Online Information Retrieval for Language Learning

Maria Chinkina **Madeeswaran Kannan** **Detmar Meurers**

Universität Tübingen
LEAD Graduate School
Department of Linguistics
{mchnkina,mkannan,dm}@sfs.uni-tuebingen.de

Abstract

The reading material used in a language learning classroom should ideally be rich in terms of the grammatical constructions and vocabulary to be taught and in line with the learner's interests. We developed an online Information Retrieval system that helps teachers search for texts appropriate in form, content, and reading level. It identifies the 87 grammatical constructions spelled out in the official English language curriculum of schools in Baden-Württemberg, Germany. The tool incorporates a classical efficient algorithm for reranking the results by assigning weights to selected constructions and prioritizing the documents containing them. Supplemented by an interactive visualization module, it allows for a multifaceted presentation and analysis of the retrieved documents.

1 Introduction

The learner's exposure to a language influences their acquisition of it. The importance of *input* in second language (L2) learning has been repeatedly emphasized by the proponents of major Second Language Acquisition theories (Krashen, 1977; Gass and Varonis, 1994; Swain, 1985), with psycholinguists highlighting the significance of frequency and perceptual salience of target constructions (e.g., Slobin, 1985).

In line with this research, a pedagogical approach of input flood (Trahey and White, 1993) is extensively used by L2 teachers. However, manually searching for linguistically rich reading material takes a lot of time and effort. As a result, teachers often make use of easily accessible schoolbook texts. However, this limits the choice of texts, and they are typically less up-to-date and less in line with students' interests than authentic texts. In the same vein, a survey conducted by Purcell et al. (2012) revealed that teachers expect their students to use online search engines in a typical research assignment with a very high probability of 94%, compared to the 18% usage of printed or electronic textbooks.

With this in mind, we developed an online Information Retrieval (IR) system that uses efficient algorithms to retrieve, annotate and rerank web documents based on the grammatical constructions they contain. The paper presents *FLAIR*[1] (Form-Focused Linguistically Aware Information Retrieval), a tool that provides a balance of content and form in the search for appropriate reading material.

2 Overview and Architecture

The *FLAIR* pipeline can be broadly reduced to four primary operations – Web Search, Text Crawling, Parsing and Ranking. As demonstrated by the diagram in Figure 1, the first three operations are delegated to the server as they require the most resources. Ranking, however, is performed locally on the client endpoint to reduce latency.

Web Crawling

We chose to use Microsoft Bing[2] as our primary search engine given its readily available Java bindings. By default, the top 20 results are fetched for any given search query. A basic filter is applied to exclude web documents with low text content. The search is conducted repeatedly until the resulting list of documents contains at least 20 items.

[1] The online tool is accessible at: http://purl.org/icall/flair

[2] http://bing.com

Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics—System Demonstrations, pages 7–12,
Berlin, Germany, August 7-12, 2016.

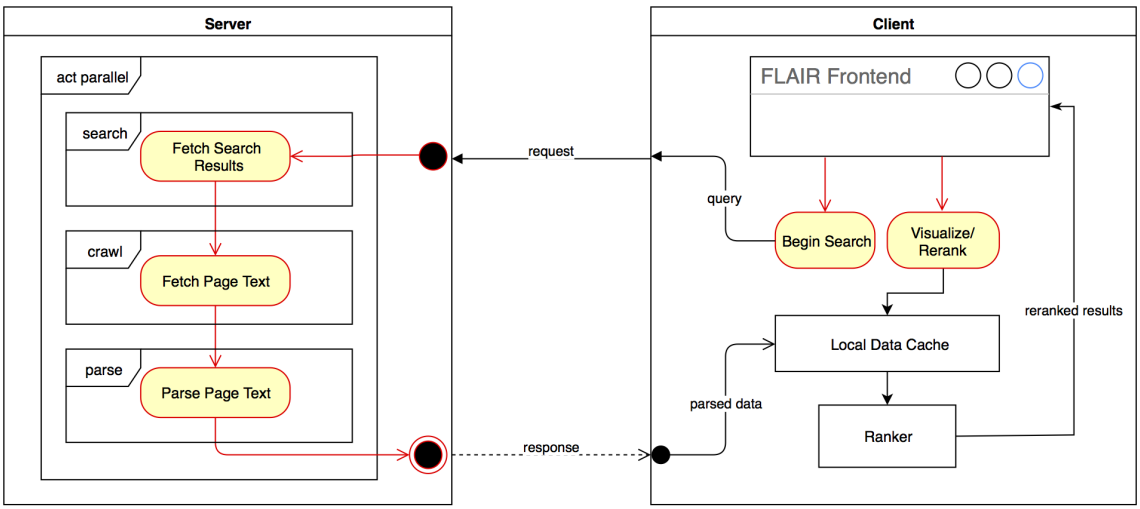

Figure 1: FLAIR architecture.

