ExhaustiveTree: 網羅的な二次元座標木を用いた表の表現

林　知司†　宮森　恒†

† 京都産業大学大学院 先端情報学研究科　〒 603–8555 京都府京都市北区上賀茂本山
E-mail: †{i2386111,miya}@cc.kyoto-su.ac.jp

あらまし　表形式のデータは，テキストとともに様々な文書中で一般的に使用されるデータ形式であり，多くのアプリケーションにとって重要で不可欠な要素である．しかし，表形式のデータは様々な構造で記述され，その内容理解やテキストとの関連付けは依然として容易ではない．従来研究では，表に含まれる階層的な情報を二次元座標木と呼ばれるツリーベースの構造で表現し，行と列の各インデックスと併用した埋め込みを用いることで，5 つのデータセットで最先端の結果を達成した．しかし，この手法では，表の上 (左) から下 (右) に向かって結合セルの大きさが徐々に小さくなることを前提としており，表内に階層構造がある場合，必ずしも座標を適切に割り当てることができない．本稿では，表中の結合セルの大きさを網羅的に把握した上で，その大きさの降順に二次元座標木を構築する手法を提案する．実験では，表のセルタイプ分類と，表とテキストとの関連付けの 2 種類のタスクにおいて，従来研究と性能を比較することにより，提案手法の有用性を検証する．

キーワード　表データ，表現学習，階層構造，座標木，セルタイプ分類

1　はじめに

表形式のデータは，Web ページやスプレッドシートなど様々な形式が存在し，データを管理するためには，不可欠な要素である．また，情報の根拠を明確にする目的で，テキストと併せて用いられることも多い．しかし，表形式のデータは様々な構造で記述され，その内容理解やテキストとの関連付けは依然として容易ではない．そのため，近年に渡っても表形式のデータを扱う研究が数多くなされている [13] [14] [15] [16] [17].

本稿では，The 17th NTCIR Conference Understanding of non-Financial Objects in Financial Reports [1](以下，UFO) タスクで提供されたデータセットと，deexcelerator(以下，DeEx) データセットを用いて検証する．UFO データセットは，HTML 形式であり，DeEx データセットは，スプレッドシート形式である．UFO タスクでは，有価証券報告書を対象に，そこに含まれる表形式データを理解する 2 つのサブタスクが設定された．Table Data Extraction(以下，TDE) サブタスクでは，与えられた有価証券報告書の表の各セルを，4 つのクラスに分類することを目的とし，Text-to-Table Relationship Extraction(以下，TTRE) サブタスクでは，有価証券報告書中の与えられたテキストに対して，それに関連する表中の該当するセルを選択することを目的としている．

図 1 は，有価証券報告書に含まれる表の例である．有価証券報告書に含まれる表の特徴として，表の右上に金額の単位が記載されるケースや，表の下に金額等の合計が結合セルの形で記載されるケースが多数見受けられた．我々は，有価証券報告書に含まれる表には，表中の結合セルやセル中のインデント等によって作られる複雑な構造が存在することに着目し，これをいくつかの方法で表の表現に取り入れる手法を開発した．

(Unit : One million yen)（単位：百万円）

ヘッジ会計の方法 Hedge accounting method	取引の種類 Type of transaction	主なヘッジ対象 Main hedged subject	契約額等 Contract amount and so on	うち 1 年超 In over one years	時価 Current market price
原則的処理方法 Processing method in general	商品スワップ取引 Swap transactions of Merchandise				
	受取変動・支払固定 Receive-floating ・ Pay-fixed 原油 Crude oil	営業未払金 Non-operating trade payables	69,132	27,452	△24,304
	商品オプション取引 Option transactions of Merchandise				
	売建 Going short プット Put Option 原油 Crude oil	Non-operating trade payables 営業未払金	33,120	15,468	△7,229
	買建 Going long コール Call Option 原油 Crude oil	Non-operating trade payables 営業未払金	42,798	20,103	△1,717
合計 Sum			145,051	63,025	△33,250

図 1　有価証券報告書に含まれる表の例

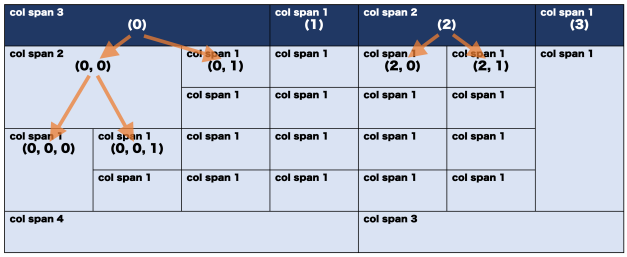

図 2　default vertical tree

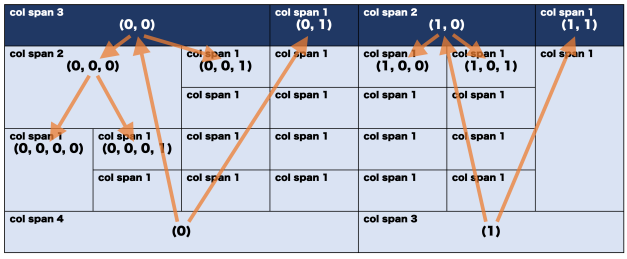

図 3　exhaustive vertical tree

2 関連研究

2.1 TaBERT

TaBERT [11] は自然言語と (半) 構造化テーブルの表現を共同で学習する事前学習済みモデルである．Wikipedia と WDC データセットによる 2,600 万のテーブルとその英語コンテキストからなる大規模なコーパスでトレーニングされている．困難な弱教師付きセマンティック解析ベンチマーク WikiTableQuestions で最先端の性能を達成した．提案手法とは，BERT のアーキテクチャを用いて学習した点において同じである．提案手法とは，表の階層構造を表現して学習した点において異なる．

2.2 TAPAS

TAPAS [11] は論理形式を生成せずにテーブル上で質問に応答する事前学習済みモデルである．BERT のアーキテクチャを拡張してテーブルを入力としてエンコードし，テキストセグメントと Wikipedia のテーブルの効果的な事前トレーニングから初期化し，エンドツーエンドでトレーニングされている．SQA では最先端の性能を達成し，WIKISQL では最先端の精能と同等のパフォーマンスを発揮した．提案手法とは，BERT のアーキテクチャを用いて学習した点において同じである．提案手法とは，表の階層構造を表現して学習した点において異なる．

