ベクトル化手法と機械学習の組合せに着目したコメントスメル検出に関する研究

工学部 工学科 応用情報工学コース
ヒューマンウェア工学研究室
B4 堂元優佑

研究概要

- **目的**
  機械学習での**コメントスメル**の自動検出に有効な
  組合せの特定

- **方法**
  ベクトル化のタイミング，ベクトル化手法，機械学習モデル
  を組み合わせた**70種類**から有効な組合せを検討

- **結果**
  - **ソースコードとコメントをベクトル化し統合したもの**を説明変数に利用することが有効
  - ベクトル化手法と機械学習モデルの中から**有用な手法・モデル**を確認

コメントスメル

コードの可読性や保守性を**低下**させるコメント
コード理解に**役立たない**コメント

- **Beautification**
  ソースコード内で意味的に関連した宣言部の区別を目的とするコメント

- **Obvious**
  ソースコードの表面的な動作をそのまま説明しているコメント

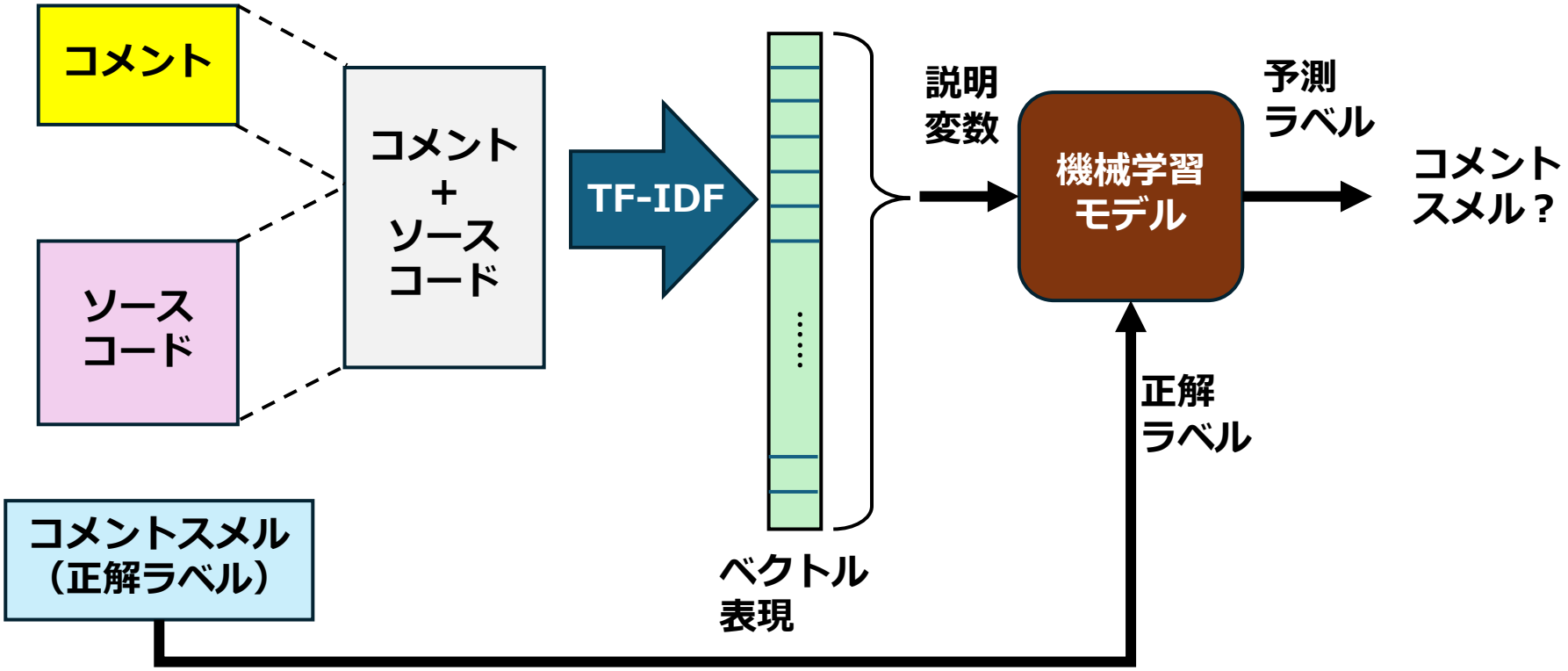
背景

- コメントスメルは可読性を低下させてしまうため，自動検出することは重要

- 機械学習や大規模言語モデルで自動検出の研究が行われている
  - 機械学習と大規模言語モデルでの**検出精度に大きな差**がない
  - 機械学習による自動検出には改善点が存在

