An Empirical Study of Abbreviations and Expansions in Software Artifacts

SCANL
Source Code Analysis And
Natural Language Laboratory

Hello, we are SCANL Lab!

We study the latent connection between source code behavior and the natural language elements used to describe that behavior

Members of the lab on this paper:

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

For more information on our tools, datasets, and research:

https://scanl.org/

Identifier Normalization

- Stemming/lemmatization
- **Expanding abbreviations and non-dictionary terms**
- Splitting
- Naming conventions

- To make it easier for people and algorithms to understand the natural language text being analyzed

Non-dictionary terms

- Non-dictionary terms are sequences of characters that can be expanded to a larger word or phrase.
  - Abbreviations are non-dictionary terms which contain letters from the phrase or word they expand to
- Make comprehension more difficult
  - For algorithms which analyze natural language artifacts
  - For humans who may not be familiar with the expansion
- Are a shorter way of conveying information
  - For those familiar with the expansion
- Appear in MANY software artifacts

(Shortened) Example

```
ECPublicKey getPublicKey() throws InvalidKeySpecException
{
      KeyFactory kf = KeyFactory.getInstance("EC");
      byte[] encoded = TestUtil.hexToBytes(pub);
      return (ECPublicKey) kf.generatePublic(new
              X509EncodedKeySpec(encoded));
}
```

Types of Abbreviations

Abbreviation Type	Definition	Example
Single Word: Prefix	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
Single Word: Dropped Letter	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: Acronym	Abbreviation made from the first letters of multiple words.	Kv → Key value Ip → Internet protocol Cfg → Control flow graph
Multi-word: Combination	Abbreviation made by dropping letters from multiple words	Oid → Object Identifier StdDev→Standard Deviation Arg → access rights

Lots of approaches

1. A. Corazza, S. Di Martino, and V. Maggio, “LINSEN: An efficient approach to split identifiers and expand abbreviations,” in Software Maintenance (ICSM), 2012 28th IEEE International Conference on, 2012, pp. 233–242.
2. D. Lawrie and D. Binkley, “Expanding identifiers to normalize source code vocabulary,” in Software Maintenance (ICSM), 2011 27th IEEE International Conference on, 2011, pp. 113–122.
3. E. Hill et al., “AMAP: automatically mining abbreviation expansions in programs to enhance software maintenance tools,” in Proceedings of the 2008 international working conference on Mining software repositories, 2008, pp. 79–88.
4. A. Alatawi, W. Xu, and J. Yan, “The Expansion of Source Code Abbreviations Using a Language Model,” in 2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC), 2018, vol. 02, pp. 370–375.
5. Y. Jiang, H. Liu, J. Qi Zhu, and L. Zhang, “Automatic and Accurate Expansion of Abbreviations in Parameters,” IEEE Trans. Softw. Eng., vol. PP, pp. 1–1, 2018.
6. D. Lawrie, H. Feild, and D. Binkley, “Extracting meaning from abbreviated identifiers,” in Source Code Analysis and Manipulation, 2007. SCAM 2007. Seventh IEEE International Working Conference on, 2007, pp. 213–222.
7. L. Guerrouj, M. Di Penta, G. Antoniol, and Y.-G. Guéhéneuc, “Tidier: an identifier splitting approach using speech recognition techniques,” J. Softw. Evol. Process, vol. 25, no. 6, pp. 575–599, 2013.
8. L. Guerrouj, P. Galinier, Y.-G. Guéhéneuc, G. Antoniol, and M. Di Penta, “Tris: A fast and accurate identifiers splitting and expansion algorithm,” in Reverse Engineering (WCRE), 2012 19th Working Conference on, 2012, pp. 103–112.
9. N. R. Carvalho, J. J. Almeida, P. R. Henriques, and M. J. Varanda, “From source code identifiers to natural language terms,” J. Syst. Softw., vol. 100, pp. 117–128, 2015.

