

略語とそれらの展開の意味が異なる語がソフトウェア
アーティファクトのどこで発生するのか

An Empirical Study of Abbreviations and Expansions in Software Artifacts

Christian D. Newman, Michael J. Decker, Reem S. Alsuhaibani,
Anthony Peruma, Dishant Kaushik, Emily Hill

発表者：作島大智

略語/展開

- 展開が明らかな場合
num → number
- 展開の可能性が複数ある場合
cfg →
configure
configuration
control flow graph
context-free grammar

略語の展開

- 一般的に前処理のひとつとして行われる
分割, ステミング, トークン化などの他のステップと一緒に実行される
- データ量を増やすことができる

略語の種類

- Hill ら [19]
- 略語の種類を分類した.

Table I. Syntax-Based Abbreviation Categorizations

Category	Abbreviation Type	Definition	Example
Single Word	<i>Prefix</i>	Abbreviation of a single word that is strictly a prefix of the full word; formed by dropping letters from the end of the full word	Pub → Public Attr → Attribute Abbrev → Abbreviation
	<i>Dropped Letter</i>	Abbreviation of a single word that is formed by dropping letters from anywhere within the full word except the first letter	Cfg → Configure Ln → Line Tty → Teletype
Multi Word	<i>Acronym</i>	Abbreviation made from the first letters of multiple words.	Kv → Key value Ip → Internet protocol Vr → Virtual reality
	<i>Combination Multi-word</i>	Abbreviation made by dropping letters from multiple words	Oid → Object Identifier StdDev → Standard Deviation Arg → access rights

背景

- 自然言語処理(NLP)は多くの研究で使われている. [2-8]
その研究に関する技術品質はNLPツールの品質に依存している.
- **識別子**にはソースコードの文字の70%を含む. [9]
→NLPの精度向上には識別子の判定を正確に行うこと大事
[10]
- つまり, **前処理の一部である略語の展開の精度を向上させる
ことで自然言語処理を使っている技術群の精度が向上する**

問題点

- 語彙はソフトウェアプロジェクト間で意味が異なる.
[13,15]
- 識別子名を使うにはすべてのプロジェクトで一般化されている前提が必要
→感情分析ツール[12]

先行研究

- 展開の精度はまだまだ(60%)
- 正規表現, 言語モデル, 文字列照合近似アルゴリズムなど多くの技術を使用している
- 各技術によって精度が高い略語と低い略語が存在する

例

成果物によって技術の精度が異なる

	技術A	技術B	技術C
成果物a	80%	70%	50%
成果物b	80%	50%	80%

RQ (1/2)

- RQ1 :

ソフトウェア成果物の略語/展開の特性は何か.

これらの特性は展開技術の精度にどの程度影響するか

RQ1-1:

すべてのアーティファクトで略語展開
の分布はどんなか

RQ1-3:

様々なタイプの略語の展開は他の
アーティファクトよりも頻繁に発生
するものがあるか

RQ1-2:

略語展開を行う単語は、通常互いに隣
接してるのか

RQ (2/2)

- RQ2 :

これらの特性に関するデータを使用して、将来の略語の展開に関する調査の実施方法と報告方法を改善できるか.

アプローチ (1/2)

- **過去研究から得られた略語と展開の特性**, **著者によって手動で導出された略語と展開のa gold set**を使用
- 様々なソフトウェア成果物で**典型的な略語と展開**を調査し, a ground truth*を作成

*正確さや整合性を確認するためのデータ

アプローチ (2/2)

- 過去研究での評価方法を調べる
 - 特性が品質にどのように影響を与えるのか
 - 特性が適切に説明され、報告されているか

目標

- 略語/展開の特性を一般化
- 展開手法と使用するソフトウェア成果物の評価方法の長所と短所を強調すること

研究を始める前に

プロジェクト収集

- srcML [22] を使って識別子とコメントのすべてを収集するため、記述言語がC++, Java, C#, Cのみに限定
- 略語を含んでいるプロジェクトを選択
- システムのサイズ/成熟度が展開の場所にどのように影響するのかを確認するために、中小規模のシステムに限定