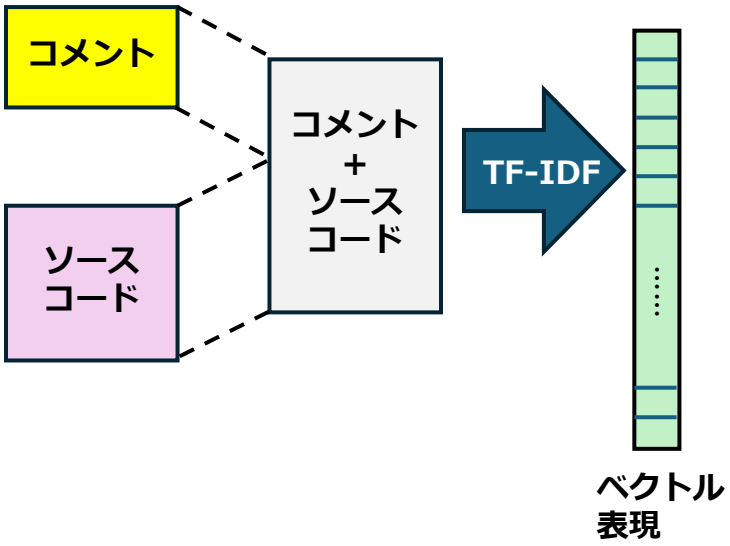
コメントスメルに有効な**手法・モデル**はどれか

先行研究でのコメントスメル検出の流れ

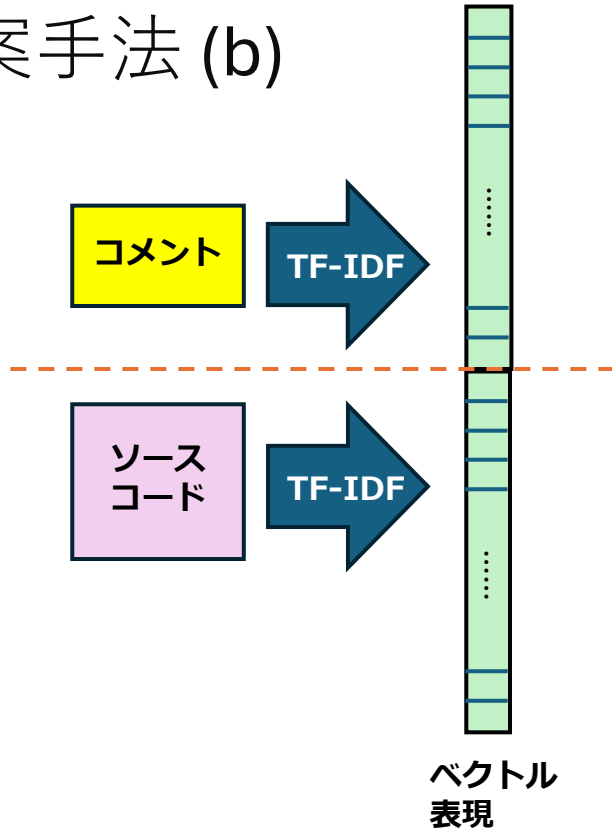


ベクトル化のタイミング

従来手法 (a)



提案手法 (b)



ベクトル化手法

(A) TF-IDF

(B) BM25

(C) Word2Vec

(D) CodeT5+

(E) CodeBERT

機械学習モデル

(1) 勾配ブースティング

(2) ランダムフォレスト

(3) 決定木

(4) サポートベクターマシン

(5) ロジスティック回帰モデル

(6) ナイーブベイズ

(7) k-近傍法

研究課題（Research Question: RQ）

RQ: 機械学習を用いたコメントスメルの自動検出において以下の**3つの観点**ではどういった組合せが有用か？

- **観点1:** **コメントとソースコードの与え方**（コメントとソースコードを別々に与えるか2つを つなげて与えるか）
- **観点2:** コメントとソースコードに対する**ベクトル化手法**
- **観点3:** 自動検出に用いる**機械学習モデル**

実験方法

Oztas, I., Torun, U. B. and Tüzün, E.: Towards Automated Detection of Inline Code Comment Smells, in Proceedings of the 29th International Conference on Evaluation and Assessment in Software Engineering, pp. 1127–1137(2025)

