Analyzing Software using Deep Learning

Token Vocabulary and Code Embeddings (Part 3)

Prof. Dr. Michael Pradel
Software Lab, University of Stuttgart
Summer 2020
Overview

- Token Vocabulary problem
- Pre-trained token embeddings
- Joint embedding space for NL & PL

Recommended papers:
- "Distributed representations of words and phrases and their compositionality", NIPS, 2013
- "Big Code != Big Vocabulary - Open-Vocabulary Models for Source Code", ICSE, 2020
- "Deep Code Search", ICSE, 2018
NL & PL Information

- Software is not just code
- Many natural language artifacts
- Applications of reasoning about both PL and NL information
  - NL-to-code search
  - Predict or check comments
  - Learn from API documentation
Joint Embedding Space

- How to reason about PL tokens and NL words together?
- Idea: Learn embedding that maps both PL tokens and NL words into a single vector space
  - Goal: Related tokens and words are close-by
  - Model learns how to related PL and NL information to each other
Deep Code Search

NL query → Learning-based Code Search Engine → Code snippet
Deep Code Search

NL query

“read an object from an xml”

Learning-based Code Search Engine

Code snippet
Deep Code Search

NL query

“read an object from an xml”

Learning-based Code Search Engine

Code snippet

```java
public static < S > S deserialize(Class c, File xml) {
    try {
        JAXBContext context = JAXBContext.newInstance(c);
        Unmarshaller unmarshaller = context.createUnmarshaller();
        S deserialized = (S) unmarshaller.unmarshal(xml);
        return deserialized;
    } catch (JAXBException ex) {
        log.error("Error-deserializing-object-from-XML", ex);
        return null;
    }
}
```
Deep Code Search

NL query

"read an object from an xml"

Learning-based Code Search Engine

Code snippet

```
public static < S > S deserialize(Class c, File xml) {
    try {
        JAXBContext context = JAXBContext.newInstance(c);
        Unmarshaller unmarshaller = context.createUnmarshaller();
        S deserialized = (S) unmarshaller.unmarshal(xml);
        return deserialized;
    } catch (JAXBException ex) {
        log.error("Error-deserializing-object-from-XML", ex);
        return null;
    }
}
```

Note: Similar terms, but no the same
Overview

Code vector → Cosine similarity ← Description vector

Code embedding network

↑

Code

Description embedding model

↑

NL description
Neural model

code vector → cosine simil. → description vector

max pooling

↑ ↑ ↑

\( t_0 \ t_2 \ t_3 \ldots \)

Code: set of tokens

Description: seq. of words

RNN

↑ ↑ ↑

\( w_1 \ w_2 \ w_3 \ldots \)
Training the Model

- **Train with pairs of code snippet** $c$ and **NL query** $d$
  - Matching pairs $(c, d_+)$
  - Non-matching pairs $(c, d_-)$

**Loss function:**

$$
\mathcal{L}(\theta) = \sum_{<C, D^+, D^-> \in P} \max(0, \epsilon - \cos(c, d^+) + \cos(c, d^-))
$$

"Deep Code Search" (Gu et al., 2018)
Results

- Model trained on 18 million Java methods and their comments (as a surrogate for NL queries)
- Evaluation with 50 questions from stackoverflow.com
  - Correct code snippet predicted at position 1 or 2 for most queries