IIAI Open Conference Publication Series
Volume title
Volume number, Article ID
DOI: https://doi.org/10.52731/XXXXXX

A Platform for Searching Texts for Desired Expressions in a User-editable Pattern Matching Environment for Language Learning

Tatsuya Katsura *, Koichi Takeuchi †

Abstract

In this paper we propose a platform of pattern matching system that can extract required phrases or sentences in texts. Finding certain expressions in texts are often needed in language learning, e.g, examples of case markers between a predicate and an argument, or possible nouns in subject of a verb in a certain meaning. In previous studies, several types of systems, containing concordancers, are proposed. However, it is not easy to apply combined patterns because the pattern mathing templates are previously fixed. Thus, we propose a flexible phrase searching system in which the users can create search patterns by combining blocks of basic search templates. One of the characteristics the proposed system is that the user can also specify where to be highligted in texts with the blocks. To realize the function of combining patterns by the users, the proposed system employs Prolog as the base of the data structure. The platform of the searching system is implemented on an Web server with JavaScript-based interface and database system. In the performance test, we shows that the proposed system can deal with relatively large sclale texts (10,000 sentences), and also demonstrate the combined patterns can be applied to the texts. In this paper we discuss the system architecture and the extendability of the pattern matching.

Keywords: Pattern matching, Concordancer, Block-based programming, Prolog

1 Introduction

Extraction of some phrases and expressions in texts is considered essential function in langugage education. For example, case markers located between a predicate and an object in Japanese language are various rules and alternations[1], then language learners need to search for examples of predicate-argument examples in Japanese texts. As a tool for searching texts, various kinds of concordancer are proposed, however, most concordancers are

* Graduate School of Environmental, Life, Natural Schience and Technology, Okayama University, Okayama, Japan

† Faculty of Environmental, Life, Natural Science and Technology, Okayama University, Okayama, Japan

[1] The verbs *morau* (get) and *eru* (obtain) are almost the same meaning but *morau* has alternation between the case markers *ni/kara* as *kare ni/kara morau* ((I) get (it) from him.), and *eru* can only takes *kara* in this meaning as *kare kara/*ni eru*.

targeted for English. From the results of natural language processing study, several dependency parsers for Japanese are available, thus it is possible to compose a complex text searching system if the language learners can build NLP tools. But language learners are not NLP or software engineer, thus, we need an environment to build patterns by searching for phrases or expressions without needing programming.

when user want to obtain some verb-nouns or predicate preposition, concodancer is available the basic functions character, word matchina are available, but more extendede cases are no method.

, however, more combined search, for example, semantic meaning of predicate buy-sell and linguitic alternation, kara/wo in Japanese dependency

日本語に対するパターンマッチングシステム

2 Related Studies

An IIAI electronic copyright form should accompany your final submission or registration of the conference. The instruction of copyright transfer is announced at the conference webpage. Authors are responsible for obtaining any security clearances for the copyright form submission.

3 Platfrom of Pattern Matching System

user editable pattern with visual handling, Construction of pattern is not easy, the user shoud apply the assumed patterns, see the results and then tune up the patterns. Thus, editing space and results should be exist.

Patterns are combinations, conbining by basic blocks and extract every where matched.

4 既存の言語パターンマッチシステムの概要と改善点

This chapter describes the overall picture of the existing pattern matching system and the problems that need to be improved. Section 2.1 describes the overall picture of the existing pattern matching system, Section 2.2 lists problems with the existing system, and Section 2.3 describes proposed improvements to the problems listed in Section 2.2.

4.1 既存の言語パターンマッチシステムの全体像

This section describes the overall picture of the existing pattern matching system. The existing pattern-matching system is shown in Figure 1. Figure 1 shows the existing pattern matching system. Existing pattern matching systems analyze text sentences from which information is to be extracted, and from the analysis results, users prepare patterns of sentence structures corresponding to sentences or parts of sentences related to the target in advance, and obtain results that match the patterns. The actual flow of the process is shown in Figure

Figure 1: 既存のパターンマッチシステムの構成図

Existing system is a front that the user actually operates. The system consists of a front end

Figure 2: 既存のパターンマッチシステムの処理の流れ

and a back end that processes data received from the front end. The details are described in sections 2.1.1 and 2.1.2 below.

4.1.1 フロントエンド

The frontend is mainly built using the JavaScript framework React, and has the following functions.

コーパスアップロード
: Upload the corpus file to be analyzed. Sentence breaks are recognized by punctuation marks, exclamation points and question marks.

コーパス解析結果表示

言語パターン構築
: Using Blockly, you can construct the language pattern you want to search for by visually combining blocks. By combining many different types of blocks, the information in the ASA analysis results can be complex. By combining many different types of blocks, it is possible to search for information on ASA parsing results in complex ways. Figure 1. Figure is a language pattern for extracting authors and works. It is also possible to export the constructed language patterns to xml format files. The exported file can also be imported.

言語パターンマッチ結果表示
: Displays the results of language pattern matching. The display format includes KWIC format, highlighting format, and table format. Figure shows the result of executing language pattern matching using the language pattern in Figure

4.1.2 バックエンド

The backend is built on Django, a Python web framework. It is implemented using asapy for analysis by ASA and prologpy for Prolog processing.

コーパス解析

Reads the corpus to be parsed, parses it with ASA, and converts it to Prolog. The generated analysis results are returned to the front end and stored in the browser's memory.

言語パターンマッチ

Query the target Prolog and search query using the Prolog processor. The generated pattern matching results are returned to the front end and stored in the browser memory.