CE-001

カウンセリング内容の機械学習を用いた分類
Tag classification for counseling content using machine learning

山口 匠[1)]　中道 万優子[1)]　馬場 隆寛[1)]
Takumi Yamaguchi　Mayuko Nakamichi　Takahiro Baba

1　はじめに

メンタルヘルスとは体の健康ではなく、メンタルの健康状態を意味する。体が軽いとか、力が沸いてくるといった感覚と同じように、心が軽い、穏やかな気持ち、やる気が沸いてくるような気持ちの時は、メンタルが健康といえるだろう。しかし、だれでも気持ちが沈んだり、落ち込んだりすることはある。日々の生活の中でストレスを感じることも少なくない。気分が落ち込んだり、ストレスを感じることは自然なことだが、このような気分やストレスが続いてしまうと、メンタルの調子をくずしてしまう原因にもなる。さらにメンタルの不調は、周囲の人に気づかれにくく、自分からも伝えづらいため、回復に時間がかかってしまうこともある。

近年、メンタルの病気は増えていて、生涯を通じて 5 人に 1 人がメンタルの病気にかかるともいわれている。メンタルの病気は特別な人がかかるものではなく、ストレスなどが積み重なることがきっかけとなって、かかってしまうことがあるように、誰でもかかる可能性がある。眠れない、気分が沈む状態が続いてしまうなど、人によって症状はさまざまである [1]。

メンタルの病気で苦しんでいる人達を支援するためにインターネットでも様々な活動が行われている。今回はその支援の 1 つであるココオルについて詳しく説明する。ココオルでは、1 人きりで悩み苦しむ人をなくし、自殺撲滅、自殺ゼロ社会の実現を目指している。活動の 1 つとして、様々な悩みを持つユーザがコメントすることで、相談スタッフが返信をする。そして、自立支援、社会復帰のサポートを通じて、世界中の全ての人たちが、愛に満ちた幸せな毎日を過ごせる社会にするために活動している [2]。

ココオルでは、ユーザの悩み相談に対してスタッフが手動で返事をするため、返事を書くまでに数日以上要することも少なくない。そこで、コメント時にどのような内容なのかを分かりやすくするために、コメントに合ったタグが付属されている。相談スタッフは返事を書く時にこのタグを活用していると報告があり、スタッフの返信作業の効率化に貢献している。しかしながら、タグ付け自体は文章の自動判別で行われているため、ユーザの悩みの書き方によっては適切なタグ付けが行われず、相談内容に即していないタグも散見される。そのため、期待しているタグの活用による作業の効率化が十分行えていない。これでは、返事を待っているユーザからも返事の催促を促され、悩み続け苦しんでいる方々の十分なサポートが不十分な結果となってしまう。そこで、我々は機械学習を用いて内容に適したタグを予測することで、ユーザの相談内容をより正確に反映したタグ付けを行った。タグの選定は、ココオルのタグの中から「つらい」「死にたい」「限界」等の 21 種類に絞った。それらからコメントの内容に適するタグを 1 つのコメントに最大 5 つまで付与するアノテーションを行った。今回はそのコメントとタグを用いてマルチラベル分類の実装を行う。

1）　久留米工業大学 情報ネットワーク工学科

2　関連研究

本研究ではテキストとタグから BERT を用いてマルチラベル分類を行うため、タグ分類、BERT の手法、マルチラベル分類を用いた関連研究、そして、テキストデータのトリミング関連を紹介する。

岸田ら [3] はゲームソフトの評価レビューを「戦闘」「物語」「サウンド」をそれぞれ「肯定」と「否定」のデータから SVM と BERT を使ってマルチラベル分類を行っている。

竹中ら [4] は Twitter のハッシュタグをベイジアンフィルターを用いて、各ハッシュタグごとに 2 値分類を行い、複数のハッシュタグの推定を行った。

藤井ら [5] は契約書のリスク判定のために条文テキストの審査項目を予測対象としたニューラルネットワークを用いたマルチラベル分類を行った。

按田ら [6] は製品レビューをテキストデータとして、言及部品を示すラベルと評価属性を示すラベルの 2 種類のラベルを用いたマルチラベル分類の実装を行っている。

安田ら [7] はニュース記事の内容を表す複数のラベルをは入力文としてのトークンとラベル構造との関係に着目したマルチラベル分類手法を行った。

深澤ら [8] はレシピタイトルを入力として適切な材料を予測するタスクに取り組み、またそれを解くモデルとして予め学習した材料クラスのベクトル表現を組み込んだマルチラベル分類モデルを提案した。

新見ら [9] は誰でもかかりうる病気であるうつ病を，未然に防ぐことを目的とし，感情を自動で可視化する機能を搭載した認知行動療法を提案するために、辞書ベースでアノテーションの付与を行ったデータを用いて，BiLSTM で 6 感情のマルチラベル分類を行った。

安道ら [10] は効率的な電子カルテの処理に向けて，記載内容を意味単位で捉えるために文をさらに分割するセグメントを定義し，その単位での意味的解析に試行的に取り組んだ．まず，医療文書で用いられる意味的な最小単位に合致するセグメントを定義し，ダミーカルテを対象にセグメント境界をアノテーションした．その上で，各セグメントに対して内容に即した 10 種類の意味ラベルを付与した．

