Learning Uniform Semantic Features for Natural Language and Programming Language Globally, Locally and Sequentially

Yudong Zhang[1], Wenhao Zheng[1,3], Ming Li[1,2]

[1]National Key Laboratory for Novel Software Technology, Nanjing University, China
[2]Collaborative Innovation Center of Novel Software Technology and Industrialization, Nanjing, China
[3]Search Product Center, WeChat Search Application Department, Tencent, China

Learning And Mining from DatA
http://lamda.nju.edu.cn

Motivations & Objectives

Motivation

- Learning semantic features for textual data (code snippets & text sentences) is essential in addressing many software mining tasks
  - Code clone detection, bug localication, code annotation, etc.
- It is **hard** to maintain original semantic similarity in the learned features, because those features usually contain incomplete semantic information
- Poor-quality features prevent models from achieving good performances

Research Goal

- Learn features with complete semantics for both natural and programming languages

Three-Level Semantic Information: Global, Local, Sequential

Source Code Snippets

```
public static int array_top1(int[] array) {
        Array sort(array);
        return array[array.length - 1];
}
public static int find_zero(int[] array) {
        int index = array.length - 1;
        while(index>= 0 && array[index] != 0) {
            index --;
        }
        return index;
}
```

Global Information

```
public static int sort1(int[] array) {
            array[4] *= 10;
            Array.sort(array);
            return array;
}
public static int sort2(int[] array) {
            Array.sort(array)
            array[4] *= 10;
            return array;
}
```

Sequential Information

```
public static int[][] loop1(int[][] array, int col) {
        for(int row=0; row<10; ++row) {
            array[row][col] = row*10 + col;

        }
        return array;
}
public static int[][] loop2(int[][] array, int col) {
        for(int row=0; row<10; ++row) {
            array[row][col] = col*10 + row;

        }
        return array;
}
```

Local Information

Text Sentences

- Global Information
  - *"Write a program in Python"*
  - *"Debug the Jave code"*
- Local Information
  - *"Depth First Search"*
- *Sequential Information*
  - *"Apply quick sort to the array"*
  - *"Quick to apply array the sort"*

1. The three levels of information are **complementary** rather than implicit
2. Both languages contain the three levels of semantic information

Proposed Framework

Framework Structure

- Encoding layer
- Multi-info layer
  - Global-info branch, local-info branch, sequential-info branch
- Feature fusion layer
- Attention layer
  - Learns importance weights for extracted global, local and sequential features

Similariity Measurement

- Cosine distance

$$D_c(e_1, e_2) = 1 - \cos\langle R(e_1), R(e_2)\rangle = 1 - \frac{e_1 \cdot e_2}{\|e_1\| \, \|e_2\|}$$