3 データセット

実装するシステムの構築指針を明らかにするため，UFO タスクで提供された HTML 形式の有価証券報告書集合となるデータコレクションの諸元を分析した．

3.1 TDE

図 4 に TDE タスクの概要図を示す．TDE データセットは，有価証券報告書に含まれる表の各セルを，Metadata，Header，Attribute，Data の 4 つのクラスに分類するためのデータセットである．例では，Metadata は (単位：百万円) と書かれた主に表の外側にあるセルが該当し，Header は第 20 期，第 21 期，第 22 期と書かれた主に見出しのセルが該当し，Attribute は流動資産，固定資産，繰延資産，資産合計と書かれたセルが該当する．それ以外の主に数値などが書かれたセルは Data に該当する．

表 1 TDE データセットの統計

	文書	表	セル
test	190	1,660	45,499
train	252	2,530	66,369
合計	442	4,190	111,868

表 1 に，TDE データセットの分布を示す．TDE データセットは，HTML 形式の有価証券報告書が 442 件あり，test データ，train データにそれぞれ 190 件，252 件に分割されている．また，表は test データ，train データにそれぞれ 1660 件，2530 件あり，セルはそれぞれ 45499 件，66369 件あった．

(単位：百万円)

	第20期	第21期	第22期
流動資産	34,323	98,961	57,158
固定資産	158,763	208,933	189,903
繰延資産	73	241	180
資産合計	193,159	308,135	247,241

	Metadata		
	Header	Header	Header
Attribute	Data	Data	Data
Attribute	Data	Data	Data
Attribute	Data	Data	Data
Attribute	Data	Data	Data

図 4 TDE タスクの概要図

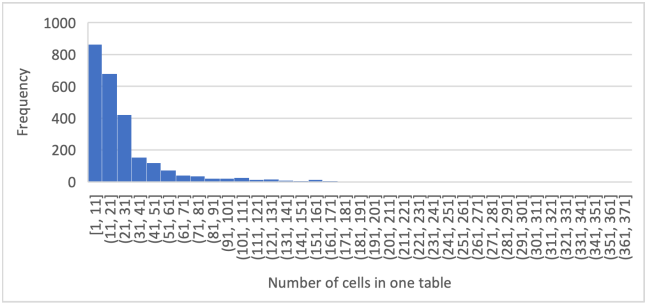

図 5 表に含まれるセル数のヒストグラム

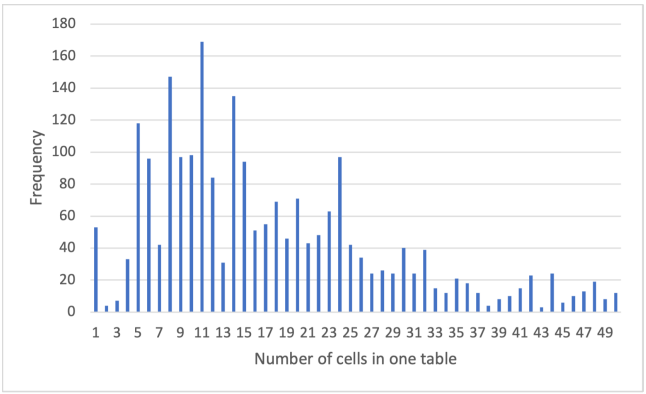

図 6 表に含まれるセル数の棒グラフ (50 個以下のみ)

図 5 に，1 つの表に含まれるセル数のヒストグラムを示す．図より，1 件の表に含まれるセル数はほとんどが 51 個以下であることがわかった．図 6 は，セル数が 50 個以下の表の頻度を

表すグラフである．図より，11 個のセルで構成された表が最も多いことがわかった．

3.2 TTRE

図 7 に TTRE タスクの概要図を示す．TTRE データセットは，有価証券報告書の与えられたテキストに対して，それに関連する表中の Name，Value に該当するセルを選択するためのデータセットである．Name は，所与のテキストが直接的に示しているセルであり，Value は，所与のテキストが間接的に示している数値などが該当する．

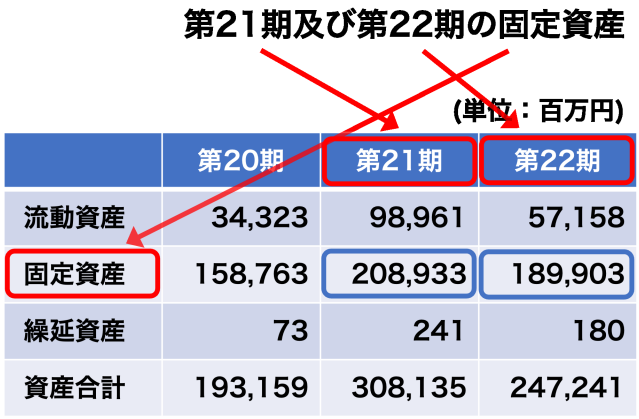

図 7　TTRE タスクの概要図

表 2 に，TTRE データセットの分布を示す．TTRE データセットは，HTML 形式の有価証券報告書が 67 件あり，test データ，train データにそれぞれ 25 件，42 件に分割されている．また，表は test データ，train データにそれぞれ 1125 件，1726 件あり，セルはそれぞれ 47517 件，80644 件あった．

表 3 に，TTRE データセットの train データにおける被検索セルが存在しないフレーズの上位 10 件を示す．数値または記号のみのフレーズが入力された場合，Name，Value に被検索セルが 1 つも該当しない事例が多数あることがわかった．なお，TTRE データセットの train データにおけるフレーズの個数は 3402 件であった．

表 2　TTRE データセットの統計

	文書	表	セル
test	25	1,125	47,517
train	42	1,726	80,644
合計	67	2,851	128,161

4 提 案 手 法

TUTA [2] は，表に含まれる階層的な情報を二次元座標木と呼ばれるツリーベースの構造で表現し，行と列の各インデックスと併用した埋め込みを用いることで，5 つのデータセットで最先端の結果を達成した．TUTA では，表内に階層構造がある場合，表の上 (左) から下 (右) に向かって，結合セルの大きさが徐々に小さくなることを前提としている．しかし，この方法では，図 2 に示すように，その前提に沿ったセルにしか座標を定めることができず，表の途中に出現する大きな結合セルには座標が割り当てられない．そこで，本稿では，表中の結合セルの大きさを網羅的に把握した上で，その大きさの降順に二次元座標木を構築する手法を提案する．