次に長い文章に対しての手法の参考文献を紹介する。純粋な BERT モデルを用いた場合、処理できる文章の単語数の上限は 512 となっている。しかし、本研究のデータにもそれを超える長さの文章がある。そこで文字数調整では 3 つの手法を紹介する。1 つ目は Sun ら [11] の Head-Tail である。先頭のみ、最後尾のみ、先頭と最後尾の方からの合計 3 つで文字数の調整を行った。それぞれを何単語ずつ取れば良いのかはデータごとで異なる。2 つ目は Random である。任意の場所から連続して単語を取得する。Epoch ごとに取得する場所を変更すれば Augmentation のような効果が期待できる。ただし、Head-Tail の手法ほど精度が上がる印象はない [12]。3 つ目は Alberti ら [13] の Sliding Window という手法である。Sliding Window は長いテキストを複数の重複しない短い

テキストにそれぞれ分割する。重複しないようにスライドした複数のテキストとなるが、BERT モデルの 512 で区切る場合は、文脈の一部が切り落とさせるため注意が必要である。その後、それぞれを BERT に適用し結果を統合する方法となっている。また、小林ら [14] は契約書のような長い文書を扱うために Sliding window による小さな単位のテキストへの分割した結果、評価データにおいては 1.8 ポイントのわずかな性能の低下で 44 %の事例が削減できることを示した．他にも、Beltagy ら [15] は、複数サイズのウィンドウを使用することで、長い文でも Attention が注目する位置を制限し、計算量を減らすことを可能にした。次に Kitaev ら [16] は Transformer の効率を向上させるために、Attention の計算量をシーケンス長 L に対して $O(L^2)$ から $O(L \log L)$ に削減した。Zaheer ら [17] は、Attention 機構の手法を変えることによって計算量を減らし、長いテキストに対応した。Pappagari ら [18] は max pooling や mean pooling を用いることで、一定以下の長さに集約し、上限をこえる長さのテキストに対応した。

本研究では、ココオルから提供されたカウンセリング内容であるテキストデータを、その内容に適する複数のタグでマルチラベル分類を行う。また、長い文章には Head-Tail という手法を用いて機械学習を行う。

3　データセット

今回使用したデータセットは、「悩み掲示板」での「心の悩み」に書き込まれた相談内容である「コメント」とそのコメントに適したものを人手でアノテーションして付与した「タグ」の 2 つをデータとした 2064 件で実験を行う。

3.1　使用したデータについて

ココオルは様々なサービスを提供しているが、本研究では、「悩み掲示板」での「心の悩み」に書き込まれた情報を収集しデータとして活用する。

はじめに、ココオルの悩み掲示板について詳細に説明する。目的としてはユーザが悩みを書き込むことで、ココオルスタッフやカウンセラが親身に相談を受け、ユーザの悩みを解消することとなっている。悩み相談は 2014 年から現在まで 25000 件以上の悩みを閲覧することができる。悩みには様々なカテゴリを確認することが可能であり、1 番多い悩みは「心の悩み」の 20000 件弱である。今回はこの「心の悩み」のデータを 19000 件収集した。他にも質問というカテゴリで掲示板に書き込める。

次に、1 つの相談にはどのような事が確認出来るかを説明する。ニックネームや日時、実際の相談内容に加えて、投稿に対して気になる場合は「気になる相談に登録」に登録することで、後日返答内容をすぐに確認することが可能となる。また、SNS アプリにある機能と似たように「共感」や「応援」の項目がある。他には、「参考：似ている悩みと対処法」である自分の悩みと似ている相談をピックアップして表示される機能がある。この機能があることで、自分以外にも同じ境遇や悩みを持った人がいることを確認し、悩み解消のための参考やココオルの目的である１人きりで悩み苦しむ人をなくすことの機能の１つであると考えられる。

最後にタグがあるが、このタグ付けは内容に含まれている単語を抜き出している仕様となっている。つまり様々な内容や事細かく記述した相談の場合は、タグの数が多く相談内容の本質をタグで確認することが困難な状態となっている。これではタグの存在意義が問われることとなる。図 1 並びに図 2 は本研究の目指すことのイメージとなっている。はじめに、この図 1 の相談内容は、相談者が今までの人生から自身を失ってしまうという内容である。次にタグの中身を確認すると「社会不安障害」「不安障害」と大まかに「不安感」と括ることが可能だと考えた。また「小学生」となっているが、内容からは相談者は少なくとも中学生以上ということが分かるため「小学生」というタグは今回は使わないほうが良いと思われる。加えて「いじめ」も小学生の時の内容で用いられているため、相談者の現在の状況では無いため必須ではないと考えた。これらから図 1「オリジナル」では「小学生」というタグがミスリードとなってしまい、タグからでは相談者が相談スタッフに 1 番聞きたい内容が分からない。本研究では図 2「改善後」のタグを目指すこととなる。過去の出来事を「トラウマ」「つらい」と表し、現状で「死にたい」「不安感」「コンプレックス」と 5 つのタグで相談者の今日までの内容をタグで表す事が出来た。まだまだ完全にタグだけでカバーは出来てはいないが、本研究では図 2 のように「タグ」から相談者が相談スタッフに聞きたい内容を「タグ」として取り扱うことを目標とする。また、相談スタッフには各々によって得意な分野が違うため、多くの質問内容に適したそれぞれの相談スタッフが早急に対応出来るようにすることが本研究の目的である。