Characteristics

- Abbreviations and expansions may appear in many different software artifacts
  - Source code, project docs, language docs, requirements docs
- There are different types of abbreviations
  - Prefix, dropped, acronyms, combination multi-word
  - Each type may appear in different artifacts more or less often
- Words that make up expansions may not appear directly next to one another in text
  - And the rate of this may differ between different software artifacts (e.g., source code vs. C++ language documentation)

Adjacency Example

- SpecRef

“This is a reference to specification number 5.0.1”

Methodologically Diverse

- Regular Expressions
- Machine learning or statistical approaches (e.g., token frequency)
- Static analysis
- Artificial Intelligence

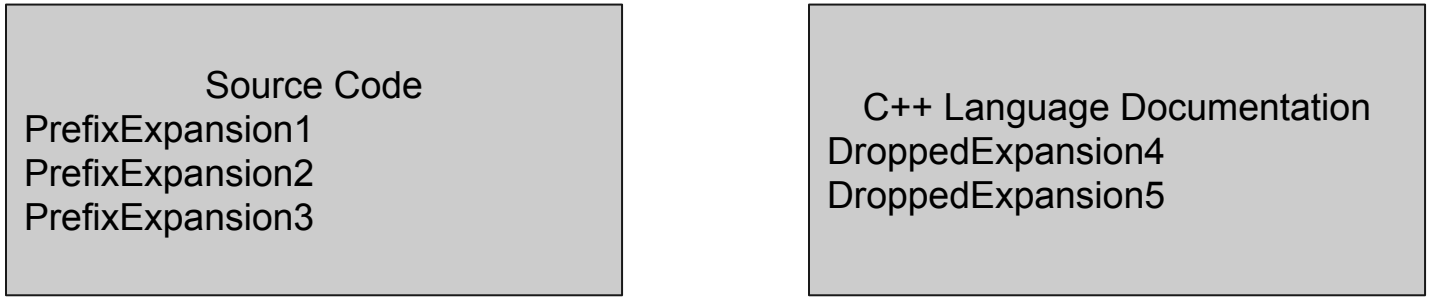
- Rely on various splitting, stemming/lemmatization and various string matching approaches (e.g., edit distance)

The Problem

- Prior to this work, these characteristics had never been formally declared as a full set; but would individually appear in different evaluations
- Methodologically diverse techniques are hard to compare with one another if they do not evaluate using the full set
  - Lack of implementation, evaluation uses only some characteristics **but not all.**

Example

- Expansion technique 1 finds expansions 1-3
- Expansion technique 2 finds expansions 4 and 5
  - If we don’t take into account the types (prefix,dropped) or the fact that they appear in different documents (code, docs) then technique 1 is best. In reality, though, they are *complementary.*