Text Extraction

The Text Extractor makes use of the Boilerpipe library[3] extracting plain text with the help of its DefaultExtractor. The choice is motivated by the high performance of the library as compared to other text extraction techniques (Kohlschütter et al., 2010).

Parsing

Text parsing is facilitated by the Stanford CoreNLP library[4] (Manning et al., 2014), which was chosen for its robust, performant and open-source implementation. Our initial prototype used the standard PCFG parser for constituent parsing, but its cubic time complexity was a significant issue when parsing texts with long sentences. We therefore switched to a shift-reduce implementation[5] that scales linearly with sentence and parse length. While it resulted in a higher memory overhead due to its large language models, it allowed us to substantially improve the performance of our code.

Ranking

The final stage of the pipeline involves ranking the results according to a number of grammatical constructions and syntactic properties. Each parameter can be assigned a specific weight that then affects its ranking relative to the other parameters. The parsed data is cached locally on the client side for each session. This allows us to perform the ranking calculations on the local computer, thereby avoid a server request-response roundtrip for each re-ranking operation.

We chose the classical IR algorithm BM25 (Robertson and Walker, 1994) as the basis for our ranking model. It helps to avoid the dominance of one single grammatical construction over the others and is independent of the normalization unit as it uses a ratio of the document length to the average document length in the collection. The final score of each document determines its place in the ranking and is calculated as:

$$G(q,d) = \sum_{t \in q \cap d} \frac{(k+1) \times tf_{t,d}}{tf_{t,d} + k \times (1 - b + b \times \frac{|d|}{avdl})} \times log \frac{N+1}{df_t}$$

where q is a *FLAIR query* containing one or more linguistic forms, t is a linguistic form, d is a document, $tf_{t,d}$ is the number of occurrences of t in d, $|d|$ is document length, $avdl$ is the average document length in the collection, df_t is the number of documents containing t, and k is a free parameter set to 1.7. The free parameter b specifies the importance of the document length. The functionality of the tool allows the user to adjust the importance of the document length with a slider that assigns a value from 0 to 1 to the parameter b.

2.1 Technical Implementation

FLAIR is written in Java and implemented as a Java EE web application. The core architecture revolves around a client-server implementation that

[3] https://code.google.com/p/boilerpipe/
[4] http://nlp.stanford.edu/software/corenlp.shtml
[5] http://nlp.stanford.edu/software/srparser.shtml

uses WebSocket (Fette and Melnikov, 2011) and Ajax (Garrett and others, 2005) technologies for full-duplex, responsive communication. All server operations are performed in parallel, and each operation is divided into subtasks that are executed asynchronously. Operations initiated by the client are dispatched as asynchronous messages to the server. The client then waits for a response from the latter, which are relayed as rudimentary push messages encoded in JSON.[6] By using WebSockets to implement the server endpoint, we were able to reduce most of the overhead associated with HTTP responses.

The sequence of operations performed within the *client* boundary is described as follows:

1. Send search query to server and initiate web search
2. Wait for completion signal from server
3. Initiate text parsing
4. Wait for completion signal from server
5. Request parsed data from server
6. Cache parsed data
7. Re-rank results according to parameters

The sequence of operations performed within the *server* boundary is described as follows:

1. Receive search query from client
2. Begin web search operation:
   (a) Fetch top N valid search results
   (b) For each search result, fetch page text
   (c) Signal completion
3. Wait for request from client
4. Begin text parsing operation:
   (a) For each valid search result, parse text and collate data
   (b) Signal completion
5. Wait for request from client
6. Send parsed data to client

[6] http://json.org

3 FLAIR Interface

The main layout consists of four elements – a settings panel, a search field, a list of results, and a reading interface, where the identified target constructions are highlighted. The interactive visualization incorporates the technique of parallel coordinates used for visualizing multivariate data (Inselberg and Dimsdale, 1991).

The visualization provides an overview of the distribution of the selected linguistic characteristics in the set of retrieved documents. Vertical axes represent parameters – linguistic forms, number of sentences, number of words and the readability score, and each polyline stands for a document having certain linguistic characteristics and thus, going through different points on the parameter axes. The interactive design allows for more control over a user-selected set of linguistic characteristics. Users can select a range of values for one or more constructions to precisely identify and retrieve documents.