前処理

- コンピュータサイエンス, 英語辞書を除くすべてのアーティファクトには必要
- すべての句読点, 特殊文字を削除
- 控えめにcamelCase、under_scores、numbersに分割
- 全ての文字を小文字に変換

プロジェクト収集

コメントの量/
1000行

プロジェクトドキュメント
の密度

Table II. System Statistics

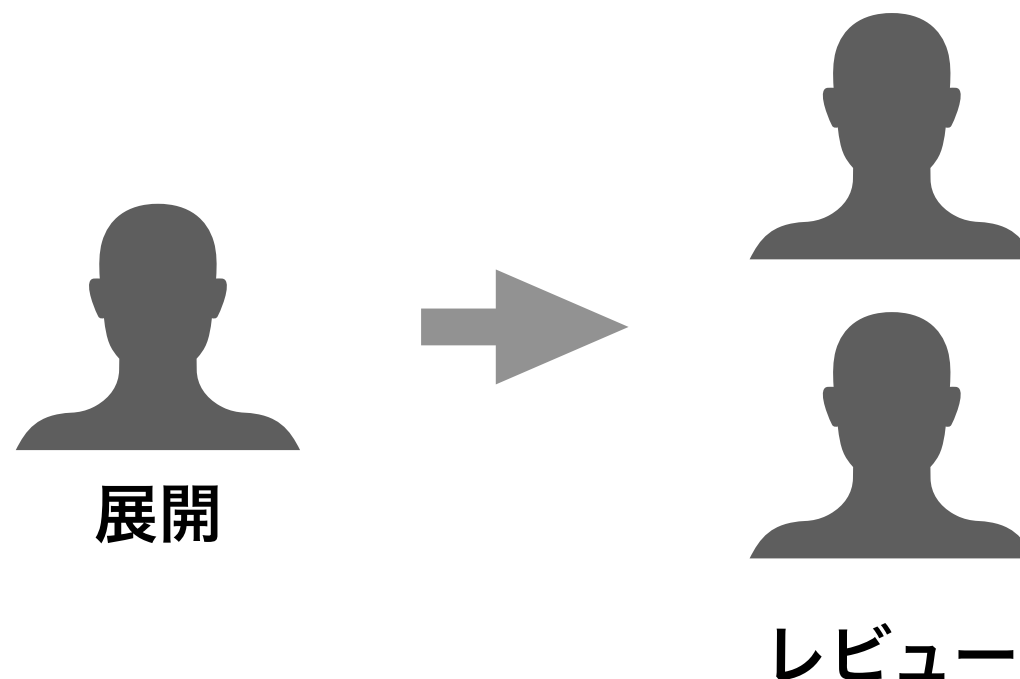
System Name	Primary Language(s)	Size (KLOC)	# Unique Abbreviations	# Unique Abbreviation-Expansions	Comment Density (in #comments per 1 KLOC)	Project Document Density (in #words per 1 KLOC)
Wycheproof	Java	9	126	156	134	439
Telegram	Java, C	781	143	164	14	4201
OpenOffice	C++, Java	4462	129	143	161	1410
Enscript	C	59	132	156	77	245
KDevelop	C++	259	189	242	88	7428
Sum/Total		5570	719	861	474	13723

一意の略語の数

一意の展開の数

データ作成

- 選択したファイルから略語を手作業で収集
- 3人でソースコードを個別にスキャンし、略語を含む識別子に関する情報を収集
- 手作業で展開し、残りの2人がレビュー



被りをスキップしたが
見落としている可能性
はある

RQ (再掲)

- RQ1 :

ソフトウェア成果物の略語/展開の特性は何か.
これらの特性は展開技術の精度にどの程度影響するか

RQ1-1:

すべてのアーティファクトで略語展開
の分布はどんなか

RQ1-3:

様々なタイプの略語の展開は他の
アーティファクトよりも頻繁に発生
するものがあるか

RQ1-2:

略語展開を行う単語は、通常互いに隣
接してるのか

RQ1-1

すべてのアーティファクトにわたる
略語展開の分布はどのようなものか

隣接

- 複数語の展開の単語は必ずしも互いに隣接しているわけではない
→ 略語とその展開を示す場所を記録する必要がある
- コードでは同じ行に候補単語があるのが隣接
同じ関数内というのは隣接ではない
- Wycheproofの識別子ptHex
→ the words 'plaintext' and 'hexadecimal'

展開の数の割合

- ソースコードに含まれる展開の分布

Table III. Total Number of Expansions Found per System in Source Code Identifiers.

