Master's Thesis

Computational Linguistics

Combining Automatic Generation of Form-based Grammar Exercises from Authentic Texts with Language Aware Text Search

Author:
Tanja Heck

Supervisors:
Prof. Dr. Detmar Meurers
Dr. Stephen Bodnar

Seminar für Sprachwissenschaft
Eberhard-Karls-Universität Tübingen

September 2021

EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN

Philosophische
Fakultät

Name:	*Heck*
Vorname:	*Tanja*
Matrikel-Nummer:	*5420771*
Adresse:	*Viktor-Renner-Straße 1-9, 72074 Tübingen*

Hiermit versichere ich, die Arbeit mit dem Titel:

Combining Automatic Generation of Form-based Grammar Exercises from Authentic Texts with Language Aware Text Search

im Rahmen der Lehrveranstaltung *Masterarbeit*

im Sommer-/~~Winter~~semester *2021* bei *Prof. Dr. Detmar Meurers*

selbständig und nur mit den in der Arbeit angegebenen Hilfsmitteln verfasst zu haben.

Mir ist bekannt, dass ich alle schriftlichen Arbeiten, die ich im Verlauf meines Studiums als Studien- oder Prüfungsleistung einreiche, selbständig verfassen muss. Zitate sowie der Gebrauch von fremden Quellen und Hilfsmitteln müssen nach den Regeln wissenschaftlicher Dokumentation von mir eindeutig gekennzeichnet werden. Ich darf fremde Texte oder Textpassagen (auch aus dem Internet) nicht als meine eigenen ausgeben.

Ein Verstoß gegen diese Grundregeln wissenschaftlichen Arbeitens gilt als Täuschungs- bzw. Betrugsversuch und zieht entsprechende Konsequenzen nach sich. In jedem Fall wird die Leistung mit **„nicht ausreichend" (5,0)** bewertet. In besonders schwerwiegenden Fällen kann der Prüfungsausschuss den Kandidaten/die Kandidatin von der Erbringung weiterer Prüfungsleistungen ausschließen (vgl. § 12 Abs. 3 der Prüfungsordnung für die Magisterstudiengänge vom 11. und 25. September 1995 bzw. § 13 Abs. 3 der Prüfungsordnung für die kulturwissenschaftlichen Bachelor- und Masterstudiengänge vom 12.10.2006 und 23.11.2007).

Datum: 15.09.2021 Unterschrift: [illegible]

Contents

Abstract

The need for automatic exercise generation has long since been acknowledged in Instructed Second Language Acquisition in order to meet the demand for supplementary practice material adapted to the learner's individual needs. While a considerable amount of research has focused on how to automatically generate form-based grammar exercises in the past years, existing approaches struggle to overcome the challenge of integrating exercises into authentic contexts that are rich in the targeted linguistic constructions. We addresses this issue by integrating our tool for exercise generation into a language aware search engine, thus assisting the task of identifying suitable documents. In addition, we incorporate a range of features including automatically generated feedback, a portable output format, context preservation and customization of the exercises. An evaluation of exercises generated from a representative sample of web documents showed that our application effectively bundles the strengths and overcomes the weaknesses of previous approaches to automatic exercise generation.

Acknowledgements

My most sincere thanks go to my supervisors Detmar Meurers and Stephen Bodnar who provided not only guidance for my work but, even more importantly, encouraging support and motivating feedback.
I am also most grateful to all those taking the time to provide assistance or professional input, including Ramón Ziai's initial presentation of H5P, Aleksandar Dimitrov's gentle introduction to VIEW, Björn Rudzewitz's invaluable insights into FeedBook's feedback generation and Jochen Saile's patient support with deployment.
As this thesis is based on existing systems for document retrieval and feedback generation, I owe my gratitude to researchers who, although some of them I have never met with, have laid the foundation for my own work.
With all my heart, I thank my sister Melanie for answering all my questions on academic writing and proof-reading this thesis; and, along with with my parents, my sister Isabella and my soon-to-be brother-in-law Riku, for being my tower of strength, always willing to lend an open ear or a strong shoulder.

List of Figures

List of Tables

Chapter 1

Introduction & Motivation

Web-based language learning resources are becoming more and more readily available, yet textbooks still constitute the primary source of language material in traditional classroom teaching (Hadjerrouit, 2010). Available practice exercises are thus usually limited to those provided by the textbook a learner follows, possibly supplemented by additional exercises offered by the instructor. Since compiling supplementary exercises is a time-consuming task, however, additional exercises are often neglected in classroom teaching. Although task repetition has been shown to be beneficial to language learning, this concept requires the tasks to be slightly modified at each repetition (Ahmadian, 2012). It is thus necessary to provide similar, but not identical, exercises which target the relevant linguistic constructions in new contexts. Contexts which cover current topics as well as the learner's interests are especially beneficial. As textbook exercises are neither personalized nor updated on a regular basis, they cannot meet this requirement. The lack of suitable exercise material can thus constitute an important bottleneck in Instructed Second Language Acquisition (SLA).

Apart from being limited in terms of numbers and diversity, textbook exercises are faced with the additional shortcoming of fixed task complexity. According to Stephen Krashen's Input Hypothesis, learning requires input at the level of *i + 1*, *i* representing the user's current level and *1* an increment indicating slightly more advanced input (Ortega, 2014). Exercise complexity which allows for optimal learning outcomes thus depends on the learner's current proficiency. Textbook exercises, however, cannot account for individual differences between learners. In order to meet a learner's individual needs, it is therefore necessary to provide exercises at different difficulty levels. Factors influencing task complexity may depend on characteristics of the text such as the used vocabulary, or else on the implementation of exercise characteristics such as the amount of information given in the instructions. The latter kind of factors allows to compile exercises of varying difficulty from the same text base.

Automated exercise generation, which applies Natural Language Processing (NLP) technology in order to compile exercises with no or minimal manual effort, addresses the issues of both exercise sparsity and exercise uniformity by making it feasible to provide large amounts of exercises at varying complexity levels. However, two main challenges need to be considered when implementing corpus-based exercise generation. (1) Items which the learner needs to provide or identify, henceforth referred to as EXERCISE TARGETS, consist in linguistic constructions. Although increasing reliability of NLP tools allows to identify target constructions with high accuracy, occasional mis-identifications cannot be prevented. It is therefore crucial to offer some editing functionality for the generated exercises in which potential errors may be removed before presenting the exercise to a learner. (2) Using arbitrary documents from the web to generate language exercises may result in large text overhead with very few exercise targets or in unsuccessful exercise generation due to a lack of relevant target constructions in the selected text. In order to ensure that the selected document contains a sufficient amount of exercise targets, exercise generation may be combined with an intelligent text search to rank the search results and filter texts that are rich in the desired constructions.