Oztasらの先行研究に用いられたコメントと関連するソースコード，コメントスメルがセットとなったデータ2,211個を使用

1. ベクトル化のタイミング
2. ベクトル化手法
3. 機械学習モデル

上記の3つの観点を組み合わせて70種類で実験

実験の組合せ

(1) 勾配ブースティング
(2) ランダムフォレスト
(3) 決定木
(4) サポートベクターマシン
(5) ロジスティック回帰モデル
(6) ナイーブベイズ
(7) k-近傍法

ベクトル化手法		機械学習モデル 1	2	3	4	5	6	7		ベクトル化のタイミング
(A) TF-IDF (B) BM25 (C) Word2Vec (D) CodeT5+ (E) CodeBERT	A	a-A-1	a-A-2	a-A-3	a-A-4	a-A-5	a-A-6	a-A-7	a	
		b-A-1	b-A-2	b-A-3	b-A-4	b-A-5	b-A-6	b-A-7	b	
	B	a-B-1	a-B-2	a-B-3	a-B-4	a-B-5	a-B-6	a-B-7	a	
		b-B-1	b-B-2	b-B-3	b-B-4	b-B-5	b-B-6	b-B-7	b	
	C	a-C-1	a-C-2	a-C-3	a-C-4	a-C-5	a-C-6	a-C-7	a	
		b-C-1	b-C-2	b-C-3	b-C-4	b-C-5	b-C-6	b-C-7	b	
	D	a-D-1	a-D-2	a-D-3	a-D-4	a-D-5	a-D-6	a-D-7	a	
		b-D-1	b-D-2	b-D-3	b-D-4	b-D-5	b-D-6	b-D-7	b	
	E	a-E-1	a-E-2	a-E-3	a-E-4	a-E-5	a-E-6	a-E-7	a	
		b-E-1	b-E-2	b-E-3	b-E-4	b-E-5	b-E-6	b-E-7	b	

赤文字が先行研究の手法

スメル分類

- 5種類のコメントスメルとコメントスメルではないもの（Not a smell）の6種類を分類

  - Beautification
  - Commented-out code
  - Obvious
  - Task
  - Vague
  - Not a smell