	Type (declarations)	Type (parameters)	Name (declarations)	Name (parameters)	Name (expr)	Functions and Methods	Class Name/Field	Global
Enscript	53 (34%)	20 (12.8%)	63 (40.4%)	25 (16%)	69 (44.2%)	75 (48.1%)	49 (31.4%)	67 (42.9%)
KDevelop	125 (51.7%)	108 (44.6%)	161 (66.5%)	121 (50%)	144 (59.5%)	161 (66.5%)	120 (49.6%)	157 (64.9%)
Open Office	104 (72.7%)	100 (69.9%)	106 (74.1%)	98 (68.5%)	107 (74.8%)	108 (75.5%)	105 (73.4%)	105 (73.4%)
Telegram	99 (60.4%)	75 (45.7%)	112 (68.3%)	103 (62.8%)	112 (68.3%)	122 (74.4%)	111 (67.7%)	102 (62.2%)
Wycheproof	30 (19.2%)	16 (10.3%)	29 (18.6%)	12 (7.7%)	13 (8.3%)	39 (25%)	17 (10.9%)	0
Total	411	319	445	359	445	505	402	434
Mean	82.20	63.80	89.00	71.80	89.00	101.00	80.40	86.80
Median	99.00	75.00	107.00	98.00	107.00	108.00	105.00	102.00
StdDev	39.26	43.57	50.13	49.62	50.13	46.40	45.04	56.80

展開の候補が同じ場所に存在する

展開の候補が様々な場所に存在する

展開の数の割合

- ソースコード以外(RQ1-2で詳細をみます)

Table IV. Total Number of Expansions Found per System in Comments, project, language, CS, and English corpora

	Comments (anywhere)	Comments (adjacent)	Project (anywhere)	Project (adjacent)	Language (anywhere)	Language (adjacent)	CS Dict.	English Dict.
Enscript	35 (22.4%)	32 (20.5%)	77 (49.4%)	67 (42.9%)	141 (90.4%)	129 (82.7%)	55 (35.3%)	110 (70.5%)
KDevelop	104 (43%)	103 (42.6%)	225 (93%)	205 (84.7%)	222 (91.7%)	194 (80.2%)	94 (38.8%)	171 (70.7%)
Open Office	105 (73.4%)	101 (70.6%)	127 (88.8%)	124 (86.7%)	121 (84.6%)	113 (79%)	54 (37.8%)	98 (68.5%)
Telegram	108 (65.9%)	104 (63.4%)	126 (76.8%)	108 (65.9%)	149 (90.9%)	126 (76.8%)	55 (33.5%)	110 (67.1%)
Wycheproof	38 (24.4%)	37 (23.7%)	45 (28.8%)	39 (25%)	95 (60.9%)	69 (44.2%)	32 (20.5%)	51 (32.7%)
Total	390	377	600	543	728	631	290	540
Mean	78.00	75.40	120.00	108.60	145.60	126.20	58.00	108.00
Median	104.00	101.00	126.00	108.00	141.00	126.00	55.00	110.00
StdDev	37.93	37.39	68.16	63.45	47.53	44.86	22.39	42.80

興味深い結果

表2

System Name	Primary Language(s)	Comment Density (in #comments per 1 KLOC)
Wycheproof	Java	134
Telegram	Java, C	14
OpenOffice	C++, Java	161
Enscript	C	77
KDevelop	C++	88
Sum/Total		474

表4

	Comments (anywhere)	Comments (adjacent)
Enscript	35 (22.4%)	32 (20.5%)
KDevelop	104 (43%)	103 (42.6%)
Open Office	105 (73.4%)	101 (70.6%)
Telegram	108 (65.9%)	104 (63.4%)
Wycheproof	38 (24.4%)	37 (23.7%)
Total	390	377
Mean	78.00	75.40
Median	104.00	101.00
StdDev	37.93	37.39