図 1: オリジナルのタグ

図 2: 改善後のタグ

3.2 タグ設定

前述した通り、相談内容に記述されたキーワードがそのまま「タグ」として振り分けられる。タグの種類は全部で 1215 個確認出来た。様々な内容の悩みがあるためタグが相当数とはなるが、実際に中身を確認してみると、半分以上はその言葉のままで流用しているが、「暴言」や「虐待」、「パワハラ」をすでにあるが「暴力」としてまとめ、「過食症」「過食嘔吐」「摂食障害」を「過食」と抽象的にまとめることが可能なタグ等が複数あることが確認出来た。現段階でのタグのメリットとしては、相談内容を確認せずともタグだけで相談者の現状を把握することが可能である。しかし、デメリットとしては、タグが多いため、似たような相談内容を利用者側は 1 ページずつ確認するか、「参考：似ている悩みと対処法」にある数個から探す必要がある。他にも、大方同じ内容でも相談者の言葉選びによってタグの振り分けが異なり、「参考：似ている悩みと対処法」のリストから外れてしまうこともある。これらの事情を踏まえ、これからは内容を簡単に示すためのタグではなく、利用者や返答する側にとって活用しやすいタグを目指す必要があることを提案する。

今回の研究で使用したタグは「いじめ」,「うつ病」,「つらい」,「イライラ」,「コンプレックス」,「トラウマ」,「不安感」,「依存」,「嫌がらせ」,「孤立」,「寂しい」,「後悔」,「悪口」,「暴力」,「死にたい」,「無気力」,「無視」,「眠れない」,「罪悪感」,「過食」,「限界」の計 21 種類のタグとなる。これらは 1200 以上あるタグの内の使用頻度が上位 20 %に分類されるタグに厳選した。

3.3 タグの頻出度

収集したデータには「名前」「時間」「相談内容」「タグ」の 4 つの項目がある。今回使用した項目は、「相談内容」とその内容に適する 21 種類のタグに人手で 1 ～ 5 つまでのタグ付けをした「タグ」の 2 つの項目を使用する。実際に使用するデータは、全体のデータを時系列に収集したデータ 19000 件の内ランダムに置き換えた後に、時間の許す限りにタグ付けのアノテーションした 2064 件で学習することとする。

3.4 タグの偏り

表 1 のタグ頻度から懸念点を記述する。「つらい」はデータの中でおよそ 50 %に近い頻度で出現している。逆に「いじめ」「寂しい」「悪口」「嫌がらせ」「無視」の 5 つは出現頻度は 100 以下と割合では 5 %程となっている。出現回数が上位のタグは抽象的な言葉が多いため自ずと回数が多くなることが分かった。最大の懸念点としては「つらい」「不安感」の 2 つのタグが他のタグを予測するときに大きな足掛かりとなる可能性があるのではないかと考えている。

表 1: タグ頻度

タグ名	出現回数
つらい	960
不安感	736
限界	495
死にたい	425
コンプレックス	411
孤立	356
暴力	271
無気力	215
依存	214
後悔	207
イライラ	178
罪悪感	168
トラウマ	158
うつ病	156
眠れない	112
過食	110
いじめ	97
寂しい	96
悪口	89
嫌がらせ	27
無視	21

4 手法

4.1 前処理

3 章で述べたように今回は全部で 2064 件のデータを使う。データには「コメント」と人手でアノテーションをした「タグ」の 2 つの項目がある。

4.1.1 テキストクリーニング

「コメント」ではテキストクリーニングとして改行、全角と半角記号、全角と半角スペースのそれぞれを削除した。次に「タグ」では「つらい」や「死にたい」等の文字列となっているため、数値のベクトル形式に変換するためにワンホットエンコーディングを使用する。それぞれのタグを各カテゴリとすると、各カテゴリに対して 1 つの要素が 1 で他の要素が 0 となるように表現される。このような変換によって、カテゴリカルな変数を数値のベクトルとして扱うことができる。

4.1.2 データのトリミング

データのトリミングとは、最初の方と最後の方の部分を抽出する。具体的な方法は、データを適当な位置で切り取るか、必要な文字数を指定して抽出するという方法がある。本研究のデータである相談内容の文字数は、図 3 の「文字数と頻度表」から最大のもので 6000 文字を超えるデータがある。平均は 604 文字、中央値は 449 文字となった。全ての相談内容の中身を確認したところ、始めや終わりに相談者がカウンセラーに問いたい内容が含まれ、真ん中部分は相談者の過去の経緯が語られていることが多くのデータで分かった。よって、最初と最後の内容をデータとして使うためにトリミングを行った。このトリミングの実行の有無の実験データを 5 章で発表する。

