

Определение наилучшего ответа на StackOverflow

Никита Подгузов

Научный руководитель: Рауф Курбанов

Санкт-Петербургский Академический университет

30 марта 2018 года

Возможности сервисов вопросов и ответов:

- Задавать вопрос (и отмечать правильный ответ)
- Отвечать на вопросы, заданные другими пользователями
- Голосовать за понравившиеся ответы

YAHOO! ANSWERS

Quora

 stackoverflow

Особенности системы:

- Узкоспециализированная
- Большая база вопросов
- Наличие сниппетов кода в вопросах и ответах

stackoverflow Questions Developer Jobs Tags Users Search...

What is the "-->" operator in C++?

After reading [Hidden Features and Dark Corners of C++/STL](#) on [comp.lang.c++.moderated](#), I was completely surprised that the following snippet compiled and worked in both Visual Studio 2008 and G++ 4.4.

Here's the code:

```
#include <stdio.h>
int main()
{
    int x = 10;
    while (x--> 0) // x goes to 0
    {
        printf("Id ", x);
    }
}
```

I'd assume this is C, since it works in GCC as well. Where is this defined in the standard, and where has it come from?

asked 8 years, 5 months ago
viewed 595,100 times
active 20 days ago

Ask Question

7204

To better understand, the statement could be written as follows:

```
while( (x--) > 0 )
```

That's a very complicated operator, so even ISO/IEC JTC1 (Joint Technical Committee 1) placed its description in two different parts of the C++ Standard.

Joking aside, they are two different operators: -- and > described respectively in §5.2.6/2 and §5.9 of the C++03 Standard.

21 Answers

active oldest votes

--> is not an operator. It is in fact two separate operators, -- and >.

The conditional code decrements `x`, while returning `x`'s original (not decremented) value, and then compares the original value with `0` using the `>` operator.

edited Feb 13 '17 at 8:32
community wiki
28 revs, 20 users 35%
GManNickG

2183

edited Mar 2 '17 at 6:49
community wiki
11 revs, 8 users 33%
Kirill V. Lyachinsky

Введение

Постановка задачи

Проблемы:

- Большая доля "неразрешенных" вопросов
- Нет возможности помочь оценить правильность ответов пользователю, задавшему новый вопрос

Хотим научиться определять правильные ответы, используя базу вопросов StackOverflow

Введение

Google & StackOverflow

Google ruby capitalize

About 369.000 results (0,34 seconds)

Capitalize first letter in ruby - Stack Overflow
<https://stackoverflow.com/questions/3724913/capitalize-first-letter-in-ruby>
Sep 16, 2010 - Unfortunately, it is impossible for a machine to upcase/downcase/capitalize properly. ... That's why Ruby's String class only supports capitalization for ASCII characters, because there it's at least somewhat well-defined.

How to convert a string to lower or upper case in Ruby 21 Dec 2014
ruby - Using 'capitalize' or 'capitalize!' 29 Sep 2014
Capitalize the first letter of each word - Ruby 26 Jan 2014
Capitalize only first character of string and leave others alone ... 12 Nov 2011
More results from stackoverflow.com

Class: String (Ruby 2.2.0) - Ruby-Doc.org
<https://ruby-doc.org/core-2.2.0/String.html>
Jump to **capitalize** - capitalize => new_str click to toggle source. Returns a copy of str with the first character converted to uppercase and the remainder to lowercase. Note: case conversion is effective only in ASCII region. "hello".capitalize #=> "Hello" "HELLO".capitalize #=> "Hello" "123ABC".capitalize #=> "123abc".
::try_convert - capitalize! - chomp - gsub

Old version

Google ruby capitalize

Alle Nieuws Shopping Afbeeldingen Video's Meer Instellingen Tools

Ongeveer 370.000 resultaten (0,34 seconden)

Capitalize first letter in ruby - Stack Overflow
<https://stackoverflow.com/questions/3724913/capitalize-first-letter-in-ruby>
The upcase method capitalizes the entire string. I need to capitalize only the first letter. Also, I need to support several popular languages, like German and Russian. How do I do it?
7 antwoorden

Best Answer 197 votes	Antwoord 2 van 7 37 votes	Antwoord 3 van 7 18 votes
It depends on Ruby version you use. Ruby 2.4 and higher It just works, as since this version ruby supports Unicode case mapping. "wapw".capitalize	capitalize first letter of first word of string "kirk douglas".capitalize #=> "Kirk douglas" capitalize first letter of each word in rails: "kirk"	Unfortunately, it is impossible for a machine to upcase/downcase/capitalize properly. It needs way contextual information