- コメントの密度とプロジェクト文書で見つかった展開の数との間に明確な相関関係がない

略語/展開の分布

固有の拡張の分布

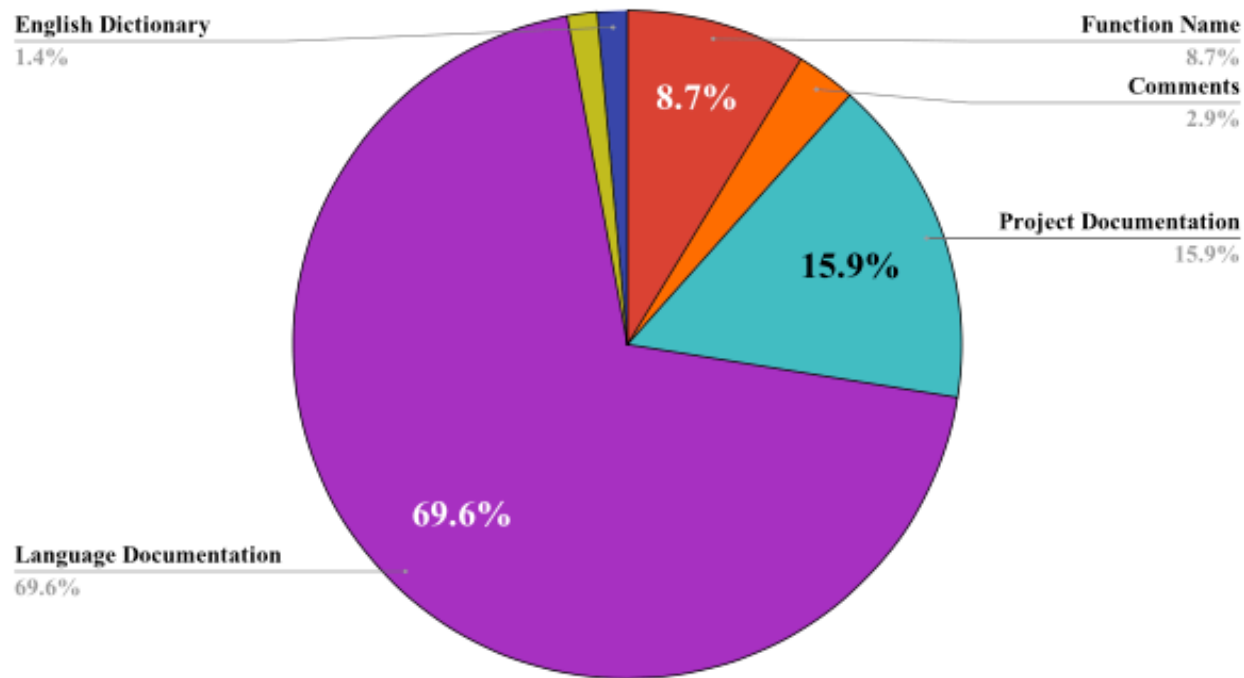


Figure 1. Where Do Abbreviation Expansions Uniquely Occur? (Total of 69)

