Annotation Guideline for Code and Named Entity Recognition in StackOverflow

Jeniya Tabassum, Mounica Maddela, Wei Xu, Alan Ritter
Department of Computer Science and Engineering
The Ohio State University
{tabassum.13, maddela.4, xu.1265, ritter.1492}@osu.edu

1 Definition of Software Entities

In this section, we define 28 different entities, that we found in the StackOveflow corpus.

1.1 Application

Mention of software names (which is not referred as code library) should be annotated with APPLICATION. For example: Mosh, JKplayer, api-java-client, java SDK, etc.

1.2 Language

Any mention of programming language name should be annotated with LANGUAGE. For example: Python, java, CSS, cpp etc.

1.3 Version

The numeric value or string referring to the versions of programming language or software or any other entity should be annotated with VERSION. For example: in the string 'Python 2.7', 'Python' should be annotated with LANGUAGE and '2.7' with VERSION. Similarly in 'Windows XP', 'XP' is the version.

1.4 Algorithm

Any mention of algorithm or protocols should be annotated with ALGORITHM. For example: UDP (User Datagram Protocol), DFS, RBM, etc.

1.5 Operating System

All the mentions of OS names should be annotated with OS. For example: linux, iOS, Windows etc.

1.6 Device

Any basic hardware requirement for software development system should be annotated with DEVICE. For example: phone, mobile, GPU, etc.

1.7 Error-Name

The text that contains the name of an error should be annotated with ERROR NAME. For example: Overflow, OutofRange, Invalid Indices, etc.

1.8 User Name

If the text contains any mention of a username then it should be annotated with USER NAME. For example: John, Maya, Clark, etc.

1.9 Data Structure

The mention of any data structure element should be annotated with DATA STRUCTURE. For example: array, linked list, hash table, heap, etc.

1.10 Data Type

The text that contains the type description of variables should be annotated with DATA TYPE. For example: string, char, double, etc.

1.11 Library

The name of the code frameworks or libraries should be annotated with LIBRARY. We can think it as the string we include in the code with with $\#include/import$ statements. We are also including name of frameworks and toolkits inside libraries tags. For example: Numpy, Scipy. Autograd API, toolkit, framework should be annotated as library

1.12 Library Class

The class names defined under a Library LIBRARY CLASS. For example: ItemTemplate, PersistentGenericSet, actionManager, etc.

1.13 User Class

The class defined by programmers should be annotated with CLASS. C Struct type should also be annotated with CLASS, as it can be treated as a

class without function. For example: my_Class, Test_Class, etc.

1.14 Library Variable

The variable defined under a library should be annotated with LIBRARY VARIABLE. For example: math.inf, swing.color, ActonListener.Value, etc. However, count should not be considered as value. Also if there are a list of values, then mark the full span.

1.15 User Variable

The name of the variables/class/objects should be annotated with USER VARIABLE. For example: user_id, numberOfRowsInSection, etc.

1.16 Library Function

The common code library functions should be annotated with LIBRARY FUNCTION. For example: numpy.isinf(), Math.floor(), etc.

1.17 User Function Name

The name of the user defined functions should be annotated with USER FUNCTION. For example: any user defined function name : hello(), my_function(), etc.

1.18 File Type

If there is any mention of any file format name, then it should be annotated with FILE TYPE. For example: json, jar, etc.

1.19 File Name

If there is any mention of any file name, then it should be annotated with FILE NAME. For example: WindowsStoreProxy.xml, myaccess.htaccess etc. We should also annotate the file name with complete file path (if given), e.g., the string 'myfolder/myfile.txt' should be annotated with FILE NAME.

1.20 UI Element

The name of any user interface element should be annotated with UI ELEMENT. For example: image, button, scroll bar, text box etc.

1.21 Website

If there is any mention of any website name then it should be annotated with WEBSITE. Note that we only need to annotate the name of the websites. We do not need to annotate links as the website. For example: MSDN, Google, Yahoo etc.

1.22 Organization

Any mention of an organization name should be annotated with ORGANIZATION. For example: Apache, Microsoft Research, Fair, etc.

1.23 Licence

Any mention of a licence name should be annotated as LICENCE. For example: Creative Commons Attribution [Lincence] 4.0[version] International License.

1.24 HTML XML Tag Name

Any mention of a html tags/ user created xml tag (and also the names like div class names) should be annotated with HTML XML TAG NAME. For example: ⟨h1⟩, ⟨div⟩, ⟨img⟩, etc.

1.25 Value

The tokens which contains value of a variable or output of a function should be annotated as VALUE. For example: "hello world", '255.255.255.0', 30.5, True, etc.

1.26 In Line Code

Any line of code should be annotated as IN LINE CODE. For example: linux commands like: "grep -rnw", sql coammnads like "select * from Table Car", etc.