表 3　被検索セルが存在しないフレーズ

フレーズ	頻度	フレーズ	頻度
(1)	50	※ 1	18
2	34	(注)	18
(2)	34	1	18
3	25	(3)	18
4	18	※ 2	15

なお，以下では，TUTA で定まる二次元座標木を default tree と呼び，そのうち垂直方向 (列) と水平方向 (行) に関する木をそれぞれ “default vertical tree”，“default horizontal tree” と呼ぶこととし，提案手法で定まる二次元座標木を “exhaustive tree” と呼び，そのうち垂直方向 (列) と水平方向 (行) に関する木をそれぞれ “exhaustive vertical tree”，“exhaustive horizontal tree” と呼ぶこととする．

4.1 exhaustive tree

提案手法を図 3 を用いて説明する．まず，exhaustive vertical tree について，列方向の結合セル (col span) の最も大きいセルが含まれる 1 列を親ノードに設定する．ただし，セルのサイズが同一であった場合，より上部にあるセルを親ノードに設定する．例の表では，(0)，(1) と記されたの 2 つのセルが該当する．次に，親ノードに該当する各セルについて，列の範囲が内包されるセルのうち，列方向の結合セル (col span) のサイズが次に大きいセルが含まれる 1 列 (列の範囲が内包されるセルのみ) を親ノードに対する子ノードにする．ただし，セルのサイズが同一であった場合，より上部にあるセルを子ノードに設定する．例の表では，親ノード (0) の列方向の結合セル (col span) が 4 に対して，表の左側 4 列の中で，次に列方向の結合セル (col span) が大きいセルは，(0, 0) と記載されたセルであるため，当該の 1 列のうち，列の範囲が内包されるセルを親ノードに対する子ノードにする．ここでは，(0, 0)，(0, 1) と記されたの 2 つのセルが該当する．同様の手順を再起的に繰り返し，列方向の結合セル (col span) のサイズが 1 になったら終了とする．exhaustive horizontal tree についても，同様に，行方向の結合セル (row span) のサイズをもとに木構造を構築する．

4.2 データ変換

HTML 形式の有価証券報告書を，TUTA モデルの入力に適した形式に変換する方法を説明する．

4.2.1 表 抽 出

HTML 文書から table タグで囲まれた範囲を抽出した．ただし，アノテータにより cell-id が 1 つも付与されていない表は

```
<td data-ufo-tde-cell-id="S100LO2J-0102010-tab2-r5c6"
    data-ufo-tde-cell-type="data"
    style="border-left: 1px solid #000000;
        border-top: 1px solid #000000;
        border-right: 1px solid #000000;
        border-bottom: 1px solid #000000;
        vertical-align: middle">
    <p style="margin-right: 42px; text-align: right">
        <span style="font-size: 12px">812.6</span>
    </p>
</td>
```

図 8　HTML 形式のセルの例

表 4　使用したセルの特徴

特徴	説明	例
V	セルの値	812.6
DT	データタイプ	1
HF	数式であるか	0
LB	左部に罫線があるか	1
TB	上部に罫線があるか	1
BB	下部に罫線があるか	1
RB	右部に罫線があるか	1
BC	背景色	#fffff
FC	文字の色	#000000
FB	文字が太字であるか	0

除外した．

4.2.2　表 正 規 化

HTML 形式の表を NFKC(Normalization Form Compatibility Composition) 方式により正規化した．また，△記号は全て-記号に置換した．

4.2.3　Json 形式への変換

HTML 形式の表を TUTA モデルの入力に適した json 形式に変換した．使用したセルの特徴を表 4 に示す．また，HTML 形式のセルの例を図 8 に示す．V はセルの値を表しており，例では「812.6」となる．DT はセルの値のデータタイプを表しており，テキストであれば 0，数値であれば 1，空欄であれば 5 とした．数値であるかの判定は，セルの値の過半数が数値であるかという基準とした．HF は，セルの値に数式が含まれるかを表しており，全て 0 とした．LB，TB，BB，RB は，それぞれ，左部，上部，下部，右部に罫線があれば 1, なければ 0 とした．BC は背景色，FC は文字の色を設定した．FB はフォントが太字か否かを表しており，太字であれば 1，そうでなければ 0 とした．また，表の特徴として，各セルの座標，結合セルの座標，表の木構造も使用した．

4.2.4　英語への翻訳

Google 翻訳 API を用いて，セルの値を翻訳した．日本の通貨単位である「円」を「circle」と誤訳するケースがあったため,「circle」を全て「yen」に置換した．

4.3　TDE

事前学習済み TUTA-implicit モデルを Metadata, Header, Attribute, Data の 4 つのクラスに分類するように微調整した．TUTA には，TUTA-base モデル，TUTA-implicit モデル，TUTA-explicit モデルの 3 種類のモデルがあるが，TUTA-implicit モデルは，学習時に位置埋め込みを使用したモデルである．先行研究で，TUTA-implicit モデルの性能が最も高いことが示されているため，本研究では，TUTA-implicit モデルを使用した．モデルのアーキテクチャは変更せずに使用した．バッチサイズは 2 に設定し，学習率を 8e-6 として，最大 100 エポックまで学習させた．

4.4　TTRE

セルのクラスを考慮したセル検索手法を提案する．図 9 は TTRE サブタスクにおける提案手法の処理概要である．おおまかな流れとしては，まず，所与のフレーズと同一文書内の表の各セルのテキストを Text Encoder に入力し，それぞれの埋め込み表現を取得する．そして，フレーズとセルのテキストの類似度を計算し，Name を決定する．次に，表の特徴を Cell Type Classifier のためのモデルに入力し，表の各セルのクラスを取得する．最後に，Name の情報と，表の各セルのクラスをもとに Value を決定する．

4.4.1　Name

まず，所与のフレーズと同一文書内の表の各セルのテキストを multilingual-e5 [3] モデルにより埋め込み表現を取得した．そして，フレーズとセルのテキストの埋め込み表現のコサイン類似度を計算した．次に，類似度が高い順にランキングし，閾値以上の上位 K 件を Name とした．類似度と取得する件数の閾値は，学習データにより Name の性能が最も高くなる値に設定した．ただし，フレーズが数値または記号のみで構成されていた場合は除外した．また，出現頻度の多い日本語の “注” フレーズも除外した．

4.4.2　Value

表の特徴を TUTA モデルに入力し，表の各セルを Metadata, Header, Attribute, Data の 4 つのクラスに分類した．次に，Name と推定されるセルと同一の行または列に属するセルを抽出し，そのうち，Data クラスに分類されたセルを Value とした．