一意に展開できない略語の分布

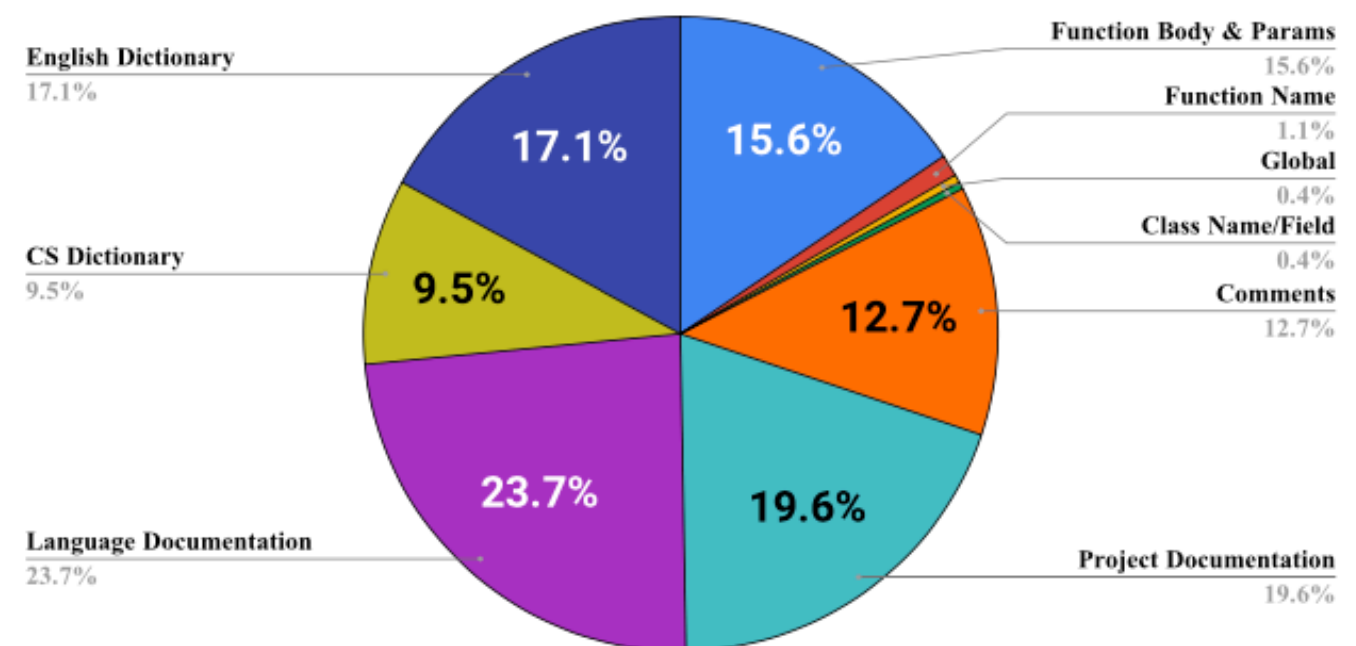


Figure 2. Where Do Abbreviation Expansions Occur? (Total of 3067)

RQ1-2

略語展開を行う単語は，通常互いに隣接してるのか

隣接と非隣接の比較

プロジェクト・言語ドキュメントで多く隣接が存在
コメントでは隣接関係は影響が顕著ではない

Table IV. Total Number of Expansions Found per System in Comments, project, language, CS, and English corpora

	Comments (anywhere)	Comments (adjacent)	Project (anywhere)	Project (adjacent)	Language (anywhere)	Language (adjacent)	CS Dict.	English Dict.
Enscript	35 (22.4%)	32 (20.5%)	77 (49.4%)	67 (42.9%)	141 (90.4%)	129 (82.7%)	55 (35.3%)	110 (70.5%)
KDevelop	104 (43%)	103 (42.6%)	225 (93%)	205 (84.7%)	222 (91.7%)	194 (80.2%)	94 (38.8%)	171 (70.7%)
Open Office	105 (73.4%)	101 (70.6%)	127 (88.8%)	124 (86.7%)	121 (84.6%)	113 (79%)	54 (37.8%)	98 (68.5%)
Telegram	108 (65.9%)	104 (63.4%)	126 (76.8%)	108 (65.9%)	149 (90.9%)	126 (76.8%)	55 (33.5%)	110 (67.1%)
Wycheproof	38 (24.4%)	37 (23.7%)	45 (28.8%)	39 (25%)	95 (60.9%)	69 (44.2%)	32 (20.5%)	51 (32.7%)
Total	390	377	600	543	728	631	290	540
Mean	78.00	75.40	120.00	108.60	145.60	126.20	58.00	108.00
Median	104.00	101.00	126.00	108.00	141.00	126.00	55.00	110.00
StdDev	37.93	37.39	68.16	63.45	47.53	44.86	22.39	42.80

18個(~10%)

26個(~16%)

隣接していない複数語の展開

Table V. Number of Non-Adjacent Multi-Word Expansions

	Type (params)	Type (decls)	Name (decls)	Name (expr)	Name (params)	Total
Enscript	0	3	4	3	1	11
KDevelop	5	5	8	7	7	32
Open Office	9	6	5	6	6	32
Telegram	8	9	9	8	7	41
Wycheproof	6	6	11	2	7	32
						計148

- すべての略語展開の約19%は複数語で隣接していない
(ソースコードのみに着目すると148/798)

RQ1-2：結果

- **言語・プロジェクトドキュメント複数語の10-16%**
ソースコードにおける複数語の19%が非隣接
- ツールの有効性が効果的でない場合、隣接・非隣接を考慮することの有効性を理解することが重要

RQ1-3

さまざまなタイプの略語の展開が他のものよりも頻繁に
発生するアーティファクトがあるか

略語の種類(再掲)

- Hill ら [19]
- 略語の種類を分類した.

