあるが，本研究では最も簡単な生成要因として会話的疑問文を考える．会話的疑問文は測りたい類似性に応じて適宜作成する．この疑問文と与えられた文の会話的連続性をとることで，その生成要因との因果関係を測り，これを介して与えられた 2 つの文の類似度を計算する．会話的連続性は次のように定義する．例えば，文 1「好きなことは何ですか？」に対して，文 2「絵を描くことが好きです」は会話として成り立つため，会話的連続性が高いとする．一方で文 1「好きなことは何ですか？」, 文 2「傘を持ち歩くといいかも」は会話として成り立たないため，会話的連続性が低いとする．本研究では会話的連続性を NSP を用いて計算する．

具体的な類似度計算の方法について述べる．まず文 1, 文 2 のそれぞれを疑問文と比較し，その会話的連続性 c_1, c_2 をとる．次にその統計量をもって類似性 s_I を計算する．統計量には例えば c_1, c_2 がどちらもある閾値 θ_I 以上になった場合に類似性があるものとして 1, それ以外は 0 を返す関数

$$s_I(\text{文 } 1, \text{文 } 2) = \begin{cases} 1 & (c_1 \geqq \theta_I \wedge c_2 \geqq \theta_I) \\ 0 & (\text{other}) \end{cases} \tag{2}$$

として定義する．

この手法ではこの疑問文から類似性の観点の解釈することができ，さらに疑問文を変更すれば分類したい問題に応じてその類似性の観点を意図的に制御することができる．また直接比較と比べて，学習モデル (NSP) の学習目的と今回のモデルの使用目的は一致しており，その意味で最適性がある．

3.3 意味の類似度の強弱のクラス

間接比較の類似尺度では間接文の疑問文を変更することで文の類似性に強弱をつけることができる．本論文で用いる類似性の強弱のクラスについて解説する．文のペアが主語か述語のいずれか一方が同じものを表しているとき，片側一致型の類似性を持つという．また，文のペアが述語の肯定・否定にかかわらず主張点が一致しているとき，主張部一致型の類似性を持つという．文のペアが文の主張点が述語の肯定・否定に関わらず一致しているとき，主張完全一致型の類似性を持つという．

片側一致型の類似性を持つ文のペアの例として「色がいいね」と「サイズがいいわ」,「色が薄いな」と「色が赤い」などが挙げられる．主張部一致型の類似性を持つ文のペアとして「色がいいね」と「色が微妙」,「色がいいね」と「カラー最高か」などが挙げられる.「色がいいね」と「カラー最高か」は主張完全一致の類似性も持つ．類似性は通常連続的であるから，すべての類似性がこの型に正確に分類できるわけではないことに注意されたい．

間接比較の類似尺度では「何がよかったですか？」や「色はどうでしたか？」などを間接文に用いることで片側一致型の類似性を測ることができる．また,「色はいいですか？」などを間接分に用いることで主張部一致型の類似性を測ることができる．

4 検索アルゴリズム

本論文ではライブコメントの類似コメントの検索方法として，BERT の特徴量ベクトルを用いた検索と NSP を用いた検索を用い，これらを比較する．これらはそれぞれ式 (1) の直接比較の類似尺度 s_D，式 (2) の間接比較の類似度 s_I を用いた場合の分類と対応する．

4.1 BERT のベクトル表現を用いた検索

ここでは BERT を用いた直接比較による類似コメント検索のアルゴリズムの詳細を示す．使用する特徴量の計算とランキ

ングの方法について分けて解説する．

4.1.1 BERT によるテキストのベクトル表現の計算

次の手順で BERT から入力テキストのベクトル表現を得る．

（1） 入力テキストを形態素解析する．

（2） 形態素を半角スペースで連結した文字列を SentencePiece [12] に入力し，サブワードの列 $\boldsymbol{w} = (w_1, w_2, \ldots, w_D) \in \Sigma^D$ を得る．ここで w_i $(i = 1, \ldots, D)$ はテキストから得られたサブワードをテキスト先頭から順に添字付けしたものであり，Σ はサブワードの集合である．

