Explanation of the Test and Analysis of the Results

# Explanation

This document will analyze the results of the study that was completed on the html editor to evaluate the performance of the editor.

The html editor optimizes the process of programming in html because it follows the programmer’s patters to provide a personalized auto-complete. The auto-complete list created by the editor is ranked according to how much the programmer is using an element of the list, which could be an html element, an attribute type, an attribute value, or even a group of elements. The study evaluated how close the autocomplete presented by the editor was to providing the programmer the element that he/she is going to type next.

The study was completed by collecting the html source code of 100 web pages and imputing these source code into the editor. The source code parsed and divided in to three text files: A Training set file, a Testing set file and an Answer file. The Answer file corresponds to the expected element that should be on the auto-complete list.

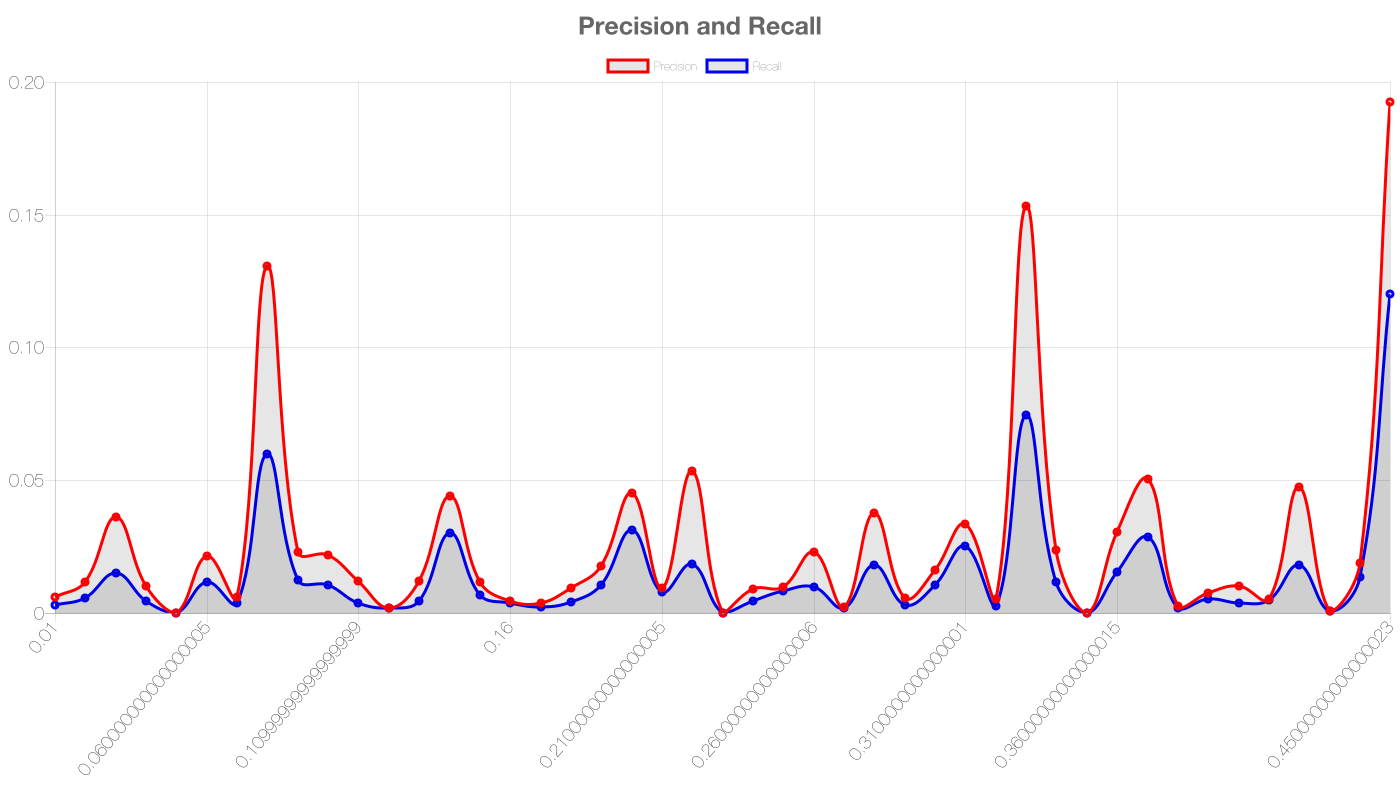
The system used Node.js to execute the study in the server-side, and a library called jsdom that created an instance of the editor in the server-side. The Training set was used to allow the editor to populate an autocomplete list. Afterwards, the Testing set was used to simulating a programmer typing code, in turn the editor “presented” a frequency ranked auto-complete table of all html elements that the “programmer” had previously imputed. The editor did not present a table during the study, but the table that would be presented if the editor was not running on the server-side can be retrieved from the instance of the editor. Later, the ranked auto-complete list provided by the editor was compared to the Answer file that was created from the html source code.

The Training and Testing set that were given to the editor was specifically created to test how well the editor is on presenting the programmer valuable attributes of elements on the auto-complete. However, the system can be used to test for other instances, such as for how well the editor presents an auto-complete for group of elements, but new Training set, Testing set, and Answers have to be created.

