ProFoUnd: Program-analysis-based Form Understanding

**Motivation and Contributions**

In the context of information extraction, it is often necessary to return all information that is relevant from a given website. This is a challenging task, especially for websites that have a structured and well-designed layout. ProFoUnd, the system described in this paper, is designed to automatically identify form components and extract their content from a website.

**ProFoUnd Architecture**

We take a program-analysis approach to identify all form components on a website. This involves analyzing the source code of the website to identify the form components and their values. The architecture of ProFoUnd is divided into several stages:

1. **ProFoUnd Architecture**: Identifying the form fields on the website.
2. **Error Message Analysis**: Analyzing error messages to identify form constraints.
3. **Variable Identification**: Identifying variables used in the form.
4. **Ternary Operators**: Handling ternary operators in the form.
5. **DOM Access**: Accessing the DOM to identify form components.
6. **Form Field Validation**: Validating the form fields.
7. **Form Field Interception**: Intercepting form fields for validation.

**An Example**

Given a search interface, ProFoUnd can automatically identify the form fields and their values. This is achieved by analyzing the source code of the website and using program analysis techniques to identify the form fields and their values.

**Experiments and Results**

We conducted experiments to evaluate the performance of ProFoUnd. The results show that ProFoUnd is able to accurately identify form fields and their values. We also performed a comparison with other systems and found that ProFoUnd outperforms them in terms of accuracy.

**Where did we struggle?**

ProFoUnd had difficulties in handling form fields with complex conditions. In some cases, the system was not able to correctly identify the form fields and their values.

**Future Work**

We plan to further improve ProFoUnd by adding more sophisticated program analysis techniques. We also plan to extend the system to handle more complex form fields with nested conditions.

Authors

Michael Benedikt, Tim Furche, Andreas Savvides, Pierre Senellart

Contact

michael.benedikt@cs.ox.ac.uk
tim.furche@cs.ox.ac.uk
andreas.savvides@ibm.com
pierre.senellart@telecom-paristech.fr

Digital Home

http://diadem.cs.ox.ac.uk/profound/