5　実　　験

本節では，exhaustive tree による手法が，表構造理解に有効であるかを TDE データセットと TTRE データセットを用いて検証する．

5.1　評 価 方 法

5.1.1　TDE

表の木構造を考慮しない手法 (no tree)，従来手法 (default tree)，提案手法 (exhaustive tree) の 3 種類の手法を比較することにより有用性を検証する．全ての手法において，5 回ずつ実行し，マクロ平均の適合率，再現率，F1 値のそれぞれの平均を比較した．

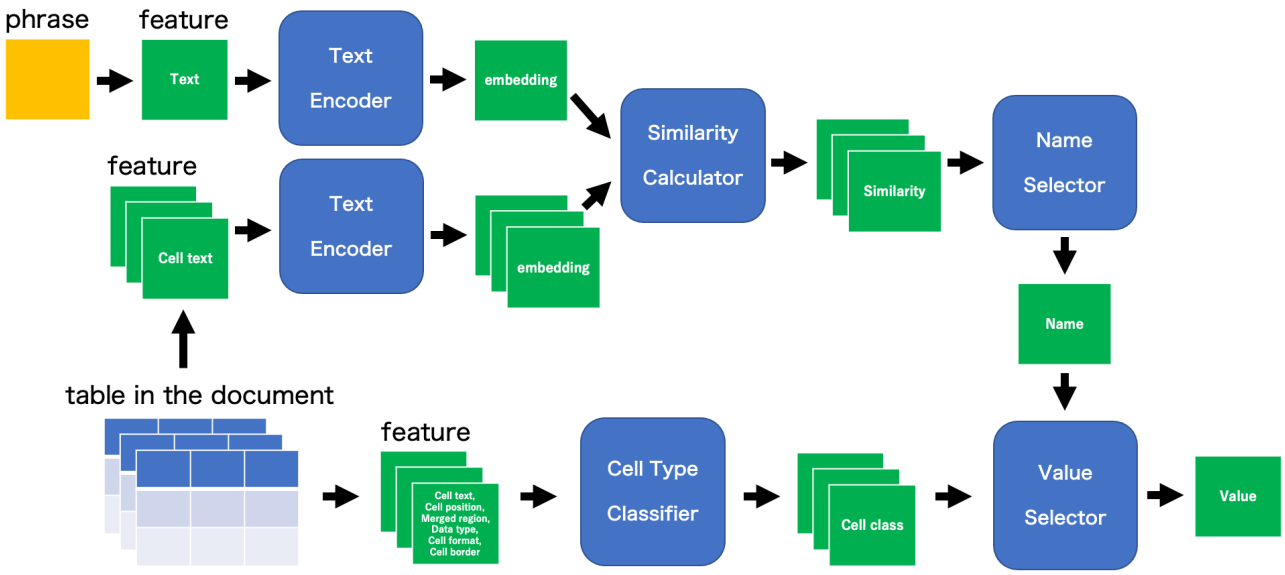

図 9　TTRE サブタスクにおける提案手法の処理概要

表 5　TDE データセットでの評価結果

Method	Metadata	Header	Attribute	Data	マクロ平均		
	F1	F1	F1	F1	適合率	再現率	F1
TUTA $_{\text{no tree}}$	**0.6744**	0.8620	**0.9886**	**0.9046**	**0.8863**	**0.8365**	**0.8574**
TUTA $_{\text{default tree}}$	0.6654	**0.8652**	0.9878	0.9002	0.8847	0.8330	0.8546
TUTA $_{\text{exhaustive tree}}$	**0.6743**	**0.8679**	**0.9901**	**0.9017**	**0.8887**	**0.8375**	**0.8585**

5.1.2　TTRE

表の木構造を考慮しない手法 (no tree)，従来手法 (default tree)，提案手法 (exhaustive tree)，セルのクラスを考慮しない手法の 4 種類の手法を比較することにより有用性を検証する．セルのクラスを考慮しない手法とは，Name と推定されるセルと同一の行または列に属する全てのセルを Value とする手法である．また，Text Encoder の違いによる影響についても検証する．Text Encoder は，multilingual-e5 モデル，sentence-luke モデル [4]，sentence-bert モデル [5]，deberta-v2 モデル [6]，mdeberta-v3 モデル [7] の 5 種類の性能を比較した．

最先端の性能を誇る汎用的なテキスト埋め込みモデルの 1 つに，E5(EmbEddings from bidirEctional Encoder rEpresentations) がある．E5 では，ラベルなしの大規模データセットである CCPairs を用いて弱教師ありの対照学習によって事前学習している．この事前学習済みモデルに対して，ラベル付きの小規模データセットでファインチューニングを施すことで，さらに高品質なテキスト埋め込みを獲得できる．multilingual-e5 は，xlm-roberta-base の事前学習済みパラメータを初期値として，多種多様な多言語データセット (Filtered mC4，CC News，NLLB，Wikipedia，Filtered Reddit，S2ORC，Stackexchange，xP3，Miscellaneous unsupervised SBERT data) を用いた対照学習による事前学習をしている．この多言語データセットで事前学習したモデルに対して，ラベル付きのデータセット (MS MARCO，NQ，Trivia QA，NLI from SimCSE，ELI5，DuReader Retrieval，KILT Fever，KILT HotpotQA，SQuAD，Quora，Mr. TyDi，MIRACL) を用いてファインチューニングしている．

LUKE は，BERT での事前学習タスクに加え，新たに単語とエンティティの文脈化表現を学習したモデルであり，エンティティに関する様々なタスクにおいて最先端の性能に到達している．このモデルは，与えられたテキスト中の単語とエンティティを独立したトークンとして扱い，それらの文脈化表現を出力する．ここではエンティティに着目するため，自己注意機構を拡張することで，単語およびエンティティを考慮したスコアを算出可能にしている．

SBERT は，従来の BERT よりも優れた文埋め込みを獲得するために，2 つの BERT のそれぞれの出力にプーリング層を追加したシャムネットワーク構造のモデルである．この SBERT は，従来の BERT よりも文埋め込みの獲得だけではなく，推論時間の大幅な削減も実現した．