The Goal

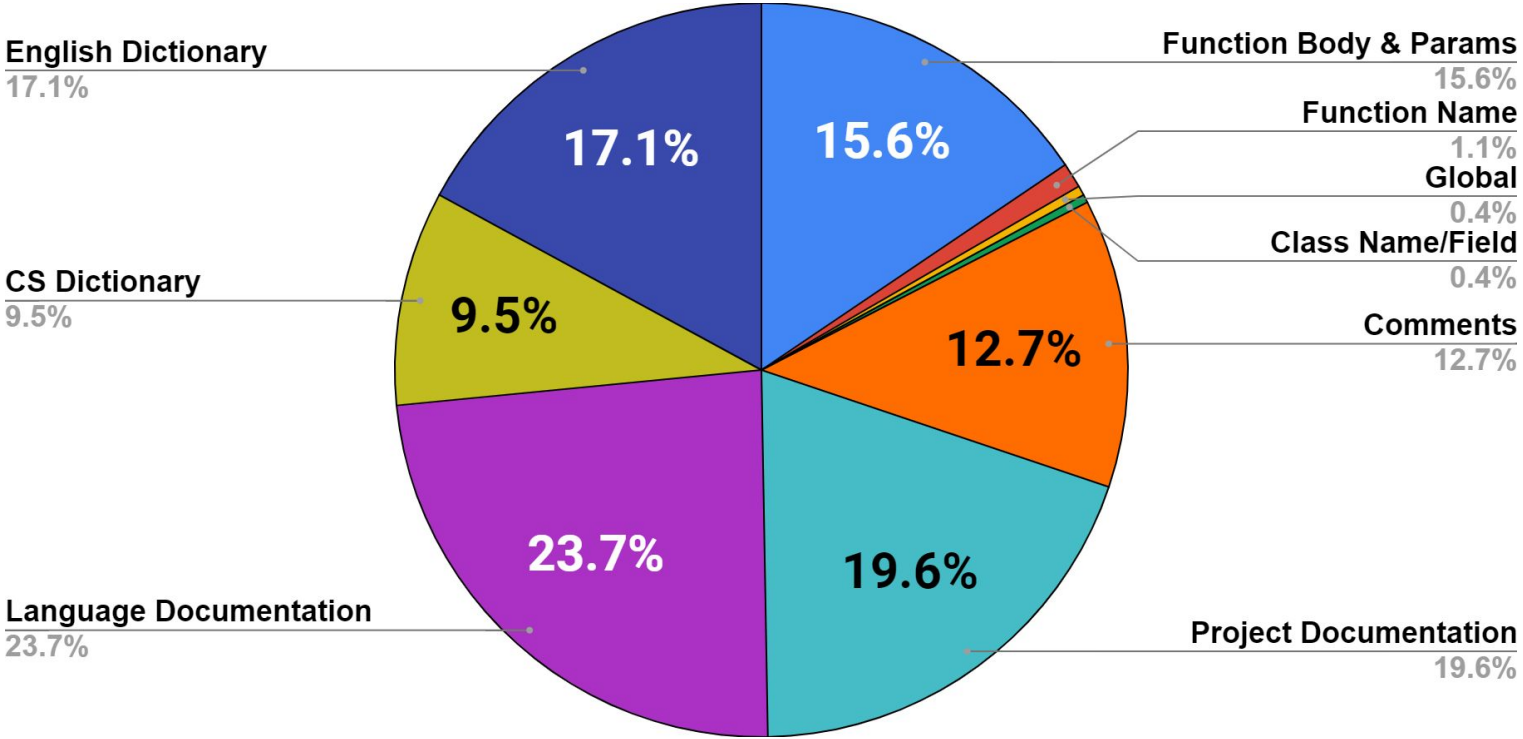
- Identify characteristics of abbreviations and expansions in different types of software artifacts
  - Examine how these characteristics appear in the wild
- Examine how prior techniques have been evaluated to understand how missing different characteristics in the evaluation makes them difficult to compare
- Highlight characteristics we find, recommend evaluation metrics for future research
  - Help clarify how different expansion techniques may synergize or where their strengths/weaknesses lie in comparison to one another

