Keyword Extraction Based on Implicit Feedback

Jakub Kříž

Supervisor: Tomáš Kramár

The goals

- colect and process the implicit feedback from the users on the internet
- use the feedback to infer which documents the user liked and disliked
- use the feedback to infer which parts of the documents interested him the most
- make keyword extraction more accurate

Implicit Feedback

- everything the user does in the web browser
- collected via implicit interest indicators
- when an indicator is detected the user is likely to be interested in that part of the document
- detected and calculated via JavaScript

The implicit interest indicators

- *global indicators* show interest in the entire document
- *local indicators* indicate the interest in a part of the document
- varying accuracy and reliability



	· · · ·	the degree of object orientation, following distinction can be made:		
Typing into a form.	· · · · · · · · · · · · · · · · · · ·	 Languages called "pure" OO languages, because everything in them is treated consistently as an object, from primitives such as characters and punctuation, all the way up to whole classes, prototypes, blocks, modules, etc. They were designed specifically to facilitate, even enforce, OO methods. Examples: Eiffel, Emerald.^[19], JADE, Obix, Scala, Smalltalle 		
	-	Languages designed mainly for OO programming, but with some procedural elements. Examples: C++, Java, C#, VB.NET, Python.		
	Comment Hello	• Languages that are historically procedural languages, but have been extended with some OO features. Examples: Visual Basic (derived from BASIC), Fortran 2003, Perl, COBOL 2002, PHP, ABAP.		
		• Languages with most of the features of objects (classes, methods, inheritance, reusability), but in a distinctly original form. Examples: Oberon (Oberon-1 or Oberon-2) and Common Lisp.		
		 Languages with abstract data type support, but not all features of object-orientation, sometimes called object-based languages. Examples: Modula-2 (with excellent encapsulation and information hiding), Pliant, CLU. 		
		OOP in dynamic languages	edit]	
		In recent years, object-oriented programming has become especially popular in dynamic programming languages. Python, Ruby and Groovy are dynamic languages built on OOP principles, while Perl and PHP have been adding object oriented features since Perl 5 and PHP 4, and ColdFusion since version 5.	,	
		The Document Object Model of HTML, XHTML, and XML documents on the Internet have bindings to the popular JavaScript/ECMAScript language. JavaScript is perhaps the best known prototype-based		
		programming language, which employs cloning from prototypes rather than inheriting from a class. Another scripting language that takes this approach is Lua. Earlier versions of ActionScript (a partial superset the ECMA-262 R3, otherwise known as ECMAScript) also used a prototype-based object model. Later versions of ActionScript incorporate a combination of classification and prototype-based object models ba	of ised	
		largely on the currently incomplete ECMA-262 R4 specification, which has its roots in an early JavaScript 2 Proposal. Microsoft's JScript.NET also includes a mash-up of object models based on the same proposal, and is also a superset of the ECMA-262 R3 specification.		•••Text selection and copying
		Design patterns	edit]	
		Challenges of object-oriented design are addressed by several methodologies. Most common is known as the design patterns codified by Gamma et al More broadly, the term "design patterns" can be used to refer to any general, repeatable solution to a commonly occurring problem in software design. Some of these commonly occurring problems have implications and solutions particular to object-oriented development.	C	
		Inheritance and behavioral subtyping	edit]	
		See also: Object oriented design		
		It is intuitive to assume that inheritance creates a semantic "is a" relationship, and thus to infer that objects instantiated from subclasses can always be safely used instead of those instantiated from the		
		superclass. This intuition is unfortunately false in most OOP languages, in particular in all those that allow mutable objects. Subtype polymorphism as enforced by the type checker in OOP languages (with		
		mutable objects) cannot guarantee behavioral subtyping in any context. Behavioral subtyping is undecidable in general, so it cannot be implemented by a program (compiler). Class or object hierarchies need to) be 🚽	

Processing the Indicators

- the document is broken down to text elements headings, paragraphs
- each element starts with a ranking of o

Evaluation

- we evaluate the two methods used for keyword extraction
 - » users considered 100% of the keywords extract-
- when an indicator is detected the ranking of the target element is increased
- the increase depends on reliability and accuracy of the indicator
- copied and selected keywords are extracted directly from the text and using tf-idf from the ranked elements

ed from the copied text to accurately represent their interest in the document

- » users considered 62% of the keywords extracted from the selected text
- experimental evaluation of the extraction from ranked elements is currently in progress

S T U • • • • • • • F I I T •

• • • • •

Institute of Informatics and Software Engineering Faculty of Informatics and Information Technologies Slovak University of Technology in Bratislava, Slovakia