DeBERTa は，BERT と RoBERTa を改良したモデルであり，各単語とその単語の相対的な出現位置をディスエンタング

表 6　TTRE データセットでの評価結果

Method	Name			Value			Total
	適合率	再現率	F1	適合率	再現率	F1	F1
multilingual-e5-small	**0.3687**	**0.4434**	0.3198	0.0898	0.4939	0.1154	0.2176
multilingual-e5-small + TUTA $_{\text{no tree}}$	**0.3687**	**0.4434**	0.3198	**0.2921**	0.4918	**0.2708**	0.2953
multilingual-e5-small + TUTA $_{\text{default tree}}$	**0.3687**	**0.4434**	0.3198	0.2682	0.4867	0.2562	0.2880
multilingual-e5-small + TUTA $_{\text{exhaustive tree}}$	**0.3687**	**0.4434**	0.3198	0.2794	0.4911	0.2604	0.2901
multilingual-e5-base	0.3556	**0.4574**	**0.3221**	0.0857	**0.5069**	0.1186	0.2204
multilingual-e5-base + TUTA $_{\text{no tree}}$	0.3556	**0.4574**	**0.3221**	0.2733	**0.5048**	**0.2742**	**0.2982**
multilingual-e5-base + TUTA $_{\text{default tree}}$	0.3556	**0.4574**	**0.3221**	0.2514	0.5001	0.2596	0.2909
multilingual-e5-base + TUTA $_{\text{exhaustive tree}}$	0.3556	**0.4574**	**0.3221**	0.2613	0.5044	0.2639	0.2930
multilingual-e5-large	**0.3728**	0.4355	**0.3212**	0.0885	0.4813	0.1153	0.2182
multilingual-e5-large + TUTA $_{\text{no tree}}$	**0.3728**	0.4355	**0.3212**	**0.2874**	0.4799	0.2704	**0.2958**
multilingual-e5-large + TUTA $_{\text{default tree}}$	**0.3728**	0.4355	**0.3212**	0.2647	0.4751	0.2566	0.2889
multilingual-e5-large + TUTA $_{\text{exhaustive tree}}$	**0.3728**	0.4355	**0.3212**	0.2757	0.4793	0.2613	0.2912
sentence-luke (Japanese)	0.3401	0.4213	0.3000	0.0827	0.4643	0.1120	0.2060
sentence-luke (Japanese) + TUTA $_{\text{no tree}}$	0.3401	0.4213	0.3000	0.2668	0.4633	0.2592	0.2796
sentence-luke (Japanese) + TUTA $_{\text{default tree}}$	0.3401	0.4213	0.3000	0.2436	0.4587	0.2446	0.2723
sentence-luke (Japanese) + TUTA $_{\text{exhaustive tree}}$	0.3401	0.4213	0.3000	0.2586	0.4629	0.2528	0.2764
sentence-bert (Japanese)	0.3237	0.3948	0.2805	0.0798	0.4493	0.1067	0.1936
sentence-bert (Japanese) + TUTA $_{\text{no tree}}$	0.3237	0.3948	0.2805	0.2567	0.4477	0.2463	0.2634
sentence-bert (Japanese) + TUTA $_{\text{default tree}}$	0.3237	0.3948	0.2805	0.2363	0.4434	0.2331	0.2568
sentence-bert (Japanese) + TUTA $_{\text{exhaustive tree}}$	0.3237	0.3948	0.2805	0.2470	0.4476	0.2381	0.2593
deberta-v2-tiny (Japanese)	0.2969	0.3922	0.2753	0.0735	0.4288	0.1029	0.1891
deberta-v2-tiny (Japanese) + TUTA $_{\text{no tree}}$	0.2969	0.3922	0.2753	0.2399	0.4273	0.2441	0.2597
deberta-v2-tiny (Japanese) + TUTA $_{\text{default tree}}$	0.2969	0.3922	0.2753	0.2204	0.4225	0.2299	0.2526
deberta-v2-tiny (Japanese) + TUTA $_{\text{exhaustive tree}}$	0.2969	0.3922	0.2753	0.2304	0.4267	0.2353	0.2553
deberta-v2-base (Japanese)	0.3013	0.3992	0.2810	0.0757	0.4254	0.1051	0.1931
deberta-v2-base (Japanese) + TUTA $_{\text{no tree}}$	0.3013	0.3992	0.2810	0.2443	0.4231	0.2475	0.2642
deberta-v2-base (Japanese) + TUTA $_{\text{default tree}}$	0.3013	0.3992	0.2810	0.2249	0.4194	0.2337	0.2574
deberta-v2-base (Japanese) + TUTA $_{\text{exhaustive tree}}$	0.3013	0.3992	0.2810	0.2334	0.4233	0.2373	0.2591
deberta-v2-large (Japanese)	0.3151	0.3823	0.2739	0.0765	0.4165	0.1015	0.1877
deberta-v2-large (Japanese) + TUTA $_{\text{no tree}}$	0.3151	0.3823	0.2739	0.2511	0.4152	0.2390	0.2564
deberta-v2-large (Japanese) + TUTA $_{\text{default tree}}$	0.3151	0.3823	0.2739	0.2323	0.4107	0.2260	0.2500
deberta-v2-large (Japanese) + TUTA $_{\text{exhaustive tree}}$	0.3151	0.3823	0.2739	0.2409	0.4146	0.2306	0.2523
mdeberta-v3-base	0.2345	0.2665	0.1983	0.0547	0.2791	0.0700	0.1342
mdeberta-v3-base + TUTA $_{\text{no tree}}$	0.2345	0.2665	0.1983	0.2014	0.2786	0.1822	0.1903
mdeberta-v3-base + TUTA $_{\text{default tree}}$	0.2345	0.2665	0.1983	0.1832	0.2773	0.1721	0.1852
mdeberta-v3-base + TUTA $_{\text{exhaustive tree}}$	0.2345	0.2665	0.1983	0.1898	0.2782	0.1729	0.1856

ル注意機構を用いることで，独立した 2 つのベクトルとしての表現を可能にしている．また事前学習時には，デコーダを単語の絶対的な出現位置をマスクできるように拡張している．それに加え，ファインチューニング時には仮想敵対的学習を行っている．DeBERTa V2 では，トークナイザーの語彙力が増強され，入力トークンの局所依存性を学習するために畳み込み層を追加し，注意機構では位置射影行列と内容射影行列は共有され，相対位置を符号化するためにバケットを使用している．DeBERTa V3 は，DeBERTa の改良版であり，Gradient Disentangled Embedding Sharing を使用している．mDeBERTa は，多言語データセットである CC100 を用いて DeBERTa を学習したモデルである．

sentence-luke モデル，sentence-bert モデル，deberta-v2 モデルは日本語のモデルを使用し，multilingual-e5 モデル，mdeberta-v3 モデルは多言語モデルを使用した．また，deberta-v2 モデルは，トークナイズの際に，Juman++ 2.0.0-rc3 [8] [9] を使用した．類似度と取得する件数の閾値は，学習データにより Name の性能が最も高くなる値に設定した．