Table I. Syntax-Based Abbreviation Categorizations

Category	Abbreviation Type	Definition	Example
Single Word	<i>Prefix</i>	Abbreviation of a single word that is strictly a prefix of the full word; formed by dropping letters from the end of the full word	Pub → Public Attr → Attribute Abbrev → Abbreviation
	<i>Dropped Letter</i>	Abbreviation of a single word that is formed by dropping letters from anywhere within the full word except the first letter	Cfg → Configure Ln → Line Tty → Teletype
Multi Word	<i>Acronym</i>	Abbreviation made from the first letters of multiple words.	Kv → Key value Ip → Internet protocol Vr → Virtual reality
	<i>Combination Multi-word</i>	Abbreviation made by dropping letters from multiple words	Oid → Object Identifier StdDev → Standard Deviation Arg → access rights

略語の種類分布

Table VI. Total Number of Abbreviations per Category

	Acronym	Dropped	Combo.	Prefix	Total
Wycheproof	41 (38.3%)	22 (20.6%)	3 (2.8%)	41 (38.3%)	107
Open Office	22 (16.7%)	34 (25.8%)	4 (3%)	72 (54.5%)	132
KDevelop	33 (14%)	60 (25.5%)	4 (1.7%)	138(58.7%)	235
Telegram	38 (23.9%)	33 (20.8%)	0 (0%)	88 (55.3%)	159
Enscript	23 (15.2%)	46 (30.5%)	1 (0.7%)	81 (53.6%)	151

分類結果

ソースコード

Table VII. Frequency at which Different Types of Abbreviations Occur in Different Source Code Locations

	Type (declarations)	Type (params)	Name (declarations)	Name (params)	Name (expr)	Functions and Methods	Class Name/Field	Global
Prefix	273 (65.5%)	219 (52.5%)	314 (75.3%)	244 (58.5%)	305 (73.1%)	335 (80.3%)	283 (67.9%)	303(72.7%)
Dropped	92 (51.1%)	70 (38.9%)	104 (57.8%)	82 (45.6%)	99 (55%)	111 (61.7%)	84 (46.7%)	94 (52.2%)
Acronym	46 (29.3%)	30 (19.1%)	53 (33.8%)	33 (21%)	41 (26.1%)	59 (37.6%)	35 (22.3%)	37 (23.6%)
Combo Multi-word	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)

ドキュメント

Table VIII. Frequency at which Different Types of Abbreviations Occur in Different Software Artifacts

	Comments (anywhere)	Project (anywhere)	Language (anywhere)	CS Dict.	English Dict.
Prefix	275 (65.9%)	351 (84.2%)	411 (98.6%)	193 (46.3%)	407 (97.6%)
Dropped	80 (44.4%)	134 (74.4%)	162 (90%)	61 (33.9%)	116 (64.4%)
Acronym	34 (21.7%)	106 (67.5%)	147 (93.6%)	34 (21.7%)	0 (0%)
Combo Multi-word	1 (8.3%)	9 (75%)	8 (66.7%)	1 (8.3%)	0 (0%)

RQ1-3：結果

- 他のものよりも多くの種類の略語を保有しているアーティファクトはある
- ドキュメントでは候補の略語展開を照合，フィルタリングするためのより多彩な方法が必要
- 言語ドキュメントではソースコードよりも頭文字を重視する必要がある

RQ (再掲)

- RQ2 :

これらの特性に関するデータを使用して、将来の略語の展開に関する調査の実施方法と報告方法を改善できるか。

過去論文の分析

- どのアーティファクトが検索されるのか
- 異なるアーティファクトの複数語への展開の際に、隣接非隣接について報告しているかどうか
- 各略語タイプで報告されている有効性