Figures 2 and 3 demonstrate *FLAIR* in use: The user has entered the query *Germany* and selected *Past Perfect* and *Present Perfect* as target constructions. After reranking the 20 retrieved documents, the interactive visualization was used to select only the documents with a non-zero frequency of both constructions.

4 Detection of Linguistic Forms

We based our choice of the 87 linguistic forms on the official school curriculum for English in the state of Baden-Württemberg, Germany.[7] As most of the linguistic structures listed there do not have a one-to-one mapping to the standard output of NLP tools, we used a rule-based approach to approximate them.

For closed word classes, string matching (e.g., *articles*) or look-up lists (e.g, *prepositions*) can be used to differentiate between their forms. However, detection of some grammatical constructions and syntactic structures requires a deeper syntactic analysis. Identification of the degrees of comparison of *long adjectives* requires keeping track of two consequent tokens and their POS tags, as is the case with the construction *used to* that cannot be simply matched (cf. the passive *"It is used to build rockets"*). More challenging structures, such

[7] The curricula for grades 2, 4, 6, 8, 10 are accessible on the website of the education portal of Baden-Württemberg: http://bildung-staerkt-menschen.de

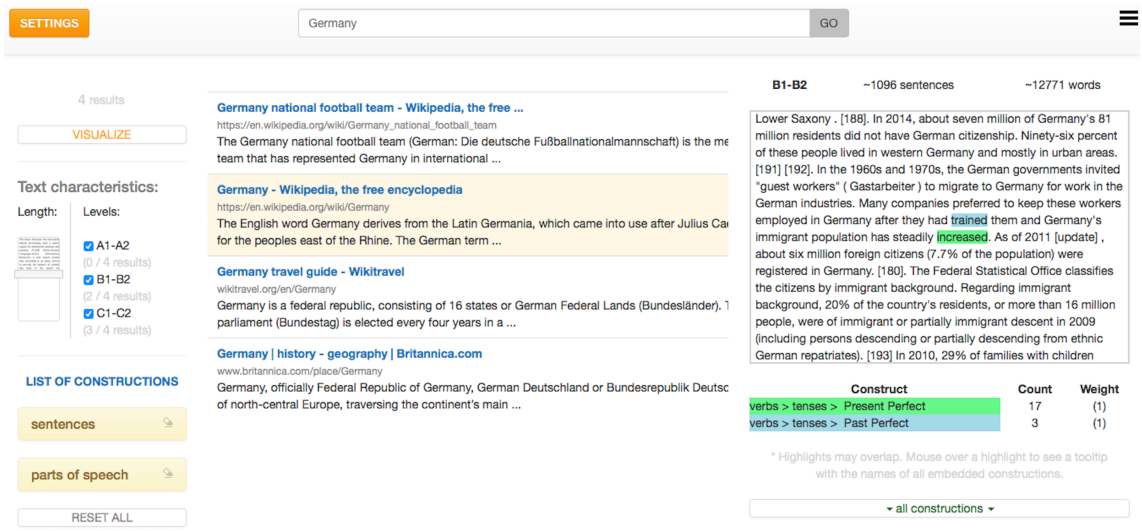

Figure 2: *FLAIR* interface: the settings panel, the list of results and the reading interface.

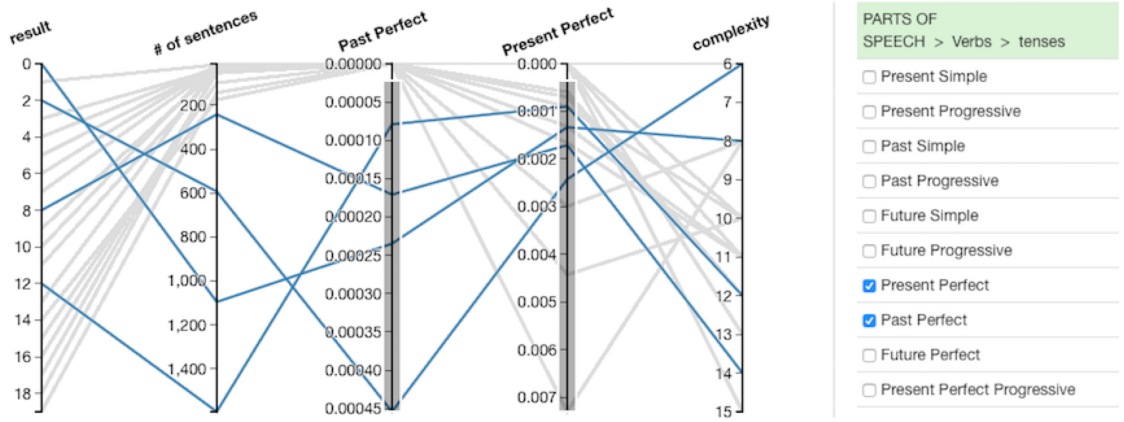

Figure 3: The visualization component of *FLAIR*. Vertical axes correspond to text characteristics and the lines going through the axes represent documents.

as *real* and *unreal conditionals* and different grammatical *tenses*, are identified by means of complex patterns and additional constraints. For a more elaborate discussion of the detection of linguistic forms, the pilot evaluation and the use cases, see Chinkina and Meurers (2016).