Beautification検出のF1スコア

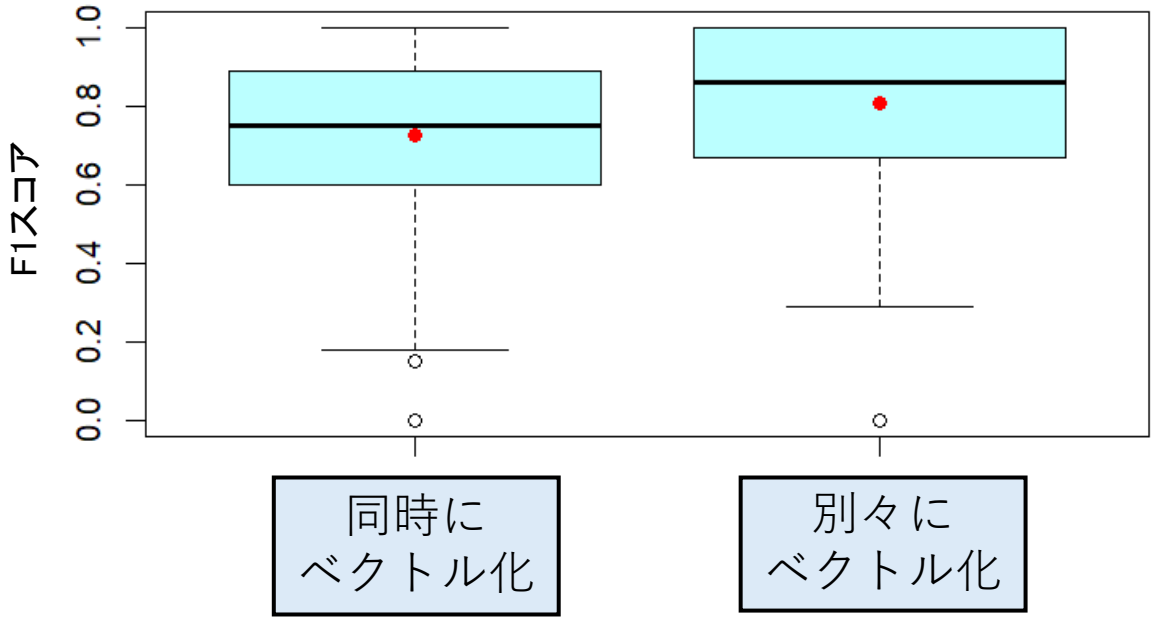
サポートベクターマシン

	1	2	3	4	5	6	7	
A	0.83	0.88	0.85	0.89	0.93	0.35	0.70	a
	0.87	0.86	0.87	0.91	0.88	0.56	0.70	b
B	0.89	0.92	0.86	0.69	0.84	0.58	0.73	a
	0.89	0.93	0.90	0.71	0.91	0.56	0.90	b
C	0.66	0.63	0.57	0.40	0.41	0.39	0.56	
	0.85	0.80	0.70	0.78	0.80	0.65	0.73	
D	0.80	0.80	0.30	0.91	0.86	0.62	0.83	a
	0.86	0.89	0.67	**0.95**	0.93	0.65	0.91	b
E	0.86	0.86	0.81	0.75	0.89	0.71	0.88	a
	0.82	0.90	0.78	0.71	0.93	0.66	0.92	b

CodeT5+

別々に
ベクトル化

観点1の比較結果

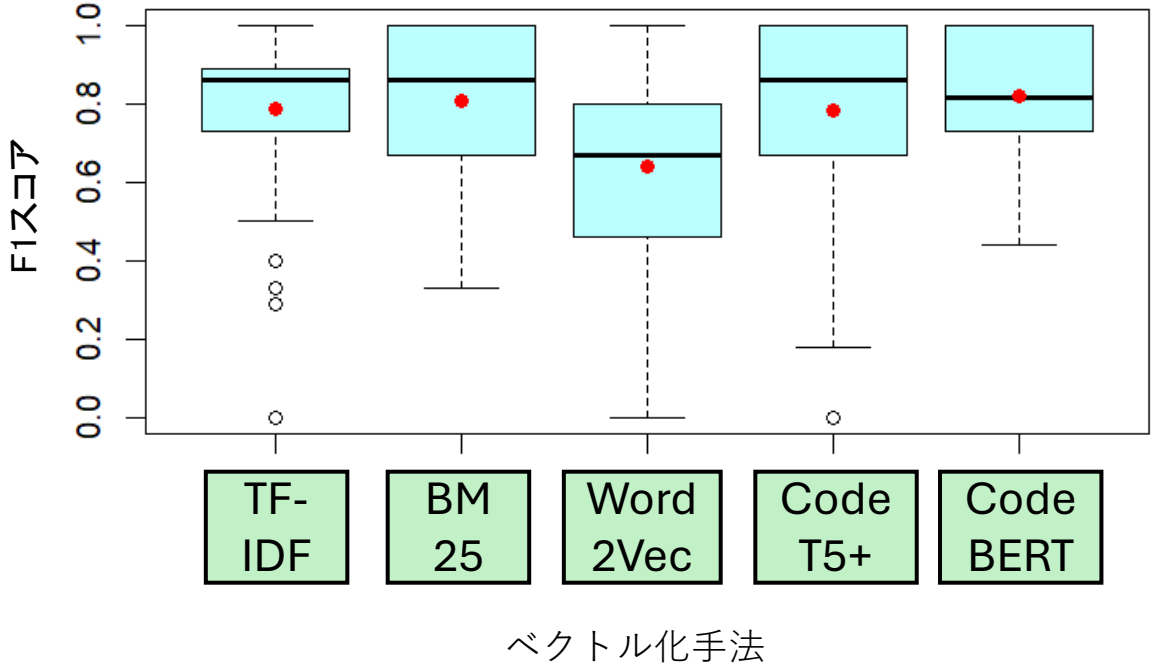


観点1の考察

- どの分類でも同時にベクトル化するよりも**別々にベクトル化した方が高い精度**を記録
  - ソースコードとコメントという**プログラミング言語**と**自然言語**という別のものを適切にベクトル化
  - スメル特有の**記号**や**文字列**をより表現可能に