（3） 文頭を表す [cls] と文の境界を表す [sep] という特殊トークンを得られたサブワードの列 $\boldsymbol{w}$ に挿入し，$\tilde{\boldsymbol{w}} = ([\text{cls}], w_1, w_2, \ldots, w_D, [\text{sep}]) \in \Sigma^{D+2}$ を構成する．

（4） $\tilde{\boldsymbol{w}}$ のサブワードをそれぞれ ID に変換したベクトル $\boldsymbol{x} = (x_1, x_2, ..., x_{D+2})^{\mathrm{T}} \in \mathbb{R}^{D+2}$ を構成する．ここで x_i は $\tilde{\boldsymbol{w}}$ の成分 $\tilde{w}_i$ を ID に変換した値である $(i = 1, \ldots, D + 2)$.

（5） BERT モデルを用いて，$\boldsymbol{Y} = \text{BERT}(\boldsymbol{x})$ を計算する．ただし，$\boldsymbol{Y} = (\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_{D+2})^{\mathrm{T}} \in \mathbb{R}^{(D+2)\times 768}$ は BERT の最終隠れ層の出力であり，x_i に $\boldsymbol{y}_i$ が対応する $(i = 1, \ldots, D+2)$.

（6） $\boldsymbol{Y}$ から特殊トークン [cls] の変換結果であるベクトル $\boldsymbol{y}_1 \in \mathbb{R}^{768}$ を取り出し，これを入力テキストのベクトル表現とする．

手順 (1),(2) で形態素解析後に SentencePiece をかける理由は，SentencePiece のみでサブワード化すると出現頻度的には多いが通常では分割しないような区切りをもってテキストを分割してしまうことがあり，パフォーマンスが維持しづらいからである．

4.1.2 BERT を用いた検索ランキング

BERT によるベクトル表現を用いた検索では，次の手順で検索ランキングを作成する．

（1） 上記の BERT によるテキストのベクトル表現の計算を用いて検索クエリのベクトル表現 $\boldsymbol{v}_0$ を計算する．

（2） 検索対象となるコメントの中から一つコメントを取り出し，BERT を用いてそのコメントのベクトル表現 $\boldsymbol{v}_1$ を計算する．

（3） $\|\boldsymbol{v}_0 - \boldsymbol{v}_1\|$ を計算し，その値を $\boldsymbol{v}_1$ の元のコメントのスコアとする．ただし $\|\cdot\|$ は L_2 ノルムである．

（4） 手順 (2), (3) をすべてのコメントに対して計算する．

（5） コメントをスコア順に並べ，スコアが閾値 $\theta_D \in (0, \infty)$ 以下のコメントのみ取り出し，これを検索ランキングの結果として出力する．

4.2 NSP による会話的連続性スコアを用いた検索

ここでは NSP を用いた間接比較による類似コメント検索のアルゴリズムの詳細を示す．使用する特徴量の計算とランキングの方法について分けて解説する．

4.2.1 NSP による会話的連続性のスコア計算

次の手順で NSP から 2 つの入力テキストの会話的連続性のスコアを得る．ここでは前文となる入力テキストをテキスト 1, 後文となる入力テキストをテキスト 2 と呼ぶ．

（1） テキスト 1, テキスト 2 をそれぞれ形態素解析し，それぞれの形態素を得る．

（2） テキスト 1 の形態素に関して，形態素を半角スペースで連結した文字列を SentencePiece に入力し，サブワードの列 $\boldsymbol{w} = (w_1, w_2, \ldots, w_{D_1}) \in \Sigma^{D_1}$ を得る．ここで w_i $(i = 1, \ldots, D_1)$ はテキストから得られたサブワードをテキスト先頭から順に添字付けしたものである．同様にしてテキスト 2 のサブワードの列 $\boldsymbol{u} = (u_1, u_2, \ldots, u_{D_2}) \in \Sigma^{D_2}$ を得る．

（3） 文頭を表す [cls] と文の境界を表す [sep] という特殊トークンを得られたサブワードの列 $\boldsymbol{w}, \boldsymbol{u}$ と連結し，$\tilde{\boldsymbol{w}} = ([\text{cls}], w_1, \ldots, w_{D_1}, [\text{sep}], u_1, \ldots, u_{D_2}, [\text{sep}]) \in \Sigma^{D_1+D_2+3}$ を構成する．

（4） $\tilde{\boldsymbol{w}}$ のサブワードをそれぞれ ID に変換したベクトル $\boldsymbol{x} = (x_1, x_2, ..., x_{D_1+D_2+3})^{\mathrm{T}} \in \mathbb{R}^{D_1+D_2+3}$ を構成する．ここで x_i は $\tilde{\boldsymbol{w}}$ の成分 $\tilde{w}_i$ を ID に変換した値である $(i = 1, \ldots, D_1+D_2+3)$.