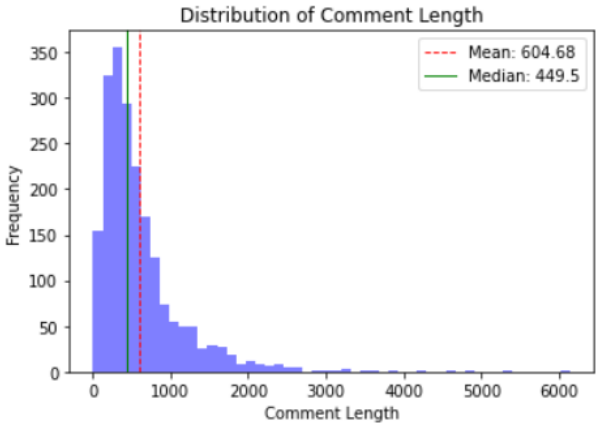

図 3: 文字数の頻度表

4.1.3 ホールドアウト法

はじめにホールドアウト法とは、データを学習用データとテスト用データに分割して評価する方法である。今回は scikit-learn の train_test_split() を用いて、学習用に 9 割、テスト用に 1 割にして交差検証を行った。データ数が少なく、それに加えて表 1 の「タグ頻度」の「嫌がらせ」「無視」のタグ数が非常に少ない事から極端な分割となった。

4.1.4 アノテーション

アノテーションの基準について詳しく説明する。今回はアノテータが「コメント」の内容を確認し、21 種類のタグを付与することとした。付与するタグは 1 個から最大 5 個までとし、もし 6 個以上の場合は相談内容の質問となる部分や複数回出現した言葉や内容に適したタグを優先して付与する。

4.2 BERT

本研究で用いた BERT について説明する。

トークナイザは HuggingFace のライブラリを使用し，東北大学による訓練済み日本語 BERT モデル[1)]で実装を行った。

4.3 評価指標

一般に精度の評価には accuracy(正解率), precision(適合率), recall(再現率), F-measure(F 値) を使用し、以下のように定義される。Accuracy は正と予測したもののうち、実際に正であったもの、または負と予測したもののうち、実際に負であるものの割合である。次に、Precision は正と予測したデータのうち，実際に正であるものの割合、Recall は実際に正であるもののうち，正であると予測されたものの割合である。最後に F-measure は Precision と Recall のトレードオフの関係となっており、それら 2 つの調和平均である。今回は多クラス分類モデルの評価では、まず、各クラスにおける True Positive(TP)、True Negative(TN)、False Positive(FP)、False Negative(FN) を算出する。表 2 では実際の混合行列の表となっている。続いて、評価指標（適合率や再現率）をマイクロ平均またはマクロ平均法を採用して計算する。

表 2: 混合行列

		Predicted values	
		Positive	Negative
True values	Positive	TP	FN
	Negative	FP	TN

次に多クラス分類の評価指標を例を交えて説明する。3 クラスの分類問題の場合、まず、クラス 1 を Positive とみなし、それ以外のクラスを Negative な事象とみなし、混合行列における TP・TN・FP・FN をそれぞれ計算する。続いて、クラス 2、クラス 3 の順で Positive とみなし、同様な計算を行う。これらの計算結果により、クラス 1・クラス 2・クラス 3 それぞれの評価指標が算出できるようになる。このように、多クラス分類の評価指標は二値分類と同じとなっている。ただし、多クラス分類の場合は評価指標がクラスの数だけ算出される。ゆえに、多クラス分類の場合は、各クラスから得られた評価指標の集計作業が必要であり、これが 2 値分類問題と異なる点と言える。多クラス分類により得られた各クラスの評価指標の集計方法として「マクロ平均法」と「マイクロ平均法」が代表的である。

1) https://www.nlp.ecei.tohoku.ac.jp/news-release/3284/

4.3.1 マクロ平均

マクロ平均法、マイクロ平均の順に説明する。まずマクロ平均法により算出した評価指標とは、各クラスの評価指標の平均となる。そのため、マクロ平均法を採用すると、各クラスのサンプル数の偏りに影響を受けることなく評価指標が算出できるという特徴がある。

例えば、クラス 1 は 10,000 サンプル、クラス 2 は 1,000 サンプル、クラス 3 は 1,000 サンプルあったとする。また、クラス 1 は適合率 0.5、クラス 2 は適合率 0.8、クラス 3 は適合率 0.8 であった場合、マクロ平均法を適合率の集計方法として採用すると、多クラス分類問題全体における適合率は (0.5+0.8+0.8)/3 = 0.7 となる。

このように、マクロ平均は全てのクラスに平等に重み付けするため、最も出現するクラスラベルに対して過度な影響を受けない。一方で、各クラスのサンプル数に応じて重み付けしたい場合は、後述するマイクロ平均法が有効となる。以下がマクロ平均適合率 (Macro-average Precision)(1) とマクロ平均再現率 (Macro-average Recall)(2) の簡単な式となる。

$$\text{Macro-average Precision} = \frac{\sum_i^n \left(\frac{TP_i}{TP_i + FP_i} \right)}{i} = \frac{\sum_i^n \text{Precision}_i}{i} \tag{1}$$

$$\text{Macro-average Recall} = \frac{\sum_i^n \left(\frac{TP_i}{TP_i + FN_i} \right)}{i} = \frac{\sum_i^n \text{Recall}_i}{i} \tag{2}$$