5.2　結果と考察

5.2.1　TDE

表 5 に TDE データセットでの評価結果を示す．マクロ平均の適合率，再現率，F1 スコアにおいて，exhaustive tree による手法，no tree による手法，default tree による手法の順で高い性能を示す結果となった．全ての手法において，Header，Attribute，Data クラスに比べ，Metadata クラスに対しては

低い性能を示す結果となった．これは，Metadata クラスに分類される事例が極端に少なく，効率的に学習できなかったことが原因として挙げられる．

5.2.2 TTRE

表 6 に TTRE サブタスクの評価結果を示す．Text Encoder の違いによる影響については，multilingual-e5-base による手法が Name において最も高い性能を示す結果となった．Cell Type Classifier においては，no tree による手法が全ての Value，Total の F1 スコアにおいて最も高い性能を示す結果となった．また，Text Encoder に multilingual-e5-base，Cell Type Classifier に no tree による手法を用いた場合，29.82%と最も高い性能を示す結果となった．

no tree による手法が最も高い性能を示した理由として，TDE サブタスクでの，Data クラスを分類するためのモデルの性能が，TTRE サブタスクに大きな影響を与えるという仮説を立てた．TDE データセットを用いて，Data クラスを分類するためのモデルの性能を検証した結果，no tree による手法が最も高い性能を示した．この結果から，Data クラスを分類するためのモデルの性能を向上させることが重要であることがわかった．

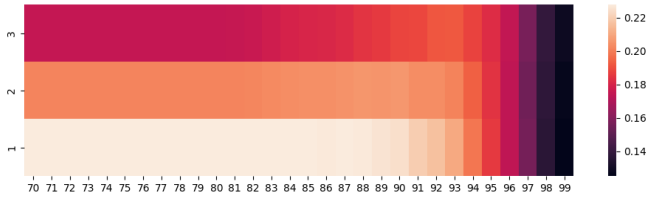

図 10　Total における類似度と上位 K 件による閾値ごとの評価結果のヒートマップ (multilingual-e5)

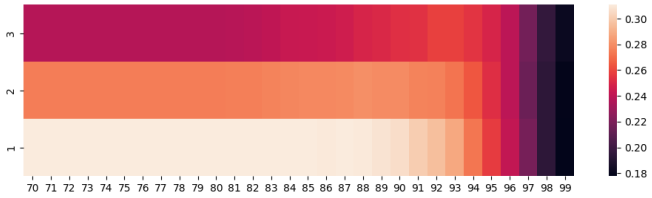

図 11　Total における類似度と上位 K 件による閾値ごとの評価結果のヒートマップ (multilingual-e5 + TUTA $_{\text{no tree}}$)

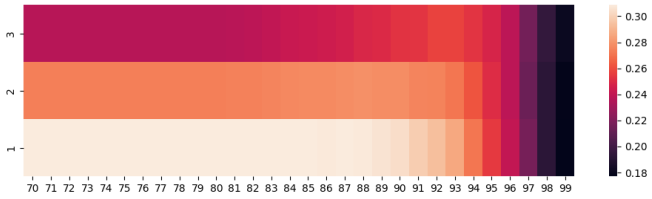

図 12　Total における類似度と上位 K 件による閾値ごとの評価結果のヒートマップ (multilingual-e5 + TUTA $_{\text{default tree}}$)

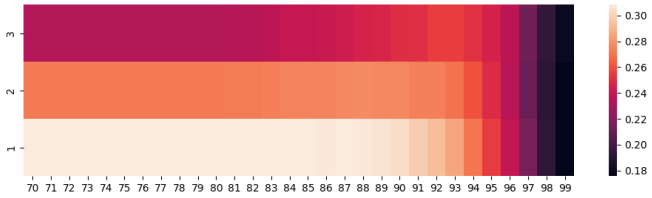

図 13　Total における類似度と上位 K 件による閾値ごとの評価結果のヒートマップ (multilingual-e5 + TUTA $_{\text{exhaustive tree}}$)

図 10〜13 に Total における類似度と上位 K 件による閾値ごとの評価結果のヒートマップ Total における類似度と上位 K 件による閾値ごとの評価結果のヒートマップを示す．縦軸は取得する被検索セルの個数の上限を表しており，横軸は取得する被検索セルのテキストと所与のテキストにおけるコサイン類似度の下限を表している．全ての手法において，所与のテキストに対して上位 1 件のセルを取得した場合に高い性能を示しており，複数セルの検索については，十分な性能を発揮できていないことが示唆された．

以下では，多言語モデルと日本語のモデルにおける性能の違いについて分析する．multilingual-e5-base は多言語モデルの中で最も高い性能を示したモデルであり，sentence-luke は日本語のモデルの中で最も高い性能を示したモデルである．TUTA を使用した手法と，TUTA を使用しなかった手法で，Value を比較すると，再現率には大きな差がなく，適合率には有意義な差があった．このことから，Value を決定する際に，セルのタイプを判定し，data クラス以外を除外することは非常に有力であるといえる．また，全ての手法で，適合率よりも再現率の方が高い結果となった．これは，Name が，所与のフレーズとセルの値の類似度のみで決められているため，実際に参照すべき表とは異なる表もまとめて取得してしまっていることが原因として考えられる．

表 7　多言語モデルと日本語モデルにおける予測結果の一致度合い

	頻度 (割合)
両方とも正解	441 (0.235)
多言語モデルのみ正解	40 (0.021)
日本語モデルのみ正解	33 (0.018)
両方とも不正解	677 (0.361)

表 7 に，Name における multilingual-e5-base を用いた手法と sentence-luke を用いた手法による予測結果の一致度合いを示す．両方の手法で正しく識別できた件数は 441 件であり，全体の 23.5%を占める結果となった．多言語モデルのみ正解できた件数は 40 件であり，全体の 2.1%を占める結果となった．日本語モデルのみ正解できた件数は 33 件であり，全体の 1.8%を占める結果となった．両方の手法で全く同じ誤りをした件数は 677 件であり，全体の 36.1%を占める結果となった．

6　追 加 実 験

本節では，exhaustive tree による手法が，表構造理解に有効であるかを DeEx データセットを用いて検証する．DeEx データセットは，セルタイプ分類のためのデータセットである．分類するクラスは，metadata, notes, data, attributes, header, derived の 6 種類であり，それぞれの分布を表 9 に示す．DeEx データセットは，data クラスに分類されるセルがほとんどであるといった特徴がある．一方で，notes クラスの分類されるセルはごく僅かであるといった特徴がある．