（5） NSP モデルを用いて，$y = \text{NSP}(\boldsymbol{x})$ を計算する．この y が 2 つの入力テキストの会話的連続性のスコアとなる．

4.2.2 NSP を用いた検索ランキング

NSP による会話的連続性スコアを用いた検索では，次の手順で検索ランキングを作成する．

（1） 検索対象となるコメントの中から一つコメントを取り出し，検索クエリをテキスト 1, 取り出したコメントをテキスト 2 とする．上記の NSP による連続性スコア計算を用いて連続性スコアを計算し，この値をコメントのスコアとする．

（2） 手順 (1) をすべてのコメントに対して計算する．

（3） コメントをスコア順に並べ，スコアが閾値 $\theta_I \in (-\infty, \infty)$ 以上のコメントのみ取り出し，これを検索ランキングの結果として出力する．

5 評価実験

ここでは前節で述べた検索方法を用いて，式 (1) の直接比較の類似尺度 s_D，式 (2) の間接比較の類似度 s_I の性能評価を行う．

5.1 実験内容

5.1.1 主張部一致型類似コメントの検索精度評価

この実験ではそれぞれの類似尺度が与えられたコメントに対して 3 章で解説した意味で主張部一致型の類似性を持つコメントを類似コメントとして判断できるかを確認する．与えられたコメントに対する類似コメントをデータベースから検索を行い，その精度を評価する．

実験内容について説明する．まず，予め人手で意味的分類を行いクラス分けしたライブコメントを用意し，各クラスから代表となるコメントを 1 件ずつ抽出しておく．次にすべてのコメントをすべて同じデータベース上に格納する．このデータベースから先程抽出した代表コメントを 1 件ずつ用いて順に検索を行う．この検索には代表コメントのテキスト情報のみを用いる．この検索結果が，代表コメントがもともと含まれていたクラスの中身と同一であるかを適合率と再現率を用いて評価する．これを各代表コメントに対して行い，適合率と再現率のそれぞれ

でマクロ平均とる．最後にこのマクロ平均から F1 値を計算し，それを検索精度の評価値とする．評価データセットの作成手順に関しては 5.2 節で示す．

この実験では検索アルゴリズムとして BERT, NSP を用いる．それ以外の設定としてファインチューニングの有無，検索クエリの疑問文化の有無，ランキングの閾値値などを変更しながら実験を行う．ファインチューニングの詳細は 5.3 節で示す．

式 (1) の直接比較の類似尺度 s_D，式 (2) の間接比較の類似度 s_I の性能評価に BERT と NSP による検索を用いるが，これらは通常検索クエリが異なることに注意する．BERT は代表コメントをそのまま検索クエリとして入力し，その類似コメントを集める．一方で NSP は代表コメントを疑問文化した文を検索クエリとして入力し，その類似コメントを集める．実験では疑問文化を行わないそのままの文，疑問文化を行って作成した文を BERT, NSP のどちらに対しても入力し，その精度を確認する．検索クエリとなるコメントの疑問文化は機械的にコメントの語尾のみをですます調の疑問形式に変換してを行う．このとき，語尾に付いた絵文字などは消去する．例えば「発表楽しみやわ」を「発表楽しみですか？」に,「日本語字幕に変わらない (° ∀ °)」を「日本語字幕に変わらないですか？」に,「画面デカいなー」を「画面デカいですか？」に変換した．

BERT のランキング出力の閾値は $\theta_D = 7.0, 7.5, 8.0$ を用いた．一方で NSP のランキング出力の閾値は $\theta_I = 0.0, 1.5, 3.0$ を用いた．これらは複数回の実験から経験的に決定した値である．