- Word2Vecで特にbの精度が向上
  - ベクトルの次元数増加の効果

観点2の比較結果



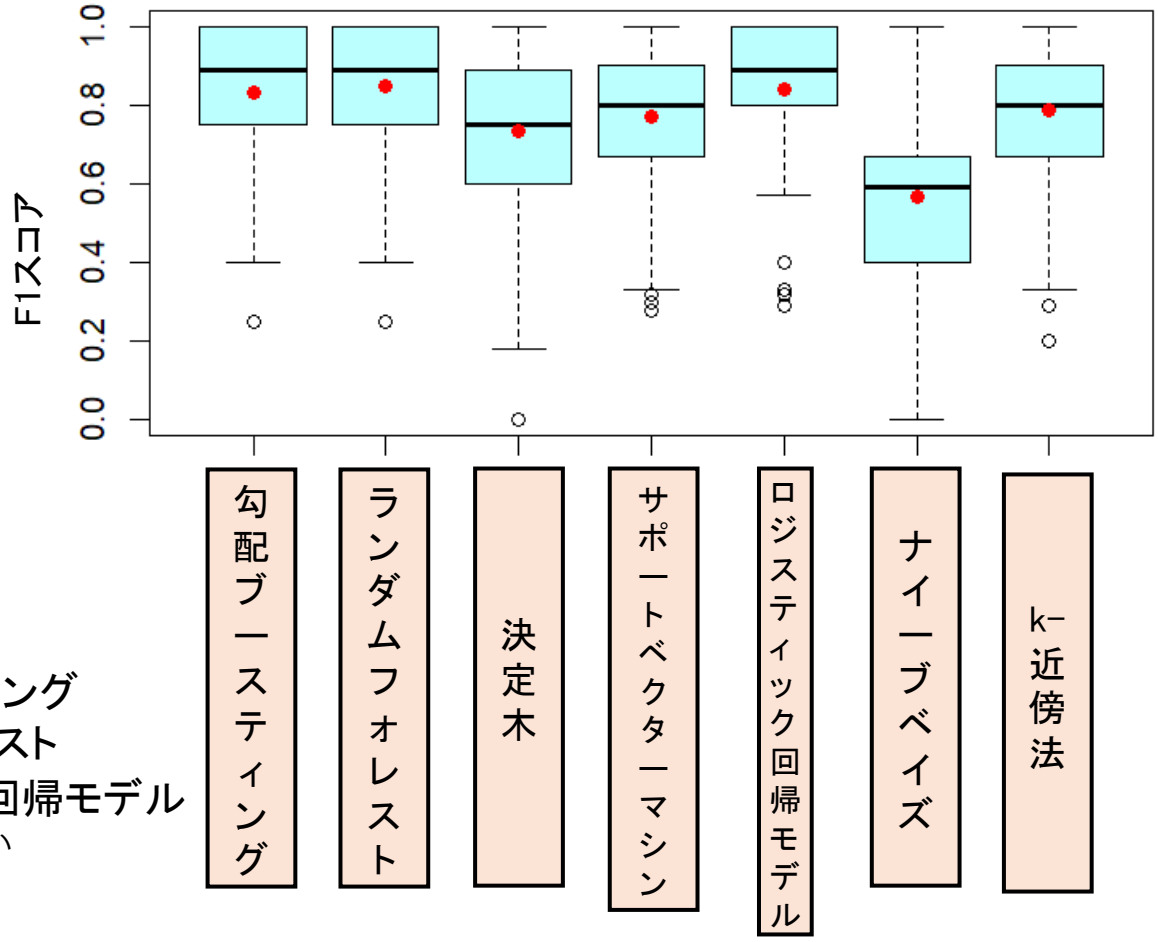
Word2Vec以外の4つが高い

観点2の考察

- 全体的にTF-IDF，BM25，CodeT5+，CodeBERTが有効
  - Word2Vecは次元数が100と他の手法と比べ小さい
  - **コメントスメル固有の表現**が得意な手法や，**事前に大量のソースコードを学習**した手法が有効

- TaskではTF-IDFが最も高い精度
  - Taskというコメントスメル固有の**アノテーションコメント**（TODO:，FIXME:)を適切にベクトルで表現

観点3の比較結果



- 勾配ブースティング
- ランダムフォレスト
- ロジスティック回帰モデル

が他と比べて高い

観点3の考察

- 全体的に**勾配ブースティング**，**ランダムフォレスト**，**ロジスティック回帰モデル**が有効

- **処理時間**では**ランダムフォレスト**，**ロジスティック回帰モデル**が有効

モデル	処理時間
勾配ブースティング	755.73秒
ランダムフォレスト	95.62秒
ロジスティック回帰モデル	62.73秒

まとめ

- **実験結果より**
  - コメントとソースコードを**ベクトル化し統合したもの**の方がすべてのスメルで精度が向上
  - **ベクトル化手法**では**コメントスメル固有の記述内容**をベクトルで表現できる手法が有効
  - **機械学習モデル**では**ランダムフォレスト**が最も有望

- **今後の課題**
  - ソースコードとコメントに対するベクトル化手法を揃えるかたちではなく，別々のものにするとどうなるか