先行研究

- コード外のアーティファクトに重要な展開が含まれている
という考えは新しいものではない
→どの技術がどのソフトウェアアーティファクトに対して
どのくらい効果的かを知ることが重要
- テキスト内で隣接に注目している過去論文はない

先行研究の比較

Table XIII. Software Artifacts used in Techniques in Literature Review

	[30]	[21]	[18]	[28]	[20]	[19]	[17]	[29]	[31]
Source code identifiers	✓	✓	✓	✓	✓	✓	✓	✓	✓
Comments			✓	✓		✓	✓	✓	✓
Project Documentation			✓						✓
Language Documentation	✓							✓	
Computer Science Dictionary/training data	✓	✓					✓	✓	✓
English Dictionary/training data	✓		✓	✓	✓		✓	✓	✓

全体の精度のみ

Table IX. Techniques that Report Overall Accuracy. ** = Technique reports combined split/expansion accuracy

Original Publication	Maximum Reported Overall Accuracy
Lawrie 2011[18]	66%
Lawrie 2007 [28]	64%
Alatawi [20]	78.%
Tidier [29]	48%
**Tris [30]	86%

略語タイプごとの精度

Table X. Techniques that Report Accuracy per Abbreviation Type

	Prefix	Dropped	Acronym	Combo	Overall Accuracy
AMAP[19]	79.7%	77%	46.9%	9.2%	63%
LINSEN[17]	86%	77.8%	36%	66.7%	62%

precision/recallを使用する手法

Table XI. Techniques that Report Overall Precision and Recall. ** = Technique reports combined split/expansion accuracy, N/A = Not reported

	Average Precision	Average Recall	F-Measure	Overall Accuracy
**Tris[30]	95%	91%	92%	86%
Jiang[21]	95%	65%	N/A	N/A
Lingua::IdSplitter [31]	86%	86%	89%	83%

Table XII. Techniques that Report Precision per Abbreviation Type (did not report recall per abbrev type)

	Prefix Precision	Dropped Precision	Acronym Precision
Jiang [21] Param Name	97%	79%	96%
Jiang [21] Param Type	100%	N/A	98%

どの手法を使えばいいのか

- 各手法を比較することが難しい
→一貫性のある包括的なメトリクスのセットを提唱
- Precision, Recall, F1, 各略語タイプの精度
- Precision, Recall, F1, ソフトウェア成果物の種類,
各成果物での展開の精度
- Precision, Recall, F1, 非隣接の精度

結果

- とりあげたメトリクスのセットを報告することを推奨
- どのアーティファクトに注目しているのかより細かい粒度で報告するとよい

まとめ

- ソースコードの展開の19%が複数語で非隣接であるのに対し、非ソースコードアーティファクトは10-16%は非隣接である
→隣接関係を考慮すると精度が向上すると考えられる
- ドキュメントによって略語展開タイプの割合が異なるため略語展開タイプの粒度で有効である手法の必要
→細かい粒度の議論が必要

所感

- まず最初に表の説明をしてほしい
表が多すぎて理解が難しい
- RQ2の回答があまりよくわからなかった

補足資料

略語

- 略語の展開は開発者にとって重要
→短い識別子を理解することは困難[16]
識別子に略語や文字ではなく単語にのみが含まれている場合、理解速度が19%向上する

略語

- ソースコードの意味を自然言語(英語)などで伝達
- ソフトウェアの役割や動作を示す単語を使用して識別子, コメント, ドキュメントを構築する
→略語の自由度は開発者次第

略語の展開

- 先行研究[17-20]
- 識別子を検出
- 周囲のコード，コメント，ドキュメント，辞書を探す
- ヒットした単語 or 語句に展開

略語/展開の隣接

- 言語ドキュメント，プロジェクトドキュメント，コメント，ソースコードに関係
- 異なるシステム/ソフトウェア成果物でのツールの有効性が効果的でない場合、ツールの有効性が低下するため、隣接および非隣接拡張のツールの有効性を理解することが重要

展開をカウント

- 手動で収集された略語/展開を取得し，1つ以上のソフトウェア成果物に一致させること
- 略語/展開に対応するプロジェクトのコード全体をスキャンし，展開と一致する場所と略語が表示されている場所を記録(メソッド名や型名の一部など)