5.1.2 片側一致型類似コメントの検索精度評価

この実験ではそれぞれの類似尺度が与えられたコメントに対して 3 章で解説した意味で片側一致型の類似性を持つコメントを類似コメントとして判断できるかを確認する．与えられた検索クエリを用いて類似コメントをデータベースから検索を行い，その精度を評価する．

実験内容について説明する．この実験ではライブ配信に対して【わからなかったこと】,【良かったこと】,【悪かったこと】,【してほしいこと】について言及しているという観点で類似したコメントをそれぞれ集めることを考える．それぞれの観点に対してコメントが対応するかどうかを 0/1 のラベルをつけたデータベースから BERT, NSP を用いて検索を行い，そのランキングの上位 k 件を取った場合の nDCG 値を評価値とする．k には 10, 20, 30 を用いる．上位 k 件の nDCG 値は検索対象のラベルが 0/1 である場合，次の計算式で与えられる．

$$\mathrm{nDCG}\,(k) = \left(\sum_{j=1}^{k} \frac{2^{r_j} - 1}{\log_2(j+1)}\right) \Bigg/ \left(\sum_{j=1}^{k} \frac{1}{\log_2(j+1)}\right)$$

ここで r_j は検索結果のランキングの第 j 位となるコメントのラベル値である．

この実験ではファインチューニングの有無に関して設定を変更しながら実験を行う．検索クエリは「何がわからなかったですか？」,「何が良かったですか？」,「何をしてほしいですか？」を用いる．BERT に関してはもとの観点に対応するコメントを作る方法がないため，NSP と同様にこれらの疑問文を検索クエリとして検索結果の比較を行う．

5.1.3 検索時間の評価

この実験では NSP による検索アルゴリズムの検索時間を確認する．NSP による検索アルゴリズムによってライブコメントをリアルタイムで検索，分類することができるかを評価するが目的である．ライブコメントのデータベースから検索クエリを変えながら 10 回検索を行い，その検索時間の平均値を計測し，単位時間あたりの処理件数を評価する．

5.2 評価用データセットの作成手順

各実験を行うために評価用データセットが必要となる．ここではそれぞれの評価データセットの作成方法について述べる．

まずライブコメントは Twitter にてハッシュタグ#AppleEvent を検索し，Apple 社の 2020 年 10 月 14 日の製品発表時のツイートをを収集した．さらにこの中で画像，URL を含んでいない 15, 385 件のツイートをライブコメントとしてすべての実験で共通で用いる．

1 つ目の実験の主張部一致型類似コメントの検索精度評価には，意味的類似度が高いコメントをクラス分けしたコメントデータセットを作成する．収集したライブコメントの中から人手で主張部一致型の類似性をもつ，すなわち述語の肯定・否定にかからわず主張点が一致していると感じるコメントを一定数集め，分類ラベル付きコメントデータを構成する．結果的に各クラス 5 件，計 14 個のクラスが構成された．この分類作業に携わった人員数は 1 名である．構成されたクラスの例を挙げると【発表が楽しみかどうか】,【発表が始まったかどうか】,【日本語字幕に変更できるかどうか】,【日本語字幕が嬉しいかどうか】などである.【発表が楽しみかどうか】のクラスのコメントの例を挙げると「発表たのしみやわ」,「どんな製品が発表されるか楽しみ」,「ワクワク〜」などと表現に幅がある．

2 つ目の実験の片側一致型類似コメントの検索精度評価には，実験で用いる観点【わからなかったこと】,【良かったこと】,【悪かったこと】,【してほしいこと】について言及していない/しているに対応して 0/1 のラベルをつけたデータセットを作成する．このラベルは観点ごとにつける．今回は収集したライブコメントの中から無作為に 300 件抽出し，そこに人手で 0/1 ラベルを振っていった．このラベル付け作業に携わった人員数は 1 名である．

3 つ目の実験の検索時間の評価には収集した 15,385 件のライブコメントすべてをデータベースとして扱い，この中から検索をかける．

5.3 プレトレーニング, ファインチューニングの手順

BERT のプレトレーニングは Wikipedia, ブログ，新聞記事，QA サイト，企業系 Web ページなどの 12.7GB 分のテキストデータを用いて行っている．ファインチューニングは Twitter のツイートとその返信のペアを用いて行った．まず，疑問符が語尾についた tweet とその tweet への返信を収集し，この組を正例ペアデータとして 10 万件作成した．次にこのツイートと返信のペアをランダムに変更した負例ペアデータを 10 万件作