Irrespective of manual or automatic compilation, traditional form-based exercises in SLA, which focus on the formation of a specific grammar item (Willis and Willis, 2001), are usually of a mechanical nature that is little captivating to language learners (Walz, 1989). In order to boost motivation, current approaches tend to contextualize the exercises (Peacock, 1997). Contextualization is here often understood as embedding exercises in some text which does not necessarily come from an authentic source but may also be compiled for the purpose of the exercise. Yet using real-world texts, which are available abundantly on the web (Chun, 2016), has the additional advantages of improving reading skills by exposing learners to authentic native language (Wilcox et al., 1999), and fostering cultural understanding (Garcia, 2008). By preserving not only the LINGUISTIC but also the VISUAL CONTEXT in terms of the original markup of the web pages, learners are in addition provided with implicit assistance through illustrating material such as images, and through linked resources providing additional information on the topic (Romney, 2016).

Generated exercises are usually used as supplementary practice material outside of class so that no instructor is available at the time when students work on the exercises. However, immediate feedback is considered an essential aspect to successful language learning (Mackey, 2006). For long-term knowledge retention, Finn and Metcalfe (2010) found immediate, scaffolding feedback, which incrementally guides a learner towards the correct answer, to be the most successful feedback method. Feedback should thus be provided with the exercise on the target platform. By integrating automatically generated feedback into the compiled exercises, post-editing effort can be reduced to verifying the generated content instead of enriching it with additional features.

As automatic, immediate feedback requires the target platform to support the relevant functionalities, compliance between the generated exercises and the distribution platform needs to be ensured. While it is possible to host both exercise generation and exercise distribution in a single application, most language classes already use some specific Learning Management System (LMS). It is therefore preferable to make the generated exercises available in the existing e-learning platform. Various file exchange formats exist which are supported by a range of common LMSs. Such exercise formats allow to generate portable exercises which may be used outside of the exercise generation system.

Taking into account these considerations, this thesis presents a tool for exercise generation integrated into a document ranking system. The tool, which we call FLAIR$_{\text{ExGen}}$, aims at providing exercises of varying complexity with integrated feedback for learners of English at beginner and lower intermediate levels. By using a portable output format, the project aims to generate exercises which may be re-used and shared outside of the exercise generation system.

The rest of this thesis is structured as follows. The subsequent sections of this chapter introduce related work on exercise generation and existing systems for document ranking and feedback generation, as well as relevant topics and exercise types including possibilities to adjust their complexity. Chapter 2 presents the existing architecture of the base system for document ranking as well as the new feature for exercise generation. Chapter 3 describes the implementation details. Chapter 4 evaluates the project and discusses emerging issues. Chapter 5 summarizes and concludes with an outlook on future work.

1.1 Related work

The requirements we impose on our exercise generation system comprise the quality criteria outlined above. While letting users supply arbitrary input texts allows to tailor exercises to the learner's interests, integrating a document ranking mechanism ensures that the texts are suitable for exercise generation. Preserving visual in addition to linguistic context helps to quickly categorize the topic of the exercise text (Romney, 2016). In order to adjust task complexity to the learner's level, exercise settings need to be configurable. Optimal support for learners while they work on the exercises requires integrated feedback that guides the student towards the correct answer. While functionalities to modify the generated exercises may be implemented within the system, providing the tasks in a portable output format supported by an editing tool is equally suitable. Such a format is preferable at any rate in order to integrate the exercises into a LMS used by the learners.

In SLA, tools designed to preserve authentic context by providing learning material based on real-world texts are referred to as Authentic Text Intelligent Computer-Assisted Language Learning (ATICALL) systems. Most of these tools focus on input enhancement in order raise a learner's awareness for specified constructions by highlighting them. The web plugin by The Alpheios Project (2018) constitutes a reading support tool for classical languages which displays vocabulary information when the user clicks on any word of the web page. The desktop application for Macintosh **COMPASS**, with its own editing interface, also provides vocabulary information for clicked words (Breidt and Feldweg, 1997). Most other ATICALL tools do not preserve the original markup of the web page. They often constitute reading assistants which present the extracted text in an enhanced manner. **SMILLE** (Zilio et al., 2017) highlights the user-specified constructions in the selected web text. **SmartReader** does not focus on specific constructions selected by the user but provides a range of meta-information when clicking on any word (Azab et al., 2013).

Similarly, **Glosser-RuG** provides vocabulary information for clicked words (Dokter et al., 1997). Preservation of context in exercise generation tools is examined in the following.

In terms of automatic exercise generation, there has been a considerable amount of research over the past years and a range of tools to generate form-based grammar exercises exist. They can be subdivided into two categories: tools that generate simple exercise sentences using a rule-based approach and tools that extract sentences which contain the targeted constructions from existing texts.

In the first category, featuring rule-based tools, the **Mgbeg** exercise generator can create a range of exercises of different types based on a grammar specification for context-free languages (Almeida et al., 2017). **GramEx** generates Fill-in-the-Blanks (FiB) and Word ordering exercises for French based on a constraint language (Perez-Beltrachini et al., 2012). The constraints which the generated sentences must satisfy can be specified by the user. The authors highlight the suitability of the exercises especially for low proficiency levels due to the simple and controlled nature of the generated exercise contexts.