4.3.2 マイクロ平均

次にマイクロ平均法は、データセットのサンプル数全体を考慮して評価指標を算出する方法であるため、各クラスのサンプル数に偏りがある場合、サンプル数の大きいクラスの評価指標が支配的になる。以下がマイクロ平均適合率 (Micro-average Precision)(3) とマイクロ平均再現率 (Micro-average Recall)(4) の簡単な式となる。

$$\text{Micro-average Precision} = \sum_{i=1}^{n} \frac{TP_i}{TP_i + TN_i} \tag{3}$$

$$\text{Micro-average Recall} = \sum_{i=1}^{n} \frac{TP_i}{TP_i + FN_i} \tag{4}$$

そのため、マイクロ平均法は各クラスラベルのサンプル数に応じて重みづけしたい場合に有効な集計手段と言える [19]。

4.3.3 F 値

次にマクロ平均、マイクロ平均の F 値について説明する。macro-F1（マクロ平均 F1）(5) スコアとは、基本的に多クラス分類のタスク（問題）に対する評価指標の一つで、クラスごとに計算した F1 スコアの平均値を指す。また、micro-F1（マイクロ平均 F1）(6) スコアとは、全ク

ラス分をまとめて（＝総計して）計算した F1 スコアの値を指す。いずれも 0.0（＝ 0 %）〜1.0（＝ 100 %）の範囲の値になり、1.0 に近づくほどより良い。また macro-F1 と micro-F1 の使い分け指針を示しておく。不均衡データの場合には、macro-F1 を使う方がよい。もう一方の micro-F1 スコアの計算方法では、特定のクラスの数に大きな偏りがある場合、適切に評価できない可能性が高いためだ。micro-F1 スコアは、多クラス分類における正解率（Accuracy）と同じ計算結果となる。計算方法をあらためて説明すると、マクロ平均（Macro Average）とは、多クラス分類のクラスごとに計算した評価指標（F1 スコアや適合率、再現率など）の平均を取ることである。また、マイクロ平均（Micro Average）とは、多クラス分類の全クラス分をまとめて（＝総計して）計算した評価指標（F1 スコアや適合率、再現率など）の値のことである [20]。

$$\text{Macro-average F1-score} = \frac{1}{n}\sum_{i=1}^{n}(F_1\text{-score})_i \tag{5}$$

$$\text{Micro-average F1-score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{6}$$

$$= \frac{2TP}{2TP + FP + FN}$$

5 評価

5.1 学習と予測結果

本研究ではマルチラベル分類の実装を行い相談内容である「コメント」とその内容に適した「タグ」を機械学習することで、「タグ」を予測することでモデルの精度を評価した。手法としては BERT を用いて実験を行った。

5.1.1 結果 (通常)

表 3 及び表 4 そして、表 5 がマクロ平均とマイクロ平均の評価指標を用いた際の予測結果である。今回は比較対象として、最大トークン数である Max_len をそれぞれ 128,256,512、学習回数である epoch を 5,10,15 に設定し、それぞれを組み合わたものの精度を評価した。

表 3 と表 4 の中で 1 番精度の高い組み合わせは Max_len:512 + epoch:5 の 0.68551 であった。データは 2064 件と決して多いと言えるデータ量ではないため学習回数である epoch は最小の 5 回が表 3 及び表 4 からどの max_len でも精度が良いということが分かった。max_len に着目して精度の比較を考えてみると、「コメント」では 4.1.2 小章にある「データのトリミング」と図 3「文字数の頻度表」で述べたように文字数が 6000 文字を超えるデータも存在し、平均は 604 文字、中央値は 449 文字となっていることからも 128,256,512 の中では平均と中央値に近似する 512 の時がどの評価指標でも概ね精度が高いことが分かった。

F1 値では Max_len:512 + epoch:5 の 0.48780 が F1 値の組み合わせで高い性能を示した。F1 値は適合率と再現率のトレードオフの関係となっているため、1 番精度の高い組み合わせの Macro-average Precision の Max_len:512 + epoch:5 の 0.68551 では Macro-average Recall が 0.34120 のため Macro-average F1-score は 0.20497 という低い性能となった。

5.1.2 結果 (トリミング適用)

表 6 及び表 7、8 では 4.1.2 小章で述べたようにデータの最初と最後の方を抽出するデータのトリミングを加えて精度を評価した。抽出する数は前半部分と後半部分をそれぞれ 100 ずつとする。よって n を用いて、n=100 と記載した。表 6 と表 7 の中で 1 番精度の高い組み合わせは Max_len:256 + epoch:5 の 0.58491 であった。1 つのデータではトリミングで 200 のデータで学習することから、近似した値に近い Max_len:256 及び、データ数からも考慮して epoch:5 の組み合わせが 1 番精度の高いものになることが分かった。

F1 値では Max_len:256 + epoch:15 の 0.44390 が F1 値の組み合わせで高い性能を示した。F1 値でのトリミング無しと比較した場合、Max_len:128 の時全ての組み合わせでトリミング有りの方がスコアが高い結果となった。Max_len:256 では両者の精度はほぼそれぞれのスコアが同等であった。Max_len:512 では Macro の epoch:15 以外はトリミング無しの方がスコアが良いことが分かった。

表 3: averaged Precision

max_len\epoch	5	10	15
128(Micro)	0.36585	0.32683	0.35122
128(Macro)	0.54091	0.40480	0.39942
256(Micro)	0.40976	0.40000	0.35610
256(Macro)	0.55145	0.44552	0.26563
512(Micro)	0.48780	0.45366	0.40976
512(Macro)	0.68551	0.50664	0.30776