表 1　主張部一致型類似コメントの検索精度

検索方法	閾値 θ	finetuning	疑問文化	適合率	再現率	F1 値
BERT	7.0	No	No	**0.964**	0.243	0.388
BERT	7.0	No	Yes	0.071	0.029	0.041
BERT	7.0	Yes	No	0.740	0.614	0.671
BERT	7.0	Yes	Yes	0.884	0.457	0.603
BERT	7.5	No	No	**0.964**	0.271	0.424
BERT	7.5	No	Yes	0.071	0.029	0.041
BERT	7.5	Yes	No	0.644	0.686	0.664
BERT	7.5	Yes	Yes	0.835	0.571	0.679
BERT	8.0	No	No	0.893	0.286	0.433
BERT	8.0	No	Yes	0.071	0.029	0.041
BERT	8.0	Yes	No	0.482	0.743	0.585
BERT	8.0	Yes	Yes	0.678	0.714	0.696
NSP	0.0	No	No	0.549	0.786	0.646
NSP	0.0	No	Yes	0.714	0.271	0.393
NSP	0.0	Yes	No	0.209	**1.000**	0.346
NSP	0.0	Yes	Yes	0.261	**1.000**	0.414
NSP	1.5	No	No	0.770	0.514	0.617
NSP	1.5	No	Yes	0.357	0.157	0.218
NSP	1.5	Yes	No	0.692	0.886	0.777
NSP	1.5	Yes	Yes	0.738	0.871	**0.799**
NSP	3.0	No	No	0.321	0.129	0.184
NSP	3.0	No	Yes	0.071	0.014	0.024
NSP	3.0	Yes	No	0.850	0.600	0.703
NSP	3.0	Yes	Yes	0.843	0.657	0.739

表 2　主張部一致型類似コメントの検索精度@5

検索方法	finetuning	疑問文化	適合率	再現率	F1 値
BERT	No	No	0.554	0.400	0.464
BERT	No	Yes	0.173	0.129	0.147
BERT	Yes	No	0.600	0.600	0.600
BERT	Yes	Yes	0.671	0.671	0.671
NSP	No	No	0.750	0.700	0.724
NSP	No	Yes	0.786	0.257	0.387
NSP	Yes	No	0.757	0.757	0.757
NSP	Yes	Yes	**0.800**	**0.800**	**0.800**

成した．最終的にこれらを合わせた 20 万件のペアデータを学習データとし，BERT, NSP のファインチューニングを行った．

5.4 実験結果

5.4.1 主張部一致型類似コメントの検索精度評価

表 1 に主張部一致型類似コメントの検索に関する実験結果の評価値を示す．表中の太文字はその列で最も高い評価値を示している．表 1 からまずわかる情報として，最も F1 値が高いモデルは NSP を用いた検索アルゴリズム ($\theta = 1.5$, ファインチューニングあり, 疑問文化あり) のモデルである．しかしながら全体的には BERT,NSP に関して精度の優劣をつけ難い．これは閾値 θ の値に対して結果が大きく変動するからである．BERT のモデルに関しては閾値 θ が $7.0, 7.5, 8.0$ と動くに連れて適合率が下がり，再現率が上がっていることがわかる．そして BERT のモデルでは $\theta = 7.5$ にて全体的に F1 値が高くなっている傾向が見て取れる．NSP モデルに関しては閾値 θ が $3.0, 1.5, 0.0$ と動くに連れて適合率が下がり，再現率が上がっていることがわかる．BERT と NSP で θ を動かす順序が逆なのはそれぞれ閾値を見ている特徴量の性質が逆であるからである．そして NSP のモデルでは $\theta = 1.5$ にて全体的に F1 値が高くなっている傾向が見て取れる．これにより，F1 値がおおよそ最も高くなるような θ の評価値を観測できていることがわかる．