- Prediction

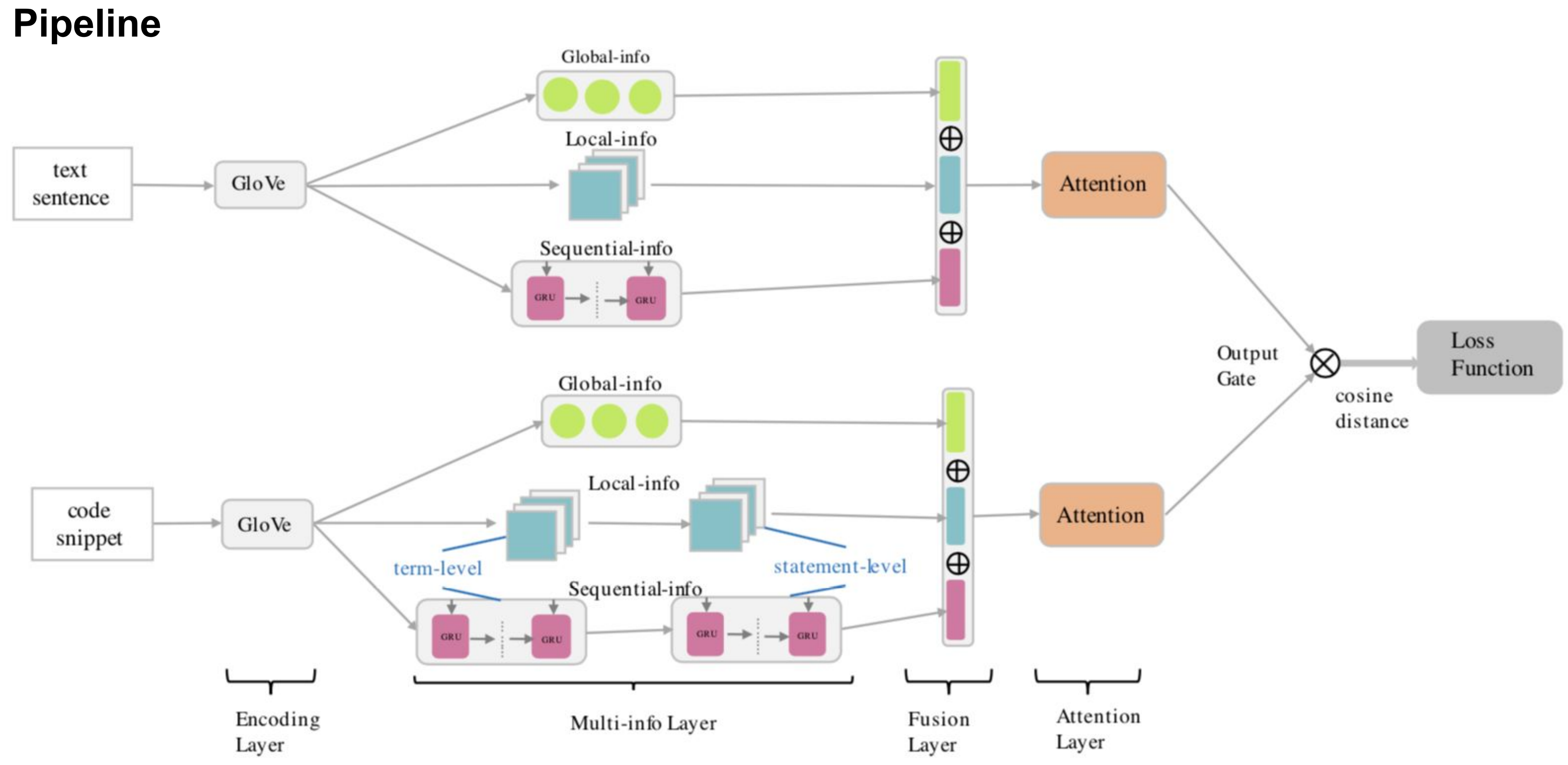
$$\text{prediction} = \text{sigmoid}(a * D_C(e_1, e_2) + b)$$

Optimization

- Triplet loss

$$\text{Loss}(e, p, n) = \max(D_c(e, p) - D_c(e, n) + \alpha, 0)$$

Pipeline



Difference between code pipeline and text pipeline

- Code snippets contain both *word-level* and *statement-level* information, so we designed hierarchical network structures in Multi-info layer to extract features
- Text sentences contain only *word-level* information

Experiments

Settings

- We evaluate our model on three real-world software mining tasks
  - Duplicate programming question classification, code clone detection, code annotation classification
- Two training strategies
  - Single training (ST): train **seperately** on different tasks
  - Joint Training (JT): train **jointly** on multiple tasks
- Statistics of datasets

Dataset	#Instances	#Train	#Valid	#Test
dup-question	535,254	495,254	20,000	20,000
code-anno	693,026	653,026	20,000	20,000
code-clone	33,690	27,690	3,000	3,000

Results

	dup-question		*code-clone*		*code-anno*	
Methods	AUC	F-measure	AUC	F-measure	AUC	F-measure
SourcererCC	–	–	53.21%	4.80%	–	–
CDLH	–	–	62.03%	59.89%	–	–
AP	72.23%	68.13%	57.20%	54.86%	62.18%	53.79%
MaLSTM	70.83%	58.67%	50.56%	55.08%	60.95%	50.29%
UniEmbed(ST)	92.64%	83.34%	70.50%	66.50%	75.15%	68.82%
UniEmbed(JT)	**95.44%**	**87.70%**	**87.69%**	**80.31%**	**78.21%**	**71.85%**

- Our model outperforms other state-of-the-art models on all evaluated tasks, and shows the superiority of the three-level semantic information for complete semantic features
- Joint training achieves better results than single training due to the shared knowledge between tasks

Contributions

- We introduce an approach to extract three-level semantic information, namely global, local and sequential information, in learning complete semantic features for textual data (code snippets & text sentences).
- We propose a novel representation learning framework called UniEmbed, which can learn uniform semantic features for natural language and programming language when both types of data are involved in the same task. Such representations are capable and effective in helping addressing a set of real-world software mining tasks.

AAAI 2019