表 4: averaged Recall

max_len\epoch	5	10	15
128(Micro)	0.36585	0.32683	0.35122
128(Macro)	0.22062	0.20055	0.31894
256(Micro)	0.40976	0.40000	0.35610
256(Macro)	0.25515	0.27800	0.33217
512(Micro)	0.48780	0.45366	0.40976
512(Macro)	0.34120	0.36450	0.30320

表 5: averaged F1-score

max_len\epoch	5	10	15
128(Micro)	0.36585	0.32683	0.35122
128(Macro)	0.11770	0.08780	0.14695
256(Micro)	0.40976	0.40000	0.35610
256(Macro)	0.14496	0.17328	0.15709
512(Micro)	0.48780	0.45366	0.40976
512(Macro)	0.20497	0.21408	0.17802

表 6: averaged Precision +トリミング (n=100)

max_len\epoch	5	10	15
128(Micro)	0.38537	0.35122	0.39024
128(Macro)	0.54822	0.39967	0.36188
256(Micro)	0.40488	0.33171	0.44390
256(Macro)	0.58491	0.32576	0.34891
512(Micro)	0.40976	0.35610	0.39024
512(Macro)	0.49397	0.55331	0.41476

表 7: averaged Recall +トリミング (n=100)

max_len\epoch	5	10	15
128(Micro)	0.38537	0.35122	0.39024
128(Macro)	0.25537	0.21944	0.31764
256(Micro)	0.40488	0.33171	0.44390
256(Macro)	0.28661	0.17047	0.39110
512(Micro)	0.40976	0.35610	0.39024
512(Macro)	0.24213	0.29394	0.31393

表 8: averaged F1-score +トリミング (n=100)

max_len\epoch	5	10	15
128(Micro)	0.38537	0.35122	0.39024
128(Macro)	0.13264	0.13547	0.19332
256(Micro)	0.40488	0.33171	0.44390
256(Macro)	0.15667	0.10472	0.22988
512(Micro)	0.40976	0.35610	0.39024
512(Macro)	0.12467	0.19817	0.17842

5.1.3 結果 (トリミング内容変更)

5.1.2 小節ではデータのトリミングとして n=100 で精度を調査した。次に Max_len の大きさによって n の値を複数変えて精度を検証することにする。始めに Max_len:128 の時、n=64 で検証し、Max_len:256 の時は、n の値は 128,250 で検証することとした。最後に Max_len:512 では、5.1.1 小節で Max_len:512 + epoch:5 の 0.68551 と精度が 1 番高かったことから、n の値を 128,150,200,250 と多く検証することとした。しかし表 3 から表 8 までの結果から学習回数を増やした epoch:15 の時に他の 2 つと比べて精度の優位性が見られなかったことから、epoch は 5 及び 10 の 2 つで検証を行った。

表 9 の Max_len:128 の時は epoch:5 での Macro-averaged Recall 以外は n=64 の時が精度が高いことが分かった。1 番精度の高い組み合わせは n=64 + epoch:5 の 0.58727 であった。トリミングをしていない表 3 での Max_len:128 epoch:5 での 0.54091 と比べて+0.04636 となった。

表 10 の Max_len:256 では、Micro-averaged Precision に注視した場合、n=250 以外の 2 つは epoch:5 での精度が高い結果となった。Macro-averaged Precision ではどの n の値でも epoch:5 での精度が高く、3 つとも 0.5 以上とはなったが、n=100 の 0.58491 が表 10 の中で 1 番高い精度となった。こちらも、トリミングをしていない表 3 での Max_len:256 epoch:5 での 0.55145 と比べて+0.03274 となった。

表 11 の Max_len:512 では、Macro-averaged Precision に注視した場合、epoch:5 での n=128,n=150,n=250 の 3 つが 0.6 以上と本研究では優れた精度を実現することが出来た。表 11 で 1 番精度の良かった組み合わせは Max_len:512 + epoch:5 + n=128 の 0.62776 となった。しかし、トリミングをしていない Max_len:512 + epoch:5 で 0.68551 だったため、-0.05775 とマイナスとなった。Max_len:512 ではトリミングしない方が結果が有望であることが判明した。

最後に F1 値でのトリミングの有無の総評を行う。まずトリミング有りでの 1 番スコアの高い組み合わせは、Max_len:512 + epoch:5 の 0.48780 となった。しかし、トリミング無しは Max_len:512 + epoch:5 の 0.48780 であるため、トリミングをしない方が F1 値は良いことが分かった。また、Macro ではどの Max_len でも 0.1 代と非常にスコアが低い。

表 9: max_len:128 の時

	epoch	Micro-Precision	Macro-Precision
n=100	5	0.38537	0.54822
n=64	5	0.40488	0.58727
n=100	10	0.35121	0.39967
n=64	10	0.36585	0.48903
		Micro-Recall	Macro-Recall
n=100	5	0.38537	0.25537
n=64	5	0.40488	0.20824
n=100	10	0.35122	0.21944
n=64	10	0.36585	0.27887
		Micro-F1-score	Macro-F1-score
n=100	5	0.38537	0.13264
n=64	5	0.40488	0.14491
n=100	10	0.35122	0.13547
n=64	10	0.36585	0.13717