ここで閾値 θ の設定値に依存せずに評価を行うために，ランキングの上位のみを考えたときの評価値を考える．特に今回の分類は各クラスが 5 件で構成されていることから上位 5 件で評価を行う．上位 5 件のみで評価を行った場合の結果を表 2 に示す．を表 2 でも表 1 と同様に，最も F1 値が高いモデルは NSP を用いた検索アルゴリズム (ファインチューニングあり, 疑問文化あり) のモデルである．ここから，主張部一致型の類似性を持つコメントを NSP モデルに対応する間接比較の類似尺度はうまく集められることが確認できた．この実験で使用した評価データはそのデータ件数が少ないため，一概に直接比較より間接比較の類似尺度のほうが良いとは断言できない．しかしながら主張部一致型の類似性を持つコメントを集める場合には間接比較の類似尺度が直接比較と同様か，もしくはそれ以上に類似コメントをうまく集めることができる可能性が高い．

ここからファインチューニング，検索クエリの疑問文化に関する影響を考える．まずファインチューニングの有無に関しては表 1, 2 からわかるように BERT, NSP のどちらに対しても F1 値を向上させていることがわかる．これはプレトレーニングで使用したデータと今回の対象となるライブコメントの性質が大きく異なるためである．次に検索クエリの疑問文化の有無に関して考える．ファインチューニング前は BERT, NSP のどちらでも疑問文化を行わないで検索を行ったほうが F1 値が高くなる傾向があったが，ファインチューニング後は疑問文化を行ったほうが F1 値が高い．しかしながらファインチューニング後は疑問文化をしなくてもある程度 F1 値が高くなっていることがわかる．これはまずプレトレーニングで使用したデータが Wikipedia やニュースサイトといったテキストであり，ここに疑問文があまり含まれていなかったため，ファインチューニング前は疑問文化した検索クエリに適切に対応できていなかったと考えられる．一方でファインチューニング後は疑問文にも汎化した結果，検索クエリが疑問文の場合に正しい検索ができるようになり，疑問文でない場合でもある程度の性能を出せる状態となったと考えられる．

5.4.2 片側一致型類似コメントの検索精度評価

表 3 に片側一致型類似コメントの検索に関する実験結果の評価値を示す．表中の太文字はその列で最も高い評価値を示している．nDCG の値は高いほうがスコアとして良い．表 3 から各 k に対して nDCG の値が最も高いのは NSP のファインチューニングありのモデルとなっていることがわかる．ここから NSP モデルに対応する間接比較の類似尺度の有効性が確認できる．本実験は BERT モデルに対応する直接比較の類似尺度にとって明らかに不利になるようなタスク設定，すなわちコメントから検索を行わないようなタスクであったが，逆に間接比較の類似度がこのような「わからなかったこと」,「良かったこと・悪

表 3　片側一致型類似コメントの検索精度@k

k	検索方法	finetuning	nDCG
10	BERT	No	0.238
	BERT	Yes	0.219
	NSP	No	0.263
	NSP	Yes	**0.552**
20	BERT	No	0.270
	BERT	Yes	0.231
	NSP	No	0.226
	NSP	Yes	0.524
30	BERT	No	0.266
	BERT	Yes	0.218
	NSP	No	0.231
	NSP	Yes	0.443

かったこと」,「してほしいこと」といった片側一致型の類似コメントを集めることに有効であることが実験結果より示された.

5.4.3　検索時間の評価

実験では GPU(GeForce GTX 1080 Ti メモリ 11GB) を 1 枚用いて NSP のランキング計算を行った．このとき，10 回の検索時間の平均は 41.38 秒であった．1 秒あたりの処理件数は約 372 件である．これにより，配信へのライブコメント投稿者が数百名程度であるならば一つの検索をリアルタイムで行うことができることが予想できる．複数の検索を回すためには複数の GPU を用いて並列計算を行ったり，投稿者の人数がさらに少ない状態であれば問題ないと考えられる．つまり NSP による検索アルゴリズムは小規模のライブ配信に関しては十分の計算速度を持つということができる．