Class: String (Ruby 2.2.0) - Ruby-Doc.org
<https://ruby-doc.org/core-2.2.0/String.html> - Vertaal deze pagina
Spring naar **capitalize** - capitalize => new_str click to toggle source. Returns a copy of str with the first character converted to uppercase and the remainder to lowercase. Note: case conversion is effective only in ASCII region. "hello".capitalize #=> "Hello" "HELLO".capitalize #=> "Hello" "123ABC".capitalize #=> "123abc" ...
::try_convert - #[] - count - dump

New version

Введение

Обзор имеющихся решений

"Towards Predicting the Best Answers in CB QAS" (Tian et al. 2013)

- Три вида фичей: $A \leftrightarrow A$, $A \leftrightarrow Q$, A
- Использование Vector Space Model + TF-IDF для определения похожести
- Использование лингвистических фичей (длина текста, количество предложений, читаемость и др.)
- Учитывается лишь наличие/отсутствие сниппетов кода
- Random Forest Classifier

Введение

Обзор имеющихся решений

"State of the art Best Answer Prediction based on Discretisation of Shallow Linguistic Features" (Gkotsis et al. 2014),

"Moving to Stack Overflow: Best-Answer Prediction in Legacy Developer Forums" (Calefato et al. 2016)

- Четыре вида фичей: $A \leftrightarrow A$, A , $user-rating$ и $answer-rating$, $thread$
- Использование лингвистических фичей (длина текста, количество предложений, читаемость и др.)
- Использование вероятностной униграммной модели для оценки вероятности ответа
- Использование группировки ответов и дискретизации фичей
- Не учитывает сниппеты кода
- Alternating Decision Tree Classifier

Введение

Минусы имеющихся решений

- Не используется текст вопроса
- Не учитываются синонимы и похожие слова, то есть игнорируется семантика
- Не используется содержание сниппетов кода

Цели и задачи

Цель: научиться определять правильность ответа на StackOverflow, используя как его текст, так и код, который может присутствовать внутри ответа

Задачи:

- Реализовать классификатор на основе рекуррентных нейронных сетей, использующий текст ответов
- Добавить извлечение фичей из кусков кода и использовать их в классификаторе
- Сравнить результаты с имеющимися работами
- Проанализировать влияние наличия фичей от сниппетов кода на точность классификации

Данные

Общие факты

Данные:

- Дамп базы вопросов StackOverflow
- XML-файл размером $\sim 50GB$
- Формат файла: *type_id, id, score, date, body*

Анализ:

- 40 миллионов постов, из них 16 миллионов вопросов и 24 миллионов ответов
- 7 миллионов вопросов (47%) без отмеченного правильного ответа
- 2 миллиона вопросов (13%), у которых нет ни одного ответа

Обработка:

- Удаляем вопросы с рейтингом ≤ 0 , а также вопросы, у которых нет ни одного ответа
- Из остальных постов извлекаем его *body*
- Сохраняем весь код, находящийся в тегах `<code>`
- Очищаем от тегов и сохраняем весь остальной текст вопроса/ответа

После обработки получили 6.5 миллионов вопросов, из них 2 миллиона вопросов (31%) без правильного ответа, а также 13.5 миллионов ответов

Подходы к анализу текста

Bag of words

Идея:

- Каждому слову сопоставляем вектор длины, равной размеру словаря
- Документ: сумма векторов слов

Проблемы:

- Не учитывается семантика
- Не учитывается порядок слов
- Большая размерность

Document 1

The quick brown
fox jumped over
the lazy dog's
back.

Document 2

Now is the time
for all good men
to come to the
aid of their party.

Term	Document 1	Document 2
aid	0	1
all	0	1
back	1	0
brown	1	0
come	0	1
dog	1	0
fox	1	0
good	0	1
jump	1	0
lazy	1	0
men	0	1
now	0	1
over	1	0
party	0	1
quick	1	0
their	0	1
time	0	1

**Stopword
List**

for
is
of
the
to

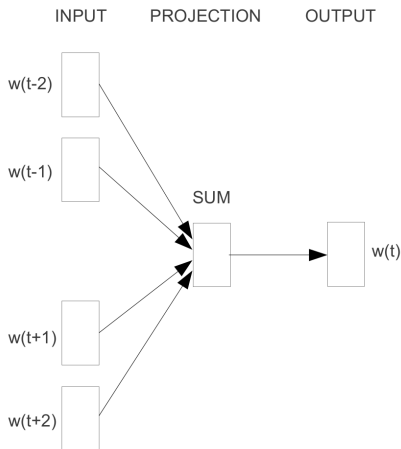
Подходы к анализу текста

Word2Vec

Идея:

- Каждому слову сопоставляем вектор фиксированной длины
- Хотим, чтобы вектор отражал смысл слова
- Обучаем на неразмеченном корпусе текстов CBOW/Skip-gram модель