5.2 まとめ

本研究での評価指標には「Micro-averaged Precision」「Macro-averaged Precision」「Micro-averaged Recall」「Macro-averaged Recall」「Micro-averaged F1-score」「Macro-averaged F1-score」の 6 つで検証した。その中で、Max_len:512 + epoch:5 の 0.68551 が 1 番高い精度となった。

次に精度向上のためにデータトリミングを行ったが、1 番精度の高い組み合わせとして Max_len:512 + epoch:5 + n=128 の 0.62776 となり、結果的にはトリミングしない方が精度が高いこととなった。加えて、F1 値以外での Max_len:512 では 4 つの評価指標全てがトリミングしない方が精度が高いことが明らかになった。

全体的に「Micro-averaged Precision」「Macro-averaged Precision」の方が「Micro-averaged Recall」「Macro-averaged Recall」と比べた時に、精度が良かった原因としてまず第一に、データセット内のクラスの不均衡が影響している可能性を考えた。Macro-averaged Precision は各クラスの適

表 10: max_len:256 の時

	epoch	Micro-Precision	Macro-Precision
n=100	5	0.40488	0.58491
n=128	5	0.42440	0.52582
n=250	5	0.40976	0.55834
n=100	10	0.33171	0.32576
n=128	10	0.41951	0.49312
n=250	10	0.45854	0.37281
		Micro-Recall	Macro-Recall
n=100	5	0.40488	0.28661
n=128	5	0.42440	0.29135
n=250	5	0.40976	0.27028
n=100	10	0.33171	0.17047
n=128	10	0.41951	0.36137
n=250	10	0.45854	0.35623
		Micro-F1-score	Macro-F1-score
n=100	5	0.40488	0.15667
n=128	5	0.42440	0.16076
n=250	5	0.40976	0.14414
n=100	10	0.33171	0.10472
n=128	10	0.41951	0.20836
n=250	10	0.45854	0.18067

合率を均等に重み付けして計算する一方、Micro-averaged Precision は全体の予測陽性数と真陽性数を使って計算するため、サンプル数の多いクラスの予測結果が大きな影響を与えることがある。

また F1 値では、トリミングの有無に関わらず 0.5 以上を超える組み合わせが無かったことは期待していた結果とは異なっていた。実際表 3 の「タグ頻度」からも「つらい」は 960 回使われているのに対して、「嫌がらせ」や「無視」は比率にして 1:40 以上となっている。やはり実験結果からもタグの偏りによる精度への影響は大きいと考えられる。そしてデータ数も訓練データに使用した数は 1857 件だったため、まずはスクレイピングした数の 19000 件の半分をデータとして使えるようにしたい。最終的には 19000 件全てを用いて機械学習を行いたい。

5.3 考察

データ数が 2064 件と少ないため 9:1 のホールドアウト検証法を用いた。モデルの訓練と評価を 1 回のみ行うため、効率的なモデル評価が可能だが、データセットのサイズや分割方法によっては、訓練セットとテストセットの偏りが生じる可能性があるため、極端なタグの偏りがある今回のデータでは、あまり好ましい方法ではない。

今後データ数やタグの偏りを改善した場合は、K 分割交差検証を用いたいと考えている。K 分割交差検証とは、データセットを K 個の均等なサブセットに分割し、そのうちの 1 つをテストセットとし、残りの K-1 個を訓練セットとして使用し、これを K 回繰り返す。そのため、データセット全体が訓練とテストに使用され、データの効率的な利用が可能であり、汎化性能の推定が安定する。また、データの偏りに対して頑健であり、統計的な信頼性が高いことがメリットである。

最後に、実際にココオルでこの自動タグ付けを活用するためには、「タグ」についても議論する必要がある。本研究では 21 種類のタグをココオルでの利用頻度が多いタグから設定したが、年齢や身分などの個人を特定するためのタグも加える必要があると考える。しかし、3.2 節で述べたようにそれぞれのタグを抽象的な言葉に変換

表 11: max_len:512 の時

	epoch	Micro-Precision	Macro-Precision
n=100	5	0.40976	0.49397
n=128	5	0.40976	0.62776
n=150	5	0.47805	0.60132
n=200	5	0.40976	0.55392
n=250	5	0.44878	0.62144
n=100	10	0.35610	0.55331
n=128	10	0.40488	0.43182
n=150	10	0.44390	0.34046
n=200	10	0.43415	0.43212
n=250	10	0.40000	0.46471
		Micro-Recall	Macro-Recall
n=100	5	0.40976	0.24213
n=128	5	0.40976	0.23658
n=150	5	0.47805	0.25856
n=200	5	0.40976	0.24464
n=250	5	0.44878	0.28812
n=100	10	0.35610	0.29394
n=128	10	0.40488	0.25971
n=150	10	0.44390	0.26791
n=200	10	0.43415	0.30549
n=250	10	0.40000	0.34413
		Micro-F1-score	Macro-F1-score
n=100	5	0.40976	0.12467
n=128	5	0.40976	0.11620
n=150	5	0.47805	0.15327
n=200	5	0.40976	0.12829
n=250	5	0.44878	0.21306
n=100	10	0.35610	0.19817
n=128	10	0.40488	0.14730
n=150	10	0.44390	0.15595
n=200	10	0.43415	0.18483
n=250	10	0.40000	0.16598