The Solution

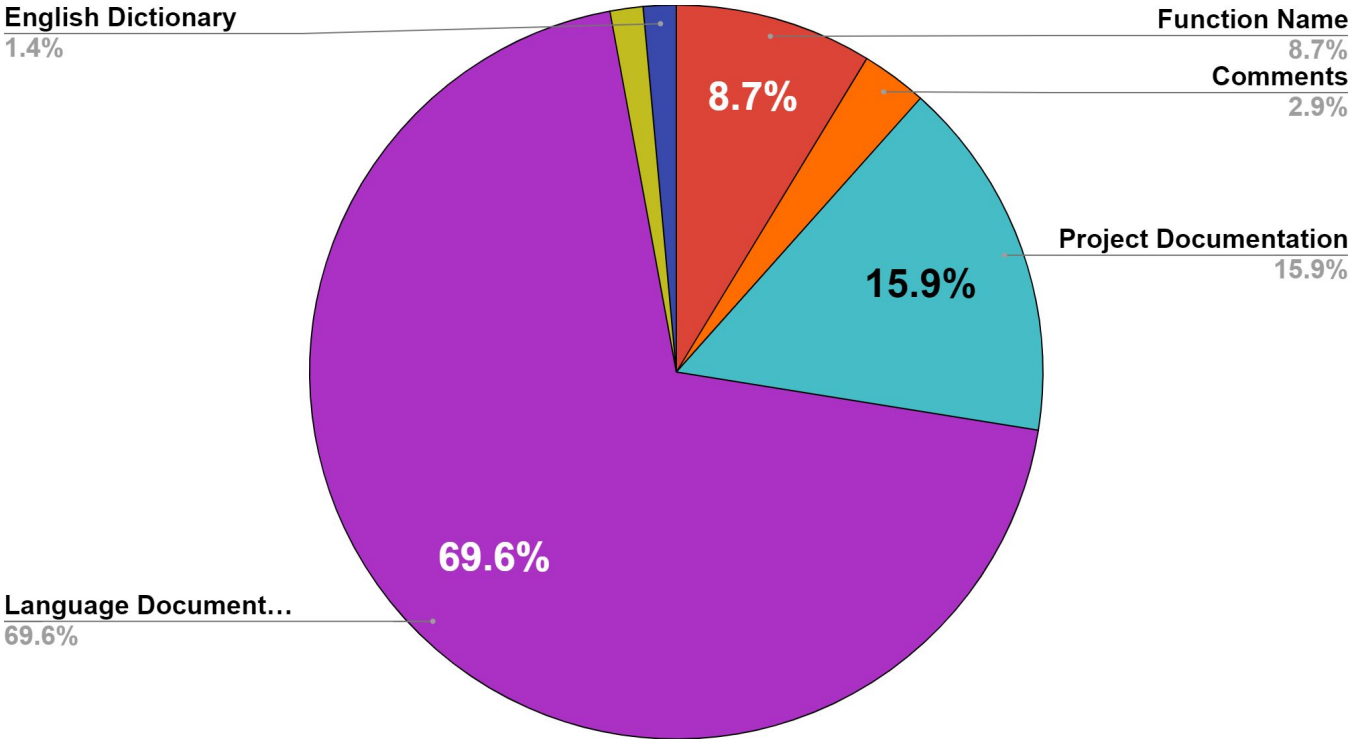
- Manually curated data set of 861 abbreviation-expansion pairs
- Find abbreviations in software artifacts including:
  - The source code, comments, system documentation, language documentation, a computer science dictionary, and an English dictionary
- Report on characteristics of abbreviations and expansions in each artifact
- Use characteristics to identify paths toward improvement

What do abbreviations and expansions look like in the wild?

Location (globally, 3067)



Location (uniquely, tot. 69)



Word Adjacency (per sys)

	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

Abbreviation type (per sys)

	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

Abbreviation type (per loc)

	Type (decls)	Type (params)	Name (decls)	Name (params)	Name (expr)	Functions	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%)

Abbreviation type (per loc)

	Comments	Project	Language	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%)

Artifacts in other research

	[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	✓		✓	✓	✓		✓	✓	✓

Takeaways

- Abbreviations and expansions are not the same between different artifacts
- Different types of abbreviations are harder to find
- Not all papers report effectiveness on different types of abbreviations
- Few papers report their accuracy on different types of artifacts
  - Difficult to understand effectiveness on different NL documents, which do have different distributions wrt abbreviation/expansions

Takeaways Cont’d

- Techniques reporting the most accuracy use smallest number of artifacts outside of the code
  - Hard to tell if this is due to their dataset, since some don’t report effectiveness on different types of identifiers
- No technique evaluates effectiveness on non-adjacent terms
  - Show up in certain types of artifacts more than others and make up ~19% (total of 148) of all expansions in the study

Conclusion

- Abbreviations and expansions have several characteristics which must be considered as a set when trying to understand what they look like “in the wild”
- We need to evaluate using all of these characteristics instead of just taking a few at a time if we want to have a more complete view of the effectiveness of our tools
    - Precision, recall, accuracy for all software artifacts and different types of abbreviations

SCANL
Source Code Analysis And
Natural Language Laboratory

We welcome additions to the dataset!

https://scanl.org/

SCANL

Source Code Analysis And
Natural Language Laboratory