表 8　DeEx データセットでの評価結果

Method	metadata	notes	data	attributes	header	derived	macro-F1
TUTA $_{\text{no tree}}$	0.8574	**0.4825**	**0.9937**	**0.8114**	**0.8701**	**0.8092**	**0.8041**
TUTA $_{\text{default tree}}$	**0.8585**	0.4419	0.9930	**0.8382**	0.8547	0.7391	0.7876
TUTA $_{\text{exhaustive tree}}$	**0.8671**	**0.5589**	**0.9931**	0.8045	**0.8638**	0.7491	**0.8061**
TUTA $_{\text{raw tree}}$	0.8516	0.4271	0.9911	0.7888	0.8070	**0.7767**	0.7737

表 9　セルタイプの分布

セルタイプ	頻度 (割合)
metadata	20,020 (0.0154)
notes	5,314 (0.0041)
data	1,232,762 (0.9458)
attributes	7,024 (0.0054)
header	21,810 (0.0167)
derived	16,538 (0.0127)

6.1 評価方法

従来の default tree による手法，提案手法の exhaustive tree による手法，表の木構造を考慮しない no tree による手法，TUTA の著者らにより提供された raw tree の 4 つの手法を比較することにより有用性を検証する．事前学習済み TUTA-implicit モデルを metadata, notes, data, attributes, header, derived の 6 つのクラスに分類するように微調整した．モデルのアーキテクチャは変更せずに使用した．データセットは，DeEx データセットを無作為に 5 分割されたものを使用した．検証には，交差検証法を用い，平均マクロ F1 を算出した．バッチサイズは 2 に設定し，学習率を 8e-6 として，最大 200 エポックまで学習させた．

6.2 結果と考察

表 8 に DeEx データセットでの実験結果を示す．提案手法がマクロ F1 において，最も性能の高い結果となった．data クラスの F1 スコアが最も高く，notes クラスの F1 スコアが最も低い結果となった．data クラスの個数が最も多く，notes クラスの個数が最も少ないことから，セルタイプの分布が大きく影響していると考えられる．提案手法が notes クラスにおいて，最も性能の高い結果となったことから，提案手法が少数のデータからも効果的に学習する能力が高い可能性がある．

表 10　表データにおける木構造で表現した際の深度

木の種類	TDE		DeEx	
	平均	標準偏差	平均	標準偏差
default vertical tree	0.3658	1.0948	0.4444	2.2411
default horizontal tree	0.2162	0.6906	0.1995	1.0503
exhaustive vertical tree	0.3919	0.8152	0.5079	1.0413
exhaustive horizontal tree	0.2255	0.6488	0.1814	0.6380
合計	1.1994	-	1.3332	-

TDE データセットと DeEx データセットを用いて，表データを木構造で表現した際の深度を調査した．その結果，表 10 に示すように，TDE データセットと DeEx データセットは複雑な表が少ないことがわかった．そのため，no tree による手法と提案手法で，あまり性能の差が見られなかった原因として，データセットに複雑な表が少なかったためであると結論づけた．

7 まとめ

本稿では，表中の結合セルの大きさを網羅的に把握した上で，その大きさの降順に二次元座標木を構築する手法を提案した．表のセルタイプ分類と，表とテキストとの関連付けの 2 種類のタスクにおいて，従来研究と性能を比較することにより，提案手法の有用性を検証した．その結果，提案手法は，表が複雑な構造をしている際に最も活躍することがわかった．表とテキストとの関連付けのタスクにおいては，セルを検索する際に，Data クラスに該当するセルを除外する手法は有効的であった．ただし，セルタイプ分類の総合的な性能ではなく，Data クラスを抽出するための性能が重要であることがわかった．

謝　辞

本研究の一部は科研費 23K11342 の助成を受けたものである．

文　献

[1] Yasutomo Kimura and Hokuto Ototake and Kazuma Kadowaki and Takahito Kondo and Makoto P. Kato, “Overview of the NTCIR-17 UFO Task,” Proceedings of The 17th NTCIR Conference, 2023.

[2] Wang, Zhiruo and Dong, Haoyu and Jia, Ran and Li, Jia and Fu, Zhiyi and Han, Shi and Zhang, Dongmei, “TUTA: Tree-based Transformers for Generally Structured Table Pre-training,” Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, pp. 1780–1790, 2021.

[3] Wang, Liang and Yang, Nan and Huang, Xiaolong and Jiao, Binxing and Yang, Linjun and Jiang, Daxin and Majumder, Rangan and Wei, Furu, “Text Embeddings by Weakly-Supervised Contrastive Pre-training,” 2022.

[4] Ikuya Yamada and Akari Asai and Hiroyuki Shindo and Hideaki Takeda and Yuji Matsumoto, “LUKE: Deep Contextualized Entity Representations with Entity-aware Self-attention,” EMNLP, 2020.

[5] Reimers, Nils and Gurevych, Iryna, “Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks,” Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing, 2019.

[6] Pengcheng He and Xiaodong Liu and Jianfeng Gao and Weizhu Chen, “DEBERTA: DECODING-ENHANCED BERT WITH DISENTANGLED ATTENTION,” International Conference on Learning Representations, 2021.

[7] Pengcheng He and Xiaodong Liu and Jianfeng Gao and Weizhu Chen, “DEBERTA: DECODING-ENHANCED