Examples in the second category, with texts based on extracted sentences, include the more vocabulary-focused mobile application **WordGap**[1]. It generates Multiple Choice (MC) exercises from English web or locally stored texts for a part of speech (POS) selected by the user in the graphical interface (Knoop and Wilske, 2013). Although the main focus is on vocabulary practice, choosing a closed class POS like prepositions allows for practice of such grammar topics. In terms of post-editing support, the tool provides neither functionalities within the system nor an interface to external editing systems. In a similar vein, VISL with its **KillerFiller**[2] extension generates FiB exercises on inflection (Bick, 2000). Texts in the 22 supported languages are taken from the system's annotated corpora. Target words are determined based on the POS selected by the user and hints in brackets are given whenever necessary for disambiguation in order to allow only a single correct answer. The tool supports post-editing of the generated exercises in the authoring tool Hot Potatoes[3]. The Tutor Assistant's **Task Generator** supports FiB and sentence formation ex-

[1]https://github.com/wordgap/wordgap
[2]https://visl.sdu.dk/visl2/killerfiller.html
[3]https://hotpot.uvic.ca/

ercises with automatic feedback for a range of grammatical constructions based on a user-supplied English text (Toole and Heift, 2001). Post-editing is possible within the Tutor Assistant. **MIRTO** provides automatically generated Mark-the-Words (MtW) and FiB exercises for texts from its corpus database or supplied by the user (Antoniadis et al., 2004; Antoniadis and Ponton, 2004). It supports roughly 10 languages and is highly configurable regarding the target forms and the specified help such as meta-information in brackets. Targets and exercise components are automatically compiled based on these settings. Tasks may be sequenced within the tool, yet post-editing of the generated exercises is not possible. The **Grammar Exercise Generator** may be used to automatically generate FiB and Single Choice (SC) exercises in 7 languages (Schwartz et al., 2004; MELERO and FONT, 2001). When using the tool within the NLP-based Game Generator, users can manually include image and game elements, but not otherwise modify the exercises. **FAST** extracts large numbers of single sentences from the web in order to convert them into MC and Error detection exercises for English (Chen et al., 2006). An editing interface is not available. **ArikIturri** does the same for Basque and also supports FiB and Word formation exercises (Aldabe et al., 2006). It uses XML formats for input and output and assumes that the linguistic constructions to target in the exercises are specified in an external assessment platform. Post-processing in order to modify the generated exercises is also outsourced to that platform. **WebExperimenter**, supporting only English, uses a machine learning approach to generate MC exercises from BBC news articles (Hoshino and Nakagawa, 2005a,b). Apart from choosing the article and the number of blanks, the user has no influence on target constructions and cannot choose between grammar and vocabulary exercises. Post-editing is not possible. In its initial version, **Sakumon** assists the user in generating an exercise from a text in its database of English web articles by offering the contained linguistic constructions as potential target words for MC or FiB exercises and providing distractor suggestions. Target words and distractors can then be manually selected from the suggestions. Hoshino and Nakagawa (2008) proposed an extension for automatic blanks generation and text upload, but no according implementation has yet been published. While the web plugin WERTi only highlights user-specified constructions, its extension **VIEW**[4] can generate MtW, FiB and SC exercises for them by directly manipulating the source HTML of the web

[4]https://addons.mozilla.org/en-US/firefox/addon/view/

page (Meurers et al., 2010). Supported languages include English, German, Spanish and Russian (Reynolds et al., 2014). Post-editing is not possible. The **ClozeFox**[5] web plugin offers similar functionalities for English and Dutch: It generates FiB and MC exercises for any visited web page (Colpaert and Sevinc, 2010). ClozeFox does not offer any post-editing functionalities, either. The web-based **Language Exercise App** generates FiB, MC and Sentence Shuffling exercises for a user-supplied text in English, Spanish, French or Basque (Naiara Perez Miguel, 2017). The exercises are highly configurable in terms of target constructions, brackets contents for FiB exercises and distractors for MC tasks. Targets may be rejected in order not to use them for the exercise, but not otherwise modified within the system. The exercises can, however, be exported in a format supported by the LMS Moodle[6]. **Lärka**[7] offers SC vocabulary and inflectional grammar exercises for Swedish at a user-specified proficiency level based on single sentences extracted from its corpus. Automatically generated feedback is included, but the generated exercises cannot be modified (Volodina et al., 2014). The web tool **COLLIE**[8] automatically generates a range of scaffolded exercises from user-supplied texts to practice French grammatical gender (Bodnar and Lyster, 2021). Supported exercise types include FiB, sorting and speaking activities. **REAP.PT** focuses on vocabulary exercises for Portuguese in a gamified environment, but also offers pronominalization, subordinate clause and passive transformation exercises (Lourenco, 2015). Feedback is generated automatically. Ferreira and Pereira Jr. (2018) present work to generate FiB, MC, True/False and Tense transposition exercises for English verbs. The implementations by EWR and Seneff (2007) and Lee et al. (2016) are both centered on generating SC exercises for English prepositions with a focus an distractor generation. They are not integrated into fully deployable systems and do not fulfill any of the other requirements we impose on our system.

Table 1.1.2 provides an overview of the presented existing work on automatic generation of grammar exercises. It highlights that, while many of these systems incorporate, at least to some extent, a selection of the characteristics we consider relevant to automatically generated, form-based grammar exercises, none of them combines the features in their entirety. Especially the selection of suitable documents is hardly

[5]https://wiki.mozilla.org/Education/Projects/JetpackForLearning/Profiles/ClozeFox
[6]https://moodle.org
[7]https://spraakbanken.gu.se/larkalabb/
[8]https://www.collietool.ca/

	Document ranking	Arbitrary text input	Configurable settings	Authentic context	Feedback generation	Post-editing support	Portable output
Mgbeg							
GramEx			(●)				
WordGap		●	(●)				
KillerFiller			(●)			●	●
Task Generator		●	(●)	(●)	●	●	
MIRTO		●	●	(●)	●		(●)
Grammar Exercise Generator		●		(●)			
FAST			(●)				
ArikIturri		●			●	●	●
WebExperimenter			(●)	(●)			
Sakumon	●	(●)	●	(●)		●	
VIEW		●	(●)	●			●
ClozeFox		●	(●)	●	●		●
Language Exercise App		●	●	(●)	(●)	(●)	
Lärka	(●)		(●)		●	●	●
COLLIE		●		(●)			
REAP.PT	(●)				●		
Ferreira and Pereira Jr. (2018)		●					
EWR and Seneff (2007)					(●)		
Lee et al. (2016)							

Table 1.1.2: Exercise generation systems

Requirements marked with a dot are fulfilled by the tool. Dots in parenthesis indicate requirements that are only partially fulfilled.

targeted at all. Support for automatically generated feedback, post-editing of the generated exercises and portable output formats varies considerably from system to system. The user often has only rudimentary influence on the properties of the generated exercises, although many systems allow to base the exercises on custom texts. Some ATICALL systems exist, yet only few preserve the entire HTML context (e.g. Meurers et al., 2010; Colpaert and Sevinc, 2010). Instead, the focus is usually on maintaining linguistic rather than visual authenticity. This is not surprising considering that practice exercises primarily target learners at beginner and lower intermediate levels who still need guidance in their learning process. The more advanced and thus more independent a learner is, the more relevant autonomously chosen reading material becomes. For this kind of language input, maintaining visual authenticity of (enhanced) texts is more important than in the context of practice exercises provided to the learner by an instructor. However, resources such as images or videos often have a decorative effect for more impact or a less intimidating first impression of the exercises. In addition, they can have supportive effects to introduce the exercise context at a glance and thus activate the learner's relevant existing knowledge (Romney, 2016). Our exercise generation component therefore aims to preserve as much context as possible, including HTML markup such as links and resources like images, while at the same time accepting minor deficiencies in the rendering which may arise from technical challenges.