5.5　考　　察

ここでは具体的な検索結果を確認することで，直接比較の類似尺度と間接比較の類似尺度を用いた場合でどのような違いが発生するかを考察する．ライブコメントの中から「色がいいね」と似たコメントを探すことを考える．評価データで用いたライブコメント全体をデータベースとし，ここから検索を行う．直接比較は「色がいいね」をそのまま検索クエリとして入力して検索する．間接比較は主張部一致型，片側一致型の 2 つの類似度を用いてそれぞれ検索を行う．主張部一致型は「色がいいね」を疑問文化した「色はいいと思いますか？」を検索クエリとして入力して検索する．片側一致型は「色がいいね」を疑問文化した「何がよかったですか？」を検索クエリとして入力して検索する．

直接比較, 間接比較の主張一致型，間接比較の片側一致型の検索結果をそれぞれ表 4, 5, 6 に示す．表 4 の直接比較を用いた場合では検索結果の第 5 位に「色がな〜」といった不満を暗示したコメントが含まれており，第 8 位に「形がいいね」といった色に関係ないコメントが含まれている．このように直接比較の場合は元コメントと似ているコメントが網羅的に回収されてしまい，一貫性がなく整理には不向きである．加えて元コメントの文章長と類似した短めのコメントしかとれていないことがわかる．ここから，直接比較は文のスタイルが似たコメントが類似コメントとして集められてしまうことがわかる．

表 4　ライブコメントの中から「色がいいね」と似たコメントを直接比較を用いて「色がいいね」で検索したランキング結果の上位 10 件．文のスタイルが似たコメントが網羅的に集められており，一貫性がない．

順位	コメント
1	色めっちゃいいな
2	グリーンがかわいいな
3	ブルーがかっこいい
4	グリーンカラーがいいなぁ
5	色がな〜
6	色が良すぎる
7	ブルーいいなぁ
8	形がいいね
9	カラフルだな
10	カラフルだな

一方で表 5 の間接比較で主張部一致型の類似コメントを集めた場合ではすべて色に関するコメントを集められている．さらに「色は先月見た」や「色とかなんでもいい」といった好印象以外のコメントが取れている．これは主張部一致型の類似コメントであり,【色についていいと思うかどうか】という観点で一貫して集められていることがわかる．また直接比較と比べて多様なスタイルのコメントがとれている．表 6 の間接比較で片側一致型の類似コメントを集めた場合では主語が欠落しているコメントがあるが，すべて何らかのよかったものを述べるコメントが集められており，こちらにも一貫性が見られる．主語が欠落した真に適切でないコメントが得られている理由は，今回の NSP の学習データを Twitter のデータを用いて簡易的に作成したためであると考えられ，人間が作成した短めの Q&A などを用いて学習を行うことでこの問題を解決できる見込みがある．

全体を通して，間接比較の類似尺度はその間接文を変更することで何の観点で類似するかを制御ができ，またその類似性の解釈が可能なことがわかる.「色がいいね」と主張完全一致型の類似コメントを探す場合には,「色はいいと思いますか？」と「何が良かったですか？」の検索結果の共通集合をとるなど，疑問文を複数組み合わせて使用すればよいと考えられる．

6　今後の展望

本論文では間接比較による類似性が様々な類似性を定義できることを示した．今回の実験では一つの間接文のみを用いてコメント間の類似性を測ったが，今後は複数の類似性を組み合わせて主張完全一致型類似性を定義できるかを検討する必要がある．また，今回は間接文を作成する際に逐一手動で疑問文を生成していたが，これを自動化することも検討内容として考えられる．疑問文の自動推定が行えれば様々な類似性を自動で作成でき，分類の自動化に有効であると考えられる．最後に，今回は間接比較の類似性を測るために NSP モデルを用いたが，この精度向上も必要である．今回は学習データを簡単な方法で作成したため，学習データを改良したり，学習方法を変更するこ

表 5　ライブコメントの中から「色がいいね」と似たコメントを間接比較を用いて「色はいいと思いますか？」で検索したランキング結果の上位 10 件．色に関する印象の類似コメントのみが集められており，文のスタイルは多様である．