Чтобы учесть специфику технического языка, обучаться лучше на текстах StackOverflow



CBOW

Подходы к анализу текста

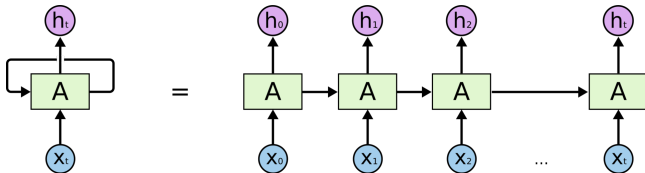
Рекуррентные нейронные сети

Хотим научиться учитывать порядок слов в тексте

Идея:

- Используем embedding слов из Word2Vec
- Отдаем на вход клетке сети новое слово и выход с предыдущей (учет контекста)
- Хотим, чтобы выход сети отражал смысл входного текста

Также используются двунаправленные рекуррентные нейронные сети для захвата контекста справа



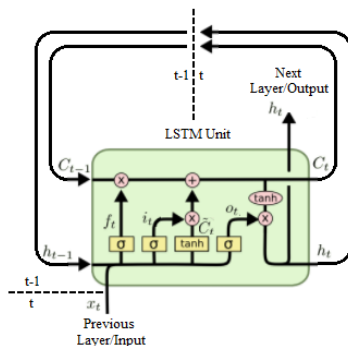
Подходы к анализу текста

LSTM

Проблема долговременных зависимостей

Идея:

- Вводится состояние клетки
- На каждом шаге забываем какую-то часть информации из состояния и добавляем новую



$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$$

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C)$$

$$C_t = f_t \otimes C_{t-1} + i_t \otimes \tilde{C}_t$$

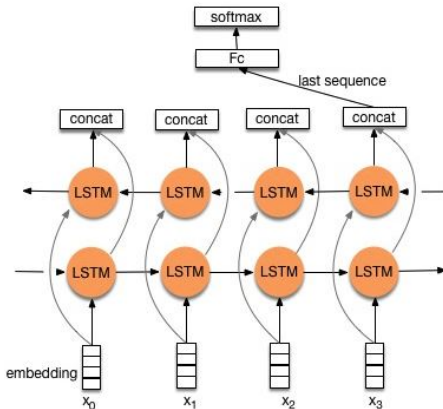
$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$$

$$h_t = o_t \otimes \tanh(C_t)$$

Классификация текстов

Базовая архитектура

- Классификация ответов на два класса: правильный/неправильный
- В качестве embedding-а используется векторное представление, обученное на текстах вопросов и ответов StackOverflow



Классификация ответов

Учет текста вопроса

-

Подходы к анализу кода

Проблемы

Код похож на текст, поэтому можно попробовать применить аналогичные методы

Проблема: нелинейная структура кода (циклы, ветвления и др.)

Замечание: тем не менее, может хорошо сработать для однострочных сниппетов

Подходы к анализу кода

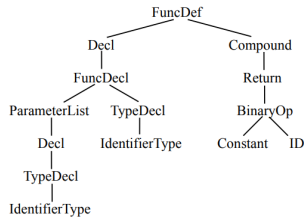
Использование синтаксического дерева

Идея:

- Вместо текста кода рассмотрим его синтаксическое дерево
- Обучаем Code2Vec, используя в качестве контекста сыновей в синтаксическом дереве, нормализованных по размеру их поддерев

```
double doubles(double doublee){  
    return 2 * doublee;  
}
```

A C code snippet



The corresponding AST

Подходы к анализу кода





Использование метода

- Обучаем модель на корпусе кода фиксированного языка программирования (например, Python)
- В качестве embedding-ов вершин синтаксического дерева используем полученные векторные представления
- Используем RNN, как в случае текста, отдавая на вход полученные embedding-и в порядке обхода *dfs*-ом

Выводы

Результаты

- *сравнение с имеющимися решениями*
- Анализ влияния наличия фичей от текста вопроса: как улучшилась классификация после добавления учета семантической похожести вопроса и ответа?
- Анализ влияния наличия фичей от сниппетов кода: как улучшилась классификация после добавления учета содержания сниппетов кода?

- ①  [Tian et al. \(2013\)](#)
Towards Predicting the Best Answers in Community-Based Question-Answering Services
- ②  [Gkotsis et al. \(2014\)](#)
It's all in the Content: State of the art Best Answer Prediction based on Discretisation of Shallow Linguistic Features
- ③  [Calefato et al. \(2016\)](#)
Moving to Stack Overflow: Best-Answer Prediction in Legacy Developer Forums
- ④  [Mou et al. \(2014\)](#)
Building Program Vector Representations for Deep Learning