Most of the exercise generation tools provide the generated exercises within the system and do not offer any export functionalities. Noticeable exceptions include KillerFiller, MIRTO and the web plugins VIEW and ClozeFox. MIRTO provides an export functionality generating a file in Moodle CLOZE syntax which can then be uploaded to that e-learning platform. However, it only supports Cloze exercises. KillerFiller uses a XML output format supported by Hot Potatoes. Since this authoring tool for exercises provides output compliant with the e-learning standard SCORM[9], it is portable in nature and allows integrating the generated exercises into any LMS that supports this standard, such as Moodle. VIEW and ClozeFox are implemented as web plugins for Mozilla Firefox so that they can be applied to any web site, thus ensuring maximum portability. Although VIEW does support building a learner model, the database is not connected to a LMS, so that the collected data cannot be integrated into already existing learner models. ClozeFox does not support a learner model at all. While both projects are open source, thus allowing for modification and extension to support additional exercise types and feedback, generated exercises cannot be post-edited. In addition, exercises are only accessible as long as the source web site exists and contains the content intended for the exercises.

Since post-editing is a hard requirement in our project, an exercise format based on

[9]https://adlnet.gov/projects/scorm

e-learning standards is most suitable for our purposes. A set of such standards have been developed to facilitate sharing content between LMSs (Bianco et al., 2004). The most commonly applied standards and specifications include SCORM, xAPI[10] and cmi5[11]. SCORM is based on a range of older standards such as AICC[12] and LOM[13] (Bohl et al., 2002). While it is still the most widely used e-learning specification (Papazoglakis, 2013), new standards are constantly being developed. xAPI covers only a fraction of SCORM's spectrum, yet its extension cmi5 is considered SCORM's more powerful successor (Victor and Werkenthin, 2016). Authoring tools to compile content compliant with those standards are abundant (e.g. Capterra, 2021; eLearning Industry, 2021).

However, manually compiling a package, usually based on XML definitions and additional resources such as images, is cumbersome. The H5P[14] (HTML5 Package) project was initiated to address this issue by providing libraries to easily create a range of exercise types, so-called CONTENT TYPES (Joubel, 2016). Table 1.1.4 summarizes how H5P combines the strengths of applying e-learning standards and using plugins by making post-editing possible and at the same time keeping the exercise files manageable. Although specifying exercise content is highly simplified in H5P, the open source framework also integrates its own authoring tool available for all content types. Users with low technical affinity can therefore generate and manipulate exercises in a graphical interface. Implemented standards such as xAPI and LTI[15] within the H5P framework increase compatibility with other systems (Joubel, 2018, n.d.b) and allow for integration of H5P packages via interfaces. Side projects by private contributors such as the H5P SCORM Packager by Rettig (2019) to convert a H5P package into a SCORM package further increase the range of compatible platforms. H5P is thus supported by numerous popular LMSs including Canvas[16], Blackboard[17] and Moodle (Joubel, 2021). An additional advantage of using this exercise format consists in H5P's support of xAPI specifications which allows communication with the host system. Most content types thus already provide data to

	LMS integration	Post-editing support	Manageability
Plugins	(●)		●
E-learning standards	●	●	
H5P	●	●	●

Table 1.1.4: Portable exercise formats

Features marked with a dot are realized by the format. Dots in parentheses indicate features that are realized only to some degree.

maintain the learner model on the distribution platform.

1.2 Document ranking systems

Tools providing intelligent text search often base their filter algorithm on learner-specific parameters. The **REAP** system selects documents from the web suitable for the student's proficiency level based on the learner model (Brown and Eskenazi, 2004). An extension to this system generates vocabulary questions for the selected texts, thus making sure that the targeted vocabulary is at the learner's proficiency level (Brown et al., 2005). **TextFinder** also matches the learner's proficiency determined from the learner model, which is generated from previous interaction of the student with the system, against readability scores calculated for the documents in its database of online news articles in order to select the most suitable one (Bennöhr, 2005). The web extension **LAWSE** can analyze arbitrary web texts and provides the calculated readability scores without relating them to the learner's proficiency (Ott and Meurers, 2011). The standalone tool **READ-X** filters web results according to a learner-specified reading level ranging from grades 1 through 12 to college level (Miltsakaki and Troutt, 2011). **SourceFinder** targets university students by identifying texts at advanced reading levels (Sheehan et al., 2007). It currently only operates on a pre-selected corpus of online journals. While these

[10]https://github.com/adlnet/xAPI-Spec
[11]https://aicc.github.io/CMI-5_Spec_Current/
[12]https://github.com/ADL-AICC/AICC-Document-Archive/
[13]https://standards.ieee.org/standard/1484_12_1-2020.html
[14]https://h5p.org/
[15]https://www.imsglobal.org/activity/learning-tools-interoperability
[16]https://www.instructure.com/product/higher-education/canvas-lms
[17]https://intelliboard.net/

tools do not or only marginally take grammatical features into account, the authoring assistance tool **Sakumon** maintains information on the article's reading level as well as on contained grammatical constructions in its database (Hoshino and Nakagawa, 2008). Filtering functionalities help the user to identify articles featuring those characteristics relevant for the exercises to be generated.

FLAIR[18] (Form-Focused Linguistically Aware Information Retrieval) constitutes a language aware search engine that combines web search with custom configurable linguistic construction weighting (Chinkina et al., 2016). Supported weight options encompass all 87 grammatical constructions covered by the English curriculum of high schools in Baden-Württemberg (Chinkina et al., 2016). FLAIR thus provides a selection of documents rich in and ranked according to the number of contained targeted constructions. This kind of re-ranking of search results filters suitable texts while at the same time not restricting the user to a single suggested document. We therefore choose this tool to pre-select texts that contain a sufficient amount of exercise targets. We then apply our exercise generation functionality to the extracted set of documents.