BERT WITH DISENTANGLED ATTENTION,” International Conference on Learning Representations, 2021.
[8] Morita, Hajime and Kawahara, Daisuke and Kurohashi, Sadao, “Morphological Analysis for Unsegmented Languages using Recurrent Neural Network Language Model,” Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pp. 2292–2297, 2015.
[9] Tolmachev, Arseny and Kawahara, Daisuke and Kurohashi, Sadao, “Juman++: A Morphological Analysis Toolkit for Scriptio Continua,” Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, pp. 54–59, 2018.
[10] Koci, Elvis and Thiele, Maik and Rehak, Josephine and Romero, Oscar and Lehner, Wolfgang, “DECO: A Dataset of Annotated Spreadsheets for Layout and Table Recognition,” 2019 International Conference on Document Analysis and Recognition (ICDAR), pp. 1280-1285, 2019.
[11] Pengcheng Yin and Graham Neubig and Wen-tau Yih and Sebastian Riedel, “TaBERT: Pretraining for Joint Understanding of Textual and Tabular Data”, 2020.
[12] Herzig, Jonathan and Nowak, Pawel Krzysztof and Müller, Thomas and Piccinno, Francesco and Eisenschlos, Julian, “TaPas: Weakly supervised table parsing via pre-training”, 2020.
[13] Jixiong Liu and Yoan Chabot and Raphaël Troncy and Viet-Phi Huynh and Thomas Labbé and Pierre Monnin, “From tabular data to knowledge graphs: A survey of semantic table interpretation tasks and methods,” Journal of Web Semantics, Volume 76, 2023.
[14] Gilbert Badaro, Mohammed Saeed, Paolo Papotti, “Transformers for Tabular Data Representation: A Survey of Models and Applications,” 2023,
[15] Zilong Wang and Hao Zhang and Chun-Liang Li and Julian Martin Eisenschlos and Vincent Perot and Zifeng Wang and Lesly Miculicich and Yasuhisa Fujii and Jingbo Shang and Chen-Yu Lee and Tomas Pfister, “Chain-of-Table: Evolving Tables in the Reasoning Chain for Table Understanding,” 2024,
[16] Tianshu Zhang and Xiang Yue and Yifei Li and Huan Sun, “TableLlama: Towards Open Large Generalist Models for Tables,” 2023,
[17] Mehmed Kantardzic and Associate Editor and Witold Pedrycz and Editor-in-Chief, “Table understanding: Problem overview,” 2022,

付　　録

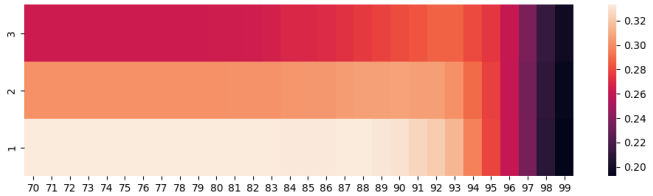

図 14　Name における類似度と上位 K 件による閾値ごとの評価結果のヒートマップ (multilingual-e5)

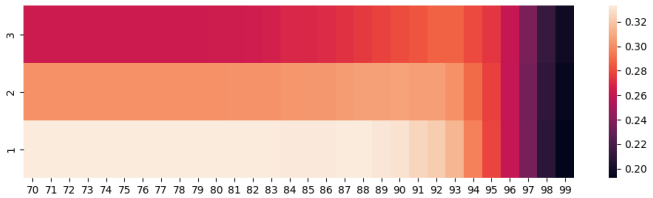

図 15　Name における類似度と上位 K 件による閾値ごとの評価結果のヒートマップ (multilingual-e5 + TUTA $_{\text{no tree}}$)

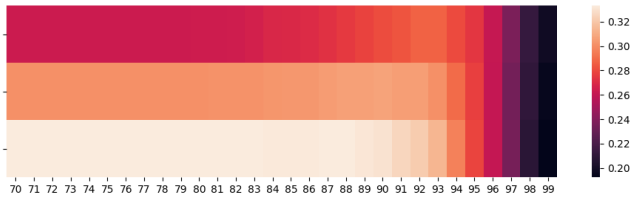

図 16　Name における類似度と上位 K 件による閾値ごとの評価結果のヒートマップ (multilingual-e5 + TUTA $_{\text{default tree}}$)

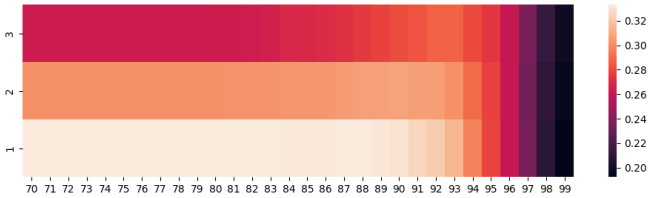

図 17　Name における類似度と上位 K 件による閾値ごとの評価結果のヒートマップ (multilingual-e5 + TUTA $_{\text{exhaustive tree}}$)

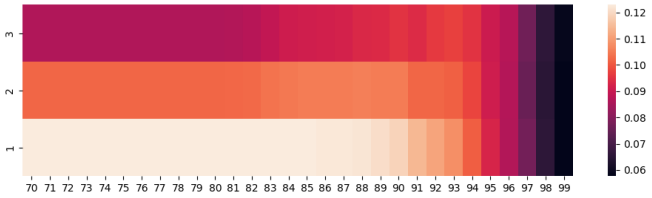

図 18　Value における類似度と上位 K 件による閾値ごとの評価結果のヒートマップ (multilingual-e5)

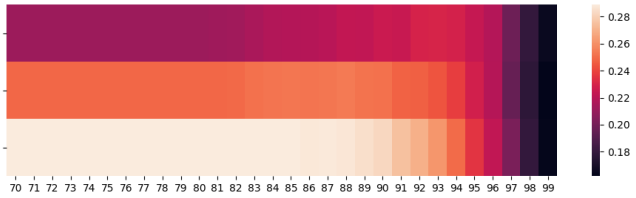

図 19　Value における類似度と上位 K 件による閾値ごとの評価結果のヒートマップ (multilingual-e5 + TUTA $_{\text{no tree}}$)

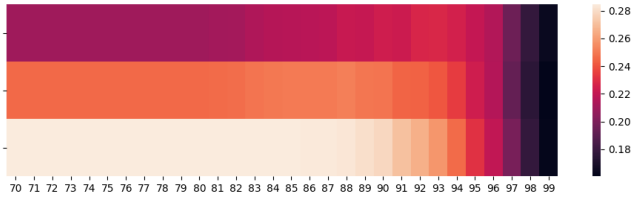

図 20　Value における類似度と上位 K 件による閾値ごとの評価結果のヒートマップ (multilingual-e5 + TUTA $_{\text{default tree}}$)

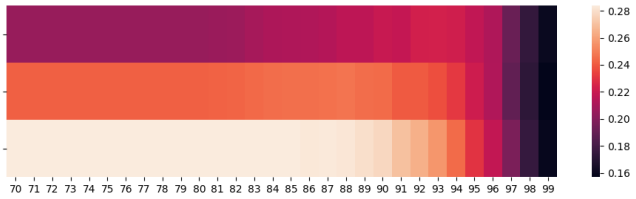

図 21　Value における類似度と上位 K 件による閾値ごとの評価結果のヒートマップ (multilingual-e5 + TUTA $_{\text{exhaustive tree}}$)