The metric used to evaluate the results of the editor was Precision and Recall at K. This is the best metric to analyze a system that provides a list of values that are ranked according to the programmer’s interest. After getting the precision and recall for each attribute tested on each html source code of a web page, the study system created a comma-separate values (CSV) file to save the results. Subsequently, the system averaged all the precisions and recall resulted for each html source code. Using the CSV file resulted from the study, the system created a graph that presented the results of the averaged precision and recall, in addition to another CSV file containing the average of each source code precision and recall results. The graphs and the average results are shown below.

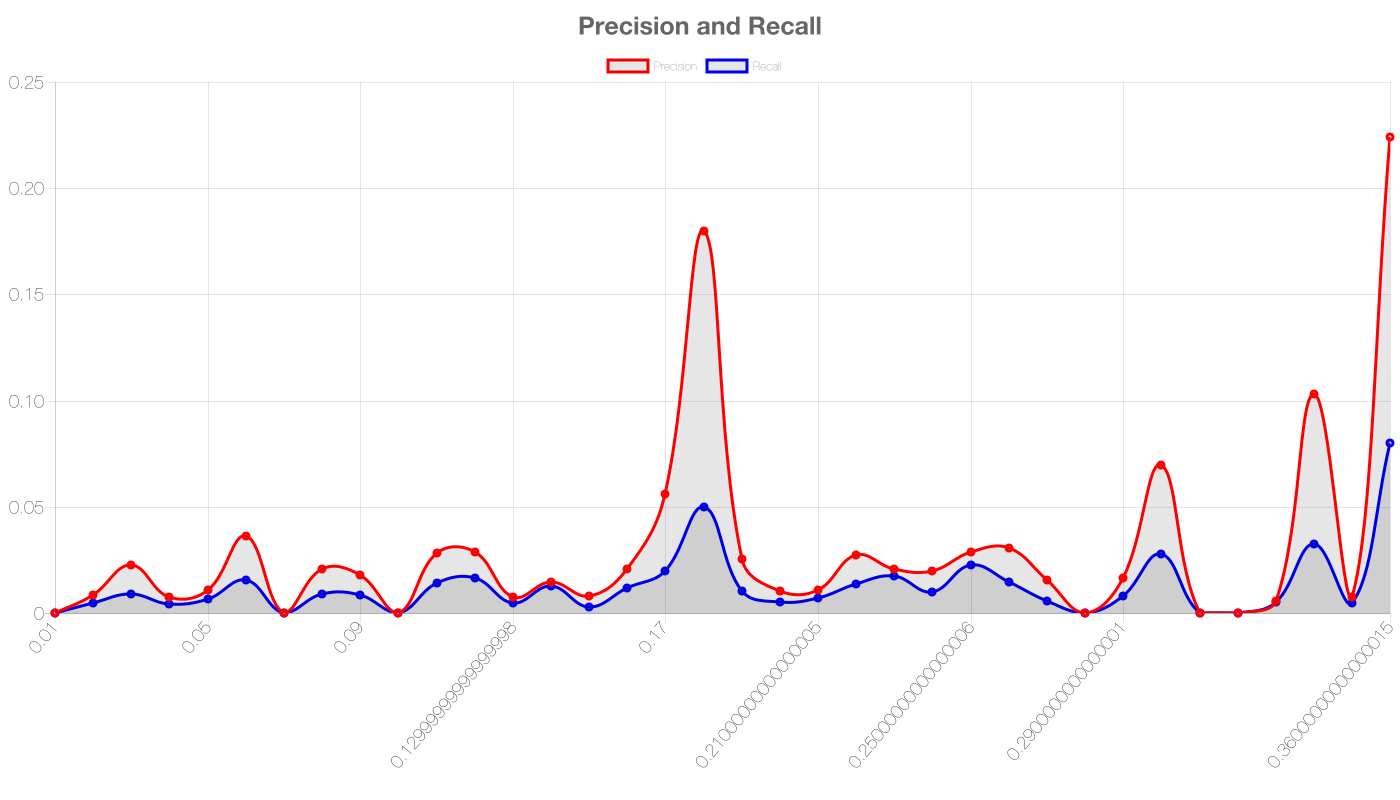
Table 1: Precision and Recall results of the first 50 web pages source code:

Table 2: Precision and Recall results of the rest 50 web pages source code:

Precision and Recall Graph of Results from HTML pages 1-50 in Alphabetical Order

Graph 1: Most of the results are bellow 0.05 for both precision and recall, which is not a desirable result.

Precision and Recall Graph of Results from HTML pages 51-100 in Alphabetical Order

Graph 2: Similarly Graph 1, most of the results are bellow 0.05, which is not a desirable result.

# Analysis

The result of both precision and recall were bellow 0.05. The average precisions results were better than the average recall results, however only in a few cases they both were above 0.05. On the cases that the precision and recall were above 0.05, the precision increased significantly more than the recall.

# How to improve

To improve the editor’s performance for presenting a good ranked auto-complete list of attributes:

* The editor could perform better if it knows when the programmer is writing an element, an attribute type, or an attribute value, and create a list in accordance to what the programmer is currently writing.
* The editor can create different auto-complete list that is more focus on what the programmer is writing.
* The editor could perform better if it trains with source code of other HTML pages, preferably from the programmer (maybe every time the programmer writes code on the editor, the editor saves the programmer’s coding style).