While FLAIR addresses the challenge of selecting texts rich in target constructions, the issue of reliability in detecting those constructions remains. Some linguistic constructions can be identified correctly with F1-scores of up to 1.0, yet Chinkina et al. (2016) report only moderate f-scores of .88 and .73 for identifying tenses and conditionals respectively, two of the central topics for lower-level learners. Providing an authoring interface to post-process the generated exercises is therefore essential.

1.3 Feedback generation systems

Early approaches to automatic feedback generation rely on **offline** compilation of feedback for possible learner answers, so-called TARGET HYPOTHESES. At runtime, the most suitable hypothesis is determined through pattern matching of the learner's answer to the pre-compiled target hypotheses. Although it is employed in a range of tools (e.g. Feuerman et al., 1987; Levison et al., 2000; Paramskas, 2013), the large amount of target hypotheses and often rigid matching strategies make this approach less and less feasible and reliable the more complex and diverse the tasks are (Faltin,

[18]http://sifnos.sfs.uni-tuebingen.de/FLAIR/

2003).

With the increasing spread of more and more reliable NLP tools, the focus of feedback generation has shifted towards **online** analysis of learner answers (e.g. Nagata, 1995; Yang and Akahori, 1998; Reuer, 2003; Delmonte, 2003; Heift, 2003; Amaral, 2007; Dickinson and Herring, 2008; Nagata, 2009, 2011; Dansuwan et al., 2013; Heift, 2015). Since standard NLP tools are designed to process correctly formed language, however, analyzing learner language requires modifying those tools (Meurers, 2012). Constraint relaxation approaches have been widely applied (Weischedel and Black, 1980; Heinecke et al., 1998; L'haire and Faltin, 2003). They modify the parser in order to allow parsing a sentence even if some linguistic constraints such as agreement are not fulfilled. Since each constraint relaxation is associated with a particular error, the nature of the learner's error is also determined in the process. A second approach introduces manually specified mal-rules to anticipate ungrammatical productions, thus allowing to parse them (Schneider and McCoy, 1998; Crysmann et al., 2008). While such approaches enable standard NLP tools to analyze learner language, they do not take into account the task context. The nature of the error is therefore often mis-diagnosed, thus making it difficult to provide helpful feedback.

Hybrid approaches that combine online and offline analysis allow to take into account the task context by deriving target hypotheses from the correct answer, while at the same time using NLP tools for more flexible matching of learner answers and target hypotheses. L'haire and Faltin (2003)'s FreeText system uses a syntactic parser which employs constraint relaxation. The authors report improved results when adding a sentence comparison component which generates ill-formed target hypotheses.

The feedback strategy applied in Rudzewitz et al. (2018)'s **FeedBook**[19] clearly separates online from offline processing. For offline processing, target hypotheses are generated from the correct answer. By pre-estimating the space of possible target answers, the feedback generation mechanism shifts the processing weight from latency-critical online processing to offline feedback generation on high performance servers. The online feedback generation then takes the task context into account and allows for flexible matching of learner input to target hypotheses. FeedBook supports both form-based and semantic exercises for a range of topics of the 7th

[19]http://feedbook.website/

grade curriculum for English of high schools in Baden-Württemberg (Rudzewitz et al., 2018). Possible exercise types include Short Answers, FiB, Mapping of text or images and MC exercises (Rudzewitz et al., 2017). Although the current implementation does not allow for external applications to use the feedback generation functionality, an ongoing project aims at modularizing the application and providing its central functionalities as microservices. We thus integrate this microservice to provide scaffolding feedback with the generated exercises.

1.4 Exercise forms

At the most basic level, exercises are defined through an exercise type as well as an exercise topic. Their range, as well as supported type-topic combinations, are determined by the application.

1.4.1 Exercise types

We considered 8 potentially suitable H5P content types for form-based ATICALL exercises:

1 True/False Questions ask a learner to decide whether a sentence is correct or test the learner's knowledge about linguistic meta-information. The former has the disadvantage of not emphasizing the correct answer, except in potentially specified feedback with additional information. The latter interrupts the text flow and is thus not suitable for exercises embedded in authentic contexts. We therefore did not consider this content type suitable for our system.

2 Single Choice exercises ask a learner to choose the correct form from a list of options. While they are often referred to as *Multiple Choice* exercises, H5P does make the distinction between SC with only a single correct answer and MC where multiple options may be correct. However, the H5P implementation does not intend the exercises to be embedded into a text. Using this type in such a scenario would therefore interrupt the text flow. We therefore did not include it in FLAIR$_{\text{ExGen}}$.

3 Multiple Choice exercises represent MC exercises. Although it is possible to mark only a single answer as the correct solution, this content type is intended for questions with multiple correct answers. In the context of form-based grammar exercises, however, there usually is exactly one correct form. Moreover, determining whether multiple forms might be correct would defeat the purpose of basing the exercises on authentic texts that inherently provide the correct answer. In addition, this type also faces the issue concerning integration into the text flow. We thus excluded it from further consideration.

4 Fill in the Blanks exercises implement one of the most common exercise types in which the learner needs to enter a more or less complex form into the provided slots called BLANKS. Each blank constitutes a placeholder for a linguistic construction. Hints are usually given in brackets following a blank. This H5P content type provides only rudimentary feedback functionalities. In fact, the only means for guiding the user towards the correct answer consists in static hints which are displayed permanently and provide the same help text irrespective of the learner's answer. We did not further consider this content type.