1.27 Output Block

Output of any system should be annotated as Output Block. For example: Output from console/any IDE

1.28 Keyboard Input

The name of any input keys should be annotated as KEYBOARD INPUT. For example: CTRL+X, ALT, fn, etc.

2 Basic Guideline

Each entity should be annotated in a way that it contains the maximum span length. For example:

- '*SGML parser*' instead of '*SGML*'
- '*Chrome browser*' instead of '*Chrome*'

While annotating LIBRARY, LIBRARY-CLASS, LIBRARY-FUNCTION, LIBRARY-VARIABLE, APPLICATION, LANGUAGE entities, we should Look for coresponding GitHub repositories to ensure the the correct entity tagging. In any such GitHub repository exists, we should add the link of the repository as note to the entity.

Entity	Example	Count	avg. #word	avg. #char
CLASS	*ItemTemplate, PersistentGenericSet, actionManager*	2202	1.09 ± 0.32	10.08 ± 5.43
APPLICATION	*Visual Studio, Weka, Homebrew, FFmpeg, Sonar Scanner*	2019	1.23 ± 0.32	10.08 ± 5.43
VARIABLE	*key, value, pt, A*	1607	1.07 ± 0.28	7.58 ± 5.79
UI ELEMENT	*textbox, wizard, button*	1536	1.13 ± 0.38	6.44 ± 2.82
INLINE CODE	*-type d, alias.⟨alias⟩, adoqry.active := true*	1358	2.65 ± 2.74	15.40 ± 18.26
FUNCTION	*.toLowerCase(), GmailApp.search(), search*	1281	1.12 ± 0.62	11.07 ± 8.09
LANGUAGE	*python, java, c++*	1060	1.02 ± 0.15	4.55 ± 2.65
LIBRARY	*Pyspark, aws-java-sdk-core, OAuth*	1186	1.17 ± 0.48	7.10 ± 3.62
VALUE	*"hello world", '255.255.255.0', 30.5, True*	1188	1.64 ± 1.39	5.6 ±6.03
DATA STRUCTURE	*tree, stack, linkedlist*	987	1.02 ± 0.14	5.26 ± 1.66
DATA TYPE	*string, integer, boolean, pointer*	634	1.05 ± 0.26	6.44 ± 1.97
FILE TYPE	*exe, dll, rich text*	556	1.01 ± 0.09	3.95 ± 1.66
FILE PATH	*Test.pdb, facebook_consumer.js, config/modules.config.php*	600	1.16 ± 0.67	13.35 ± 9.08
VERSION	*2003 R2, 4.2.6-200.fc22.x86_64, XP*	500	1.08 ± 0.39	3.76 ± 2.81
HTML XML TAG	*⟨div⟩, span, br*	277	1.07 ± 0.27	5.68 ± 3.81
DEVICE	*iPhone, Arduino MCU, 2d barcode scanner, nios cpu*	361	1.19 ± 0.46	6.09 ± 3.25
OPERATING SYSTEM	*Linux, iOS, Android*	303	1.10 ± 0.35	5.87 ± 2.14
WEBSITE	*stackexchange, GitHub, Codecourse, railstutorial.org*	182	1.10 ± 0.37	7.65 ± 3.60
USER NAME	*Chris M, @KamranFarzami, M.Deinum*	139	1.29 ± 0.53	7.63 ± 3.58
ALGORITHM	*A*-search, bubblesort, divide-and-conquer*	60	1.48 ± 0.59	8.45 ± 5.34

Table 1: Software-specific entity categories in the StackOverflow annotated corpus.

3 Automated Annotation

Using the meta-data of StackOverflow corpus, we provide the following automated annotations:

- Any part of text that appeared inside the ⟨code⟩-markdown tag is automatically annotated with IN LINE CODE.
- Any part of text that appeared inside the ⟨kbd⟩ markdown tag is automatically annotated with KEYBOARD INPUT.
- Any part of text that appeared inside the ⟨blockquote⟩ markdown tag is automatically annotated with OUTPUT BLOCK

However, these automated annotations, extracted from StackOverflow user provided tags, are not always correct. These automated tags were corrected by adding the annotation for the missing words and by removing the annotations from the incorrectly tagged words.

4 Annotated NER corpus

While training the NER tagger, we found that, some entities are quite infrequent. For example:

- OUTPUT BLOCK appeared 56 times in the annotated corpus
- ERROR NAME appeared 43 times in the annotated corpus
- ORGANIZATION appeared 28 times in the annotated corpus
- KEYBOARD INPUT appeared 32 times in the annotated corpus
- LICENCE appeared 10 times in the annotated corpus

While training and testing the model, we removed annotations for these five low frequency entities.

We also merged similar entities like below:

- LIBRARY CLASS and USER CLASS as CLASS
- LIBRARY FUNCTION and USER FUNCTION as FUNCTION
- LIBRARY VARIABLE and USER VARIABLE as VARIABLE

Table 1 shows the full corpus statistics after this merging removal.