5 Performance Evaluation

Parallelization of the tool allowed us to reduce the overall processing time by at least a factor of 25 (e.g., 35 seconds compared to 15 minutes for top 20 results). However, due to the highly parallel nature of the system, its performance is largely dependent on the hardware on which it is deployed. Amongst the different operations performed by the pipeline, web crawling and text annotation prove to be the most time-consuming and resource-intensive tasks. Web crawling is an I/O task that is contingent on external factors such as remote network resources and bandwidth, thereby making it a potential bottleneck and also an unreliable target for profiling. We conducted several searches and calculated the relative time each operation took. It took around 50-65% of the total time (from entering the query till displaying a list of results) to fetch the results and extract the documents and around 20-30% of the total time to parse them.

The Stanford parser is responsible for text annotation operations, and its shift-reduce constituent parser offers best-in-class performance and accuracy.[8] We analyzed the performance of the parser on the constructions that our tool depends on for the detection of linguistic patterns. Among the

[8]See `http://nlp.stanford.edu/software/srparser.shtml`

biggest challenges were *gerunds* that got annotated as either nouns (*NN*) or gerunds/present participles (*VBG*). *Phrasal verbs*, such as *settle in*, also appeared to be problematic for the parser and were sometimes not presented as a single entity in the list of dependencies.

The *FLAIR* light-weight algorithm for detecting linguistic forms builds upon the results of the Stanford parser while adding negligible overhead. To evaluate it, we collected nine news articles with the average length of 39 sentences by submitting three search queries and saving the top three results for each of them. We then annotated all sentences for the 87 grammatical constructions and compared the results to the system output. Table 1 provides the precision, recall, and F-measure for selected linguistic forms identified by *FLAIR*[9].

Linguistic target	**Prec.**	**Rec.**	$\mathbf{F}_1$
Yes/no questions	1.00	1.00	1.00
Irregular verbs	1.00	0.96	0.98
used to	0.83	1.00	0.91
Phrasal verbs	1.00	0.61	0.76
Tenses (Present Simple, ...)	0.95	0.84	0.88
Conditionals (real, unreal)	0.65	0.83	0.73
Mean (81 targets)	0.94	0.90	0.91
Median (81 targets)	1.00	0.97	0.95

Table 1: Evaluating the FLAIR algorithm

As the numbers show, some constructions are easily detectable (plural irregular noun forms, e.g., *children*) while others cannot be reliably identified by the parser (conditionals). The reasons for a low performance are many-fold: the ambiguity of a construction (*real conditionals*), the unreliable output of the text extractor module (*simple sentences*) or the Stanford Parser (*-ing verb forms*), and the *FLAIR* parser module itself (*unreal conditionals*). Given the decent F-scores and our goal of covering the whole curriculum, we include all constructions into the final system – independent of their F-score. As for the effectiveness of the tool ina real-life setting, full user studies with language teachers and learners are necessary for a proper evaluation of distinctive components of *FLAIR* (see Section 7).

[9]The mean and the median are given for 81 targets because six grammatical constructions did not occur in the test set.

6 Related Work

While most of the state-of-the-art IR systems designed for language teachers and learners implement a text complexity module, they differ in how they treat vocabulary and grammar. Vocabulary models are built using either word lists (*LAWSE* by Ott and Meurers, 2011) or the data from learner models (*REAP* by Brown and Eskenazi, 2004). Grammar is given little to no attention: Bennöhr (2005) takes into account the complexity of different conjunctions in her *TextFinder* algorithm.

Distinguishing features of *FLAIR* aimed at making it usable in a real-life setting are that (i) it covers the full range of grammatical forms and categories specified in the official English curriculum for German schools, and (ii) its parallel processing model allows to efficiently retrieve, annotate and rerank 20 web documents in a matter of seconds.