順位	コメント
1	ID なら最高なんだけど叶わぬ夢っぽい。 青色が良さそうなら多分買います。
2	色めっちゃいいな
3	青色いいなぁ
4	いい色だなぁ。うっとり
5	黒色もありますと
6	Twitter の背景みたいな青色いいね！
7	やっぱ、赤とネイビー入るのか。いい色。
8	え、普通にかっこよくない？ 色は先月見た気がしないでもないけど
9	どうせ黒か白しか買わんから色とかなんでもいいよ。
10	個人的には緑がいいなー！

表 6　ライブコメントの中から「色がいいね」と似たコメントを間接比較を用いて「何がよかったですか？」で検索したランキング結果の上位 10 件．主語が欠落しているコメントがあるが，全て何らかのよかったものを述べた類似コメントが集まっている．

順位	コメント
1	c-c がよかったな
2	よかった
3	よかったよかったよかったよ
4	今回の良かった
5	11 シリーズ発表の時と比にならないくらい良かった
6	今回は iPhone mini が 1 番印象的でした
7	満足な内容でした
8	今回のイベントはよかったです !! 久しぶりにワクワクしました。
9	Qi 互換か、よかった
10	LiDAR センサのおかげで研究費で買う理由が しっかりできました

とが考えられる．

7 ま と め

本研究では動画ライブ配信のチャット機能や SNS への実況投稿などに焦点を当て，投稿されるコメント文から配信者へフィードバック可能な情報にリアルタイムで集約・整理することを目的とした．特に今回はコメントの分類を考え，コメント間の類似性について議論した．これまでの類似度計算では 2 つの文を直接比較して文の意味の類似性を測るが，この場合どのような観点で文が類似しているか不明瞭であり，分類のためにその類似性を意図的に制御することも難しかった．そこで本研究では間接比較によって類似度を測ることを提案し，具体的な計算機構として NSP モデルを挙げた．間接比較では間接文の介在により，類似度の解釈性を上げることができる．本論文では直接比較と間接比較の類似度を比較するために，類似コメントの検索実験を行い，その性能を評価した．実験により間接比較の類似度が間接文を変更することで，主張部が同じ文の類似度（主張部一致型の類似性）や，部分的な類似性（片側一致型の類似性）を持つ文の類似度など，様々な類似尺度を計算可能であることが明らかとなった．また，実際のライブコメントの分類に使用できるかを確認するため，計算時間の計測実験を行い，現実的な時間で計算ができることを示した．

文　献

[1] Jiawei Han, Jian Pei, and Micheline Kamber. *Data mining: concepts and techniques*. Elsevier, 2011.

[2] Rajdeep Mukherjee, Hari Chandana Peruri, Uppada Vishnu, Pawan Goyal, Sourangshu Bhattacharya, and Niloy Ganguly. Read what you need: Controllable aspect-based opinion summarization of tourist reviews. *arXiv preprint arXiv:2006.04660*, 2020.

[3] Ramya Tekumalla and Juan M Banda. Characterizing drug mentions in COVID-19 Twitter chatter. In *Proceedings of the 1st Workshop on NLP for COVID-19 (Part 2) at EMNLP 2020*, Online, December 2020. Association for Computational Linguistics.

[4] Zhengjie Miao, Yuliang Li, Xiaolan Wang, and Wang-Chiew Tan. Snippext: Semi-supervised opinion mining with augmented data. In *Proceedings of The Web Conference 2020*, pp. 617–628, 2020.

[5] Christopher D Manning, Hinrich Schütze, and Prabhakar Raghavan. *Introduction to information retrieval*. Cambridge university press, 2008.

[6] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[7] 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.

[8] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543, 2014.

[9] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, Vol. 5, pp. 135–146, 2017.

[10] Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*, 2018.

[11] Changchun Li, Jihong Ouyang, and Ximing Li. Classifying extremely short texts by exploiting semantic centroids in word mover's distance space. In *The World Wide Web Conference*, pp. 939–949, 2019.

[12] Taku Kudo and John Richardson. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pp. 66–71, Brussels, Belgium, November 2018. Association for Computational Linguistics.

[13] Guoxiu He, Zhe Gao, Zhuoren Jiang, Yangyang Kang, Changlong Sun, Xiaozhong Liu, and Wei Lu. Read beyond the lines: Understanding the implied textual meaning via a skim and intensive reading model. *arXiv preprint arXiv:2001.00572*, 2020.