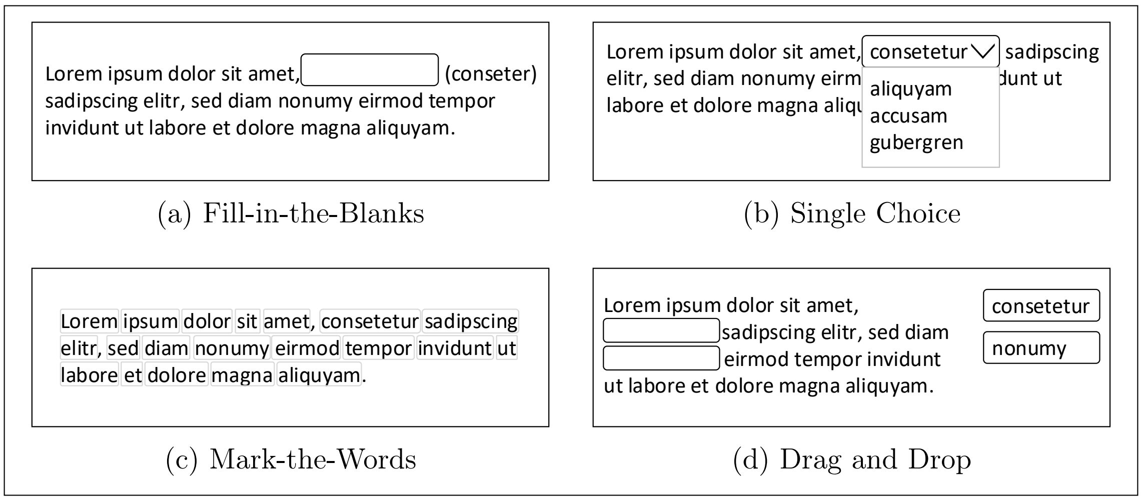
5 Advanced fill the blanks exercises combine two task types: FiB and SC exercises. The latter is implemented with dropdown fields, thus well suited to be integrated into the document text. Feedback is improved compared to `Fill in the Blanks exercises` and incorporates fuzzy matching. We therefore considered this content type suitable for our tool.

6 Essay exercises accept longer input texts in response to a question and are thus well suited for input requiring entire clauses or sentences. However, `Advanced fill the blanks exercises` automatically adjust the blank size to the length of the target so that they may also be applied to entire sentences and clauses. They also have the additional advantage of being integrated into the text flow which is not the case for `Essay exercise`. We therefore discarded this content type.

7 Drag the Words exercises represent Drag and Drop (DD) tasks with interactive, draggable components. They provide a bag of DRAGGABLE constructions and a text with empty slots called DROPPABLES. The learner needs to solve the exercise by moving each draggable into its corresponding droppable. All draggables of the text are accumulated in a single bag of words, thus not supporting sub-exercises within the same document. We nevertheless considered this content type worth pursuing for FLAIR$_{\text{ExGen}}$.

8 Mark the Words exercises provide a text in which all words constitute interactive items. The learner has to identify the occurrences of given target constructions within the text by clicking on them. Binary feedback is given in the form of colour highlighting. This content type is well suited for our application.

Figure 1.4.1 illustrates the four exercise types we chose for FLAIR$_{\text{ExGen}}$: FiB, SC, MtW and DD exercises. While none of the content types that we considered suitable fulfills all our requirements, they constitute sophisticated base types from which to derive custom content types adapted to our needs. It is also possible to develop new content types from scratch. However, modifying existing ones reduces implementation effort considerably and provides established and well-tested functionalities.


Lorem ipsum dolor sit amet, (conseter) sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam.
Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirm labore et dolore magna aliqu
aliquyam
accusam
gubergren
Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam.
Lorem ipsum dolor sit amet, sadipscing elitr, sed diam eirmod tempor invidunt ut labore et dolore magna aliquyam.
consetetur
nonumy

(a) Fill-in-the-Blanks (b) Single Choice

(c) Mark-the-Words (d) Drag and Drop

Figure 1.4.1: Exercise types

Especially in scenarios where only part of a document is used for one exercise and another part of the same document for another exercise type or configuration, it makes sense to bundle those exercises. H5P offers two sequencing content types:

1 Interactive Books combine multiple exercises and other contents like interactive videos or presentations. They support all out-of-the-box content types, but no custom content types. Since we only have exercises, other contents are not relevant to our purposes. We therefore discarded this type.

2 Quizzes combine multiple content types. Only 7 types are supported, not including `Advanced fill the blanks` or any custom content types. We still considered this type for our application with the aim of deriving a custom content type that suits our purposes.

1.4.2 Exercise topics

Practice exercises are most important at beginner and (lower) intermediate levels in order to develop automated, fluent language (Meurers et al., 2019). We therefore focus on frequent language targets outlined by the Baden-Württemberg Ministerium für Kultus (2016) for the first years of instruction of English with a focus on 7th and 8th grade. Since we aim to integrate FLAIR's document ranking as well as FeedBook's feedback generation, we consider a subset of the curriculum's topics which are supported by both tools: tenses, conditional sentences, relative pronouns, comparison and passive.

1 Tenses With respect to ***simple present*** tense, the use of the third person singular *-s* presents a challenge to many learners. Simple MtW exercises may be employed to raise awareness of the use of the suffix. The learner can either be asked to mark all present tense verbs or only those in third person singular. For more advanced practicing, the most useful exercise types constitute FiB exercises where the learner needs to give the correct present tense form for a given lemma, as well as SC exercises which ask the learner to decide between the forms with and without the suffix.
Past tenses constitute another relevant topic in this category. The regular *-ed* suffix and irregular verb forms, as well as determining the required tense need practicing. MtW exercises might again serve as introductory exercises to identify the indicated verbs. FiB exercises with the lemma given for each blank are also relevant. Similarly to SC exercises, they can be used to practice correct formation and tense determination. Reasonable distractors for SC exercises thus include irregularly and regularly formed variations of the verb and forms in other tenses. DD exercises are most instructive when multiple tenses are practiced, requiring the learner to determine the correct tense. However, they may also be used for semantic training when focusing on only one tense.

2 Conditional sentences Conditional sentences need practice both in isolation, targeting only real or only unreal conditionals at a time, as well as in combination. FiB exercises make sense in a variety of constellations: Introducing blanks only for conditional clauses, only for main clauses, for either clause randomly selected per sentence, or for both clauses. Brackets can contain hints about the required tense, mood and lemma. SC exercises may follow the same pattern, offering distractors in variations of mood and tense. DD exercises for the verbs of the two clauses could also prove valuable if semantic support should be allowed in addition to grammatical evidence. By increasing the scope of a DD exercise so as to encompass the entire text, the challenge will become of a more semantic nature.

3 Relative pronouns With relative clauses, both choosing the correct relative pronoun and word order formation need practicing. As an introduction to the topic, MtW exercises are again useful to teach the learner to recognize relative pronouns. In order to practice pronoun usage, FiB exercises are relevant as well. For SC exercises, reasonable distractors include the three most common relative pronouns *who*, *which* and *that* as well as occurrences of other relative pronouns in the same document. DD exercises operating on the entire text might also be useful, provided the text contains enough different relative pronouns to make assigning them to the correct relative clause reasonably challenging. In order to practice correct word order, DD exercises will be the most useful exercise type when applied to one relative clause at a time. Draggable elements should include the pronoun, the verb, the subject, object(s), and the rest of the sentence.