したタグを設定することが必要になるだろう。

6 おわりに

本研究では、マルチラベル分類を相談内容である「コメント」とその内容に適する「タグ」を学習し、新規「コメント」に対する「タグ」を予測する機械学習モデルとして BERT を用いた。データのトリミングや様々な評価指標を用いた結果、

1) トリミング無しの Max_len:512 + epoch:5: 0.68551 が最も高い精度となった。

2) F1 値では Max_len:512 + epoch:5: 0.48780 が F1 値の組み合わせで高い性能を示した。

実用化に向けては、予測精度の改善のため、タグの選別とデータ数増加に注力する必要があると考えられる。

謝辞

データを提供していただいた株式会社ここおるに感謝する。

参考文献

[1] メンタルヘルスについて. https://www.mhlw.go.jp/kokoro/mental_health_day/amh.html, 2023. Accessed on 2023/06/12.

[2] ココオルについて. https://cocooru.com/, 2023. Ac-

cessed on 2023/06/12.
[3] 岸田和明, 伊藤夕希也, 門脇夏紀ほか. ゲームソフトの評価レビューに対するマルチラベル分類における svm と bert の比較. 研究報告情報基礎とアクセス技術 (IFAT), Vol. 2022, No. 2, pp. 1–6, 2022.
[4] 竹中姫子, 古宮嘉那子, 小谷善行. ベイジアンフィルターを用いた twitter におけるツイートのハッシュタグ分類. 情報処理学会研究報告, Vol. 2011-IFAT-102, No. 6, pp. 1–6, 04 2011.
[5] 藤井美娜, 阿部智彦, 高橋宏治, 岩城安浩, 加藤恒昭. 契約書のリスク判定のための条文マルチラベル分類. 人工知能学会全国大会論文集, 第 34 回, pp. 4P3OS802–4P3OS802, 2020.
[6] 按田将吾, 菊地真人, 大圃忠親. 製品レビュー中の部品と評価属性に基づくマルチラベル分類のためのラベル抽出手法. 第 84 回全国大会講演論文集, Vol. 2022, No. 1, pp. 349–350, 02 2022.
[7] 安田有希, 牧野仁宣, 岡本大輝, 宮崎太郎, 後藤淳. 入力文とラベル構造との関係に着目したマルチラベル分類手法. 情報科学技術フォーラム講演論文集 (FIT), 第 19th 巻, pp. 133–134, 08 2020.
[8] 深澤祐援, 原島純, 西川荘介. マルチラベル分類による材料推薦モデル. 言語処理学会年次大会発表論文集 (Web), 第 27th 巻, pp. 4–21. 情報処理学会, 情報処理学会, 03 2021.
[9] 新見祐佳, 宮治裕. うつ病を防ぐための感情可視化機能を搭載した認知行動療法. 研究報告自然言語処理 (NL), Vol. 2020, No. 3, pp. 1–5, 2020.
[10] 安道健一郎, 奥村貴史, 小町守, 堀口裕正, 松本裕治. 診療録解析のための文のセグメント分割と意味ラベル付与. 研究報告音声言語情報処理 (SLP), Vol. 2020, No. 17, pp. 1–9, 2020.
[11] Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. How to fine-tune bert for text classification? https://arxiv.org/abs/1905.05583, 2020. Accessed on 2023/06/12.
[12] Bert の精度を向上させる手法 10 選. https://qiita.com/YuiKasuga/items/343309257da1798c1b63, 2023. Accessed on 2023/06/12.
[13] Chris Alberti, Kenton Lee, and Michael Collins. A bert baseline for the natural questions. https://arxiv.org/abs/1901.08634, 2019. Accessed on 2023/06/12.
[14] 小林尚輝, 真鍋陽俊, 小田悠介. 高速な契約書レビューのための計算量の削減. 言語処理学会年次大会発表論文集 (Web), 第 28th 巻, pp. 3–5. 情報処理学会, 情報処理学会, 03 2022.
[15] Iz Beltagy, Matthew E. Peters, and Arman Cohan. Longformer: The long-document transformer. https://arxiv.org/abs/2004.05150, 2020. Accessed on 2023/06/12.
[16] Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer. https://arxiv.org/abs/2001.04451, 2020. Accessed on 2023/06/12.
[17] Manzil Zaheer, Guru Guruganesh, Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. Big bird: Transformers for longer sequences. https://arxiv.org/abs/2007.14062, 2021. Accessed on 2023/06/12.
[18] Raghavendra Pappagari, Piotr Żelasko, Jesús Villalba, Yishay Carmiel, and Najim Dehak. Hierarchical transformers for long document classification. https://arxiv.org/abs/1910.10781, 2019. Accessed on 2023/06/12.
[19] マクロ平均・マイクロ平均法｜機械学習・多クラス分類のための性能評価. https://di-acc2.com/analytics/ai/10801/, 2023. Accessed on 2023/06/12.
[20] 一色政彦. ［評価指標］macro-f1 ／ micro-f1、多クラス分類のマクロ平均／マイクロ平均とは？ https://atmarkit.itmedia.co.jp/ait/articles/2212/19/news020.html, 2023. Accessed on 2023/06/12.