7 Conclusion and Outlook

The paper presented *FLAIR* – an Information Retrieval system that uses state-of-the-art NLP tools and algorithms to maximize the number of specific linguistic forms in the top retrieved texts. It supports language teachers in their search for appropriate reading material in the following way:

- A parsing algorithm detects the 87 linguistic constructions spelled out in the official curriculum for the English language.
- Parallel processing allows to fetch and parse several documents at the same time, making the system efficient for real-life use.
- The responsive design of *FLAIR* ensures a seamless interaction with the system.

The tool offers input enrichment of online materials. In a broader context of computer-assisted language learning, it can be used to support input enhancement (e.g., *WERTi* by Meurers et al., 2010) and exercise generation (e.g., *Language Muse*^{SM} by Burstein et al., 2012).

Recent work includes the integration of the Academic Word List (Coxhead, 2000) to estimate the register of documents on-the-fly and rerank them accordingly. The option of searching for and highlighting the occurrences of words from customized vocabulary lists has also been implemented. In addition to the already available length and readability filters, we are working on the options to constrain the search space by including

support for i) search restricted to specific web domains and data sets, such as Project Gutenberg[10] or news pages, and ii) search through one's own data set. We also plan to implement and test more sophisticated text readability formulas (Vajjala and Meurers, 2014) and extend our information retrieval algorithm. Finally, a pilot online user study targeting language teachers is the first step we are taking to empirically evaluate the efficacy of the tool.

On the technical side, *FLAIR* was built from the ground up to be easily scalable and extensible. Our implementation taps the parallelizability of text parsing and distributes the task homogenously over any given hardware. While *FLAIR* presently supports the English language exclusively, its architecture enables us to add support for more languages and grammatical constructions with a minimal amount of work.

Acknowledgments

This research was funded by the LEAD Graduate School [GSC1028], a project of the Excellence Initiative of the German federal and state governments. Maria Chinkina is a doctoral student at the LEAD Graduate School.

We would also like to thank the language teachers at Fachsprachzentrum Tübingen for trying out the tool and providing valuable feedback.

References

Jasmine Bennöhr. 2005. A web-based personalised textfinder for language learners. Master's thesis, University of Edinburgh.

Jonathan Brown and Maxine Eskenazi. 2004. Retrieval of authentic documents for reader-specific lexical practice. In *InSTIL/ICALL Symposium 2004*.

Maria Chinkina and Detmar Meurers. 2016. Linguistically-aware information retrieval: Providing input enrichment for second language learners. In *Proceedings of the 11th Workshop on Innovative Use of NLP for Building Educational Applications*, San Diego, CA.

Averil Coxhead. 2000. A new academic word list. *TESOL quarterly*, 34(2):213–238.

Ian Fette and Alexey Melnikov. 2011. The websocket protocol.

Jesse James Garrett et al. 2005. Ajax: A new approach to web applications.

[10] `http://gutenberg.org`

Susan M Gass and Evangeline Marlos Varonis. 1994. Input, interaction, and second language production. *Studies in second language acquisition*, 16(03):283–302.

Alfred Inselberg and Bernard Dimsdale. 1991. Parallel coordinates. In *Human-Machine Interactive Systems*, pages 199–233. Springer.

Christian Kohlschütter, Peter Fankhauser, and Wolfgang Nejdl. 2010. Boilerplate detection using shallow text features. In *Proceedings of the third ACM international conference on Web search and data mining*, pages 441–450. ACM.

Stephen Krashen. 1977. Some issues relating to the monitor model. *On Tesol*, 77(144-158).

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60.

Niels Ott and Detmar Meurers. 2011. Information retrieval for education: Making search engines language aware. *Themes in Science and Technology Education*, 3(1-2):pp–9.

Kristen Purcell, Lee Rainie, Alan Heaps, Judy Buchanan, Linda Friedrich, Amanda Jacklin, Clara Chen, and Kathryn Zickuhr. 2012. How teens do research in the digital world. *Pew Internet & American Life Project*.

Stephen E Robertson and Steve Walker. 1994. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 232–241.

Dan I Slobin. 1985. Crosslinguistic evidence for the language-making capacity. *The crosslinguistic study of language acquisition*, 2:1157–1256.

Merrill Swain. 1985. Communicative competence: Some roles of comprehensible input and comprehensible output in its development. *Input in second language acquisition*, 15:165–179.

Martha Trahey and Lydia White. 1993. Positive evidence and preemption in the second language classroom. *Studies in second language acquisition*, 15(02):181–204.

Sowmya Vajjala and Detmar Meurers. 2014. Assessing the relative reading level of sentence pairs for text simplification. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics (EACL-14)*, Gothenburg, Sweden. Association for Computational Linguistics.