4 Comparison Comparative and superlative forms of adjectives and adverbs are particularly challenging in the use of an analytic form as opposed to a synthetic form (Parrott, 2010, p. 89). MtW exercises are, once more, a good starting point as they help learners to recognize comparison forms. SC exercises and FiB exercises are both suitable choices to focus on the distinction between analytic and synthetic forms. While SC exercises offer both alternatives as possible choices, FiB tasks require learners to produce the correct form themselves. DD exercises will pose grammatical as well as semantic demands on the learner as some draggable items may have similar meta-linguistic properties. The correct form then needs to be selected from the grammatically possible draggables based on the semantic context.

5 Passive Since passive constructions affect the structure of the entire sentence, FiB exercises also need to offer blanks spanning complete sentences. Brackets will then contain the active counterpart of the passive sentence. An additional variant where the lemmatized agent, verb and patient are given in brackets might also be considered. Leaving a blank only for the verb constitutes yet another alternative. DD exercises may be used to practice correct word order of passive sentences, offering the subject, the verb, and the by-clause as draggables.

Practice exercises for these exercise topics thus constitute form-based grammar exercises. Table 1.4.2 illustrates that all four examined exercise types are widely applicable to the proposed topics and therefore relevant to FLAIR$_{\text{ExGen}}$.

	Fill-in-the-Blanks	Single Choice	Drag and Drop	Mark-the-Words
Simple present	●	●		●
Past tenses	●	●	●	●
Conditional	●	●	●	
Relative clauses	●	●	●	●
Comparison	●	●	●	●
Passive	●		●	

Table 1.4.2: Exercise types per topic

Each row represents a particular exercise topic, each column an exercise type. The dots indicate type-topic combinations for which FLAIR$_{\text{ExGen}}$ offers exercise generation support.

1.5 Exercise complexity

Current literature often distinguishes between task COMPLEXITY and task DIFFICULTY (Robinson, 2001). While the former relates to task-specific characteristics, the latter encompasses learner-specific properties like motivation. However, we do not make such a distinction in this thesis. The two terms will henceforth be used interchangeably with the core meaning of complexity. In a similar vein, EXERCISES are usually understood as form-based practice units whereas TASKS focus on content and meaning (Xavier, 2007). In the following, we use both terms to refer to the notion of thus defined exercises.

Estimating the complexity of an exercise is particularly important in the context of Intelligent Tutoring Systems (ITS) where the task of providing exercises at the learner's proficiency level resides with the system. Nunan (1989) outlines the importance of both grammatical and general text complexity measures such as length, propositional density or vocabulary frequency and highlights that estimating task difficulty is challenging not only due to the number of influencing factors but also because of interaction effects between them. Liu and Li (2012) identify a range of factors influencing complexity ranging from input-dependent characteristics such as clarity and diversity of the material to time-related factors like pressure put on the learner by the task setting. These factors are defined in a very general manner and need to be applied to the domain of the task in order to derive relevant characteristics. Conejo et al. (2014) outline three strategies to approach the challenge of determining task complexity. (1) Statistical methods rely on data from previous attempts of learners at solving the exercise. (2) Heuristic methods require a human expert to rate the exercises. (3) Finally, mathematical methods calculate difficulty based on some characteristics of the exercise. The set of characteristics and the formula for translating them into a difficulty score depends on the task domain and the individual implementation. Yet although Robinson et al. (1995) have long since recognized the need to define valid criteria to determine task complexity, necessary research especially concerning the complexity of exercise targets instead of the entire text context is still scarce.

For language exercises, studies related to difficulty estimation are centered on vocabulary and reading exercises. They typically use mathematical approaches (Chen and Hsu, 2008; Chen and Chung, 2008; Heilman et al., 2010; Beinborn, 2016) or combine them with a statistical approach to gradually adjust an initial score based on exercise characteristics as more and more learner data becomes available (Wu and Sung, 2017; Sandberg et al., 2014). The approaches by Pandarova et al. (2019) and Hoshino and Nakagawa (2010) differ in that exercise characteristics are not used as input to a scoring formula but to a machine learning model trained to distinguish between 'easy' and 'difficult' exercises.

For the purpose of determining the difficulty of an automatically generated exercise for an arbitrary text at compile-time, neither statistical nor heuristic approaches are applicable as the exercises have yet to be presented to students and no human evaluator is involved in the generation process. As for mathematical or machine learning methods, they are usually applied to exercises after they have been generated. It thus makes sense to also take into account text complexity measures computed on the entire textual context (Pelánek et al., 2021). However, only considering those characteristics will yield identical complexity values for any exercises compiled from the same document, regardless of the exercise type or exercise components such as instructions or distractors. Moreover, such strategies do not allow to influence the difficulty of exercises but merely to select tasks at an appropriate level from an existing pool of exercises.

In order to allow for users to influence the difficulty of an exercise for a given text it is thus necessary to supply means to configure exercise parameters. Text complexity measures determining the overall complexity of a document are already considered in the document ranking. The configurable parameters therefore do not refer to characteristics of the context but of the exercise targets. The configurable parameters depend both on the topic and on the type of the generated exercise.

1.5.1 Complexity of exercise types

Some difficulty parameters are inherent to the exercise type and apply to all exercises irrespective of the practiced grammar topic. To the best of our knowledge, published research on the relative complexity of individual task types is limited to a comparison of FiB and MC exercises (Medawela et al., 2017) which found the former to be more challenging. The study was, however, not specific to language exercises.

Mark-the-Words We generally assume that MtW exercises, where the learner only needs to identify tokens with certain meta-linguistic characteristics, are the easiest of our four exercise types. Specifying the number of target constructions in the text will provide the learner with additional information, thus reducing difficulty. We suggest that MtW exercises become more difficult when the learner needs to make more fine-grained meta-linguistic distinctions in order to determine whether a token is a target word. As this increases the number of concepts required to solve the exercise, complexity increases (Pelánek et al., 2021). We therefore assume that targeting all forms of a topic results in easier exercises than targeting only a subsample which is meta-linguistically more constrained. However, research needs yet to confirm that this hypothesis applies to linguistic grammar exercises.

Drag and Drop DD exercises could also be represented as SC exercises where each droppable field contains the values of the other draggable items as distractors. They can thus be considered to be similar in complexity to SC exercises. Both exercise types provide the correct answer as one of several options so that the challenge resides in choosing the correct form from a limited range of alternatives. Guessing the correct answer is thus possible with a probability depending on the amount of alternatives (Medawela et al., 2017). Since all draggables need to be inserted into a droppable slot, difficult exercise targets might be determined by first solving the easier items. As such a process of elimination is not possible with SC exercises, DD tasks might be considered slightly less challenging.

Single Choice In addition to the characteristics outlined for DD exercises, we generally assume that using variants which vary in a larger number of features, including well-formed and ill-formed variations, increases task complexity as the learner needs to consider a greater amount of concepts. However, if distractors are too unlikely, they will be easily discarded by learners (Hoshino, 2013). This assumption must therefore be applied with caution.

Fill-in-the-Blanks For FiB exercises, the learner needs to actually produce language. The solution space is not restricted, making this the most challenging exercise type. Specifying meta-linguistic information in brackets provides the learner with hints, thus reducing task difficulty. Giving certain characteristics such as the lemmas of all target constructions in the task description instead of in brackets introduces an additional semantic challenge to the exercise, thus increasing complexity. Example 1 demonstrates how providing the lemmas of all targets in the task description requires the learner to identify the correct slot in addition to determining the correct form, thus making the exercise more challenging.

(1) a. **Give the correct simple present form for the verbs in brackets.**
My brother _________ (eat) ice cream all the time. He _________ (love) it.

b. **Give the correct simple present form. Use one of the following verbs for each blank:** *eat*, *love*
My brother _________ ice cream all the time. He _________ it.

Depending on other exercise parameters, a generally more difficult exercise type might still be considered easier than a generally less difficult type. Example 2 illustrates that, in order to practice the correct formation of comparison forms, a FiB exercise may specify the required POS, comparison level and comparison form (Example 2a). This can be perceived as easier than the SC exercise in Example 2b. Although possible answers are already given, the learner is not provided with any meta-information on the correct POS, comparison level or comparison form.

(2) a. **Give the correct comparative form for the adjectives in brackets.**
Peter is _________ (popular, *analytic*) than Max. He is _________ (nice, *synthetic*).

b. **Select the correct comparison form from the given alternatives.**
Peter is _________ (popularer | popularlier | more popular | more popularly | popularest | popularliest | most popular | most popularly) than Max. He is _________ (nicer | nicelier | more nice | more nicely | nicest | niceliest | most nice | most nicely).

1.5.2 Topic-dependent complexity parameters

Depending on the grammar topic, difficulty parameters include variables which restrict the practiced target forms. They may apply to all or only a sub-sample of the supported exercise types.

Tenses For FiB and SC exercises, complexity can be varied by optionally including negated sentences and/or questions. Since these sentence types require knowledge of additional grammar topics, excluding them decreases exercise difficulty. With past tenses, restricting the range of possible tenses and focusing on only regular or irregular verbs instead of including both may also be used to adjust difficulty. If multiple past tenses are included, specifying the correct tense in brackets for FiB exercises will reduce task complexity. Exercise difficulty for DD as well as MtW exercises may be influenced by including only regular or irregular verbs or both.

Conditional sentences Focusing on only one conditional type or only one clause type per FiB, SC or document-spanning DD exercise reduces task difficulty. Specifying the required tense and/or mood in the brackets of FiB exercises will additionally decrease complexity.

Relative pronouns Since relative pronouns constitute a closed-class category, FiB exercises are comparable to SC exercises with the three most common relative pronouns *who*, *which* and *that* as distractors. Further narrowing down possible selection options, either as distractors for SC or as allowed target forms specified in the task description for FiB exercises, can be used to decrease exercise difficulty.

Comparison Variable parameters include the POS (adjectives and adverbs), the comparison level (comparatives and superlatives) and the comparison form (synthetic and analytic forms). By restricting the exercises to use only one of the respective forms, exercise complexity may be adjusted for all exercise types. Giving the required form as meta-information in brackets of FiB exercises or limiting the distractors of SC exercises to contain only forms matching one of the respective characteristics also results in variations of exercise difficulty.

Passive Variation in complexity can be introduced by including only passive or also active sentences and by restricting the possible tenses of target sentences. FiB exercises requiring the learner to give the entire passive sentence may be considered more difficult than those leaving blanks only for the verb. Brackets specifying the tense and whether it is an active or passive sentence will reduce difficulty. Complexity for DD exercises might be adjusted by either keeping the verb cluster as a single draggable element or splitting it into the form of *be* and the past participle, as well as by restricting the possible tenses of target sentences.

Chapter 2

System Architecture

We integrated FLAIR$_{\text{ExGen}}$ into the existing FLAIR project. While exercise generation itself is handled entirely within this system, the implementation of the new feature also needs to consider interaction with LMS systems, which are intended as deployment platforms for the generated exercises.

2.1 FLAIR base system

FLAIR's existing implementation provides input enhancement in the form of colour highlighting for linguistic constructions whose weights the user has increased. Apart from using the integrated web search, users may also upload and rank their own texts.

The implementation of the web application is based on the Google Web Toolkit (GWT)[1]. The toolkit provides libraries and an infrastructure to develop both server and client components of a web application in Java. The project is thus based on a client-server architecture with the main components as illustrated in Figure 2.1.1. The client implements the Model View Presenter (MVP) pattern: The View defines user interface (UI) components for user interactions, thus determining the visual design presented to the user; the Presenter implements the logic which populates the View with dynamically compiled values; the Model maintains the data on which the Presenter operates. FLAIR's View components are made up of GWT widgets.

[1]http://www.gwtproject.org/

Static components can be declared in XML definitions which may be referenced in the code through bindings. Dynamic components which need to be generated based on user interactions can be specified directly in the code. The Presenter mediates between the View and the Model, managing dynamic visualizations and also triggering communication with the server. Responses from the server are then used to update the Model and the View. Each of these components is in turn modularized into different functionalities. The main functionality consists in the web search and subsequent ranking of the documents. In the View, this module maintains the `Search Widget` as well as the `Ranking Configuration UI` to adjust the weights used to prioritize documents containing certain linguistic constructions in the ranking. On the Presenter level, the module forwards the search term to the server, stores the returned documents in the Model, re-ranks them according to the construction weights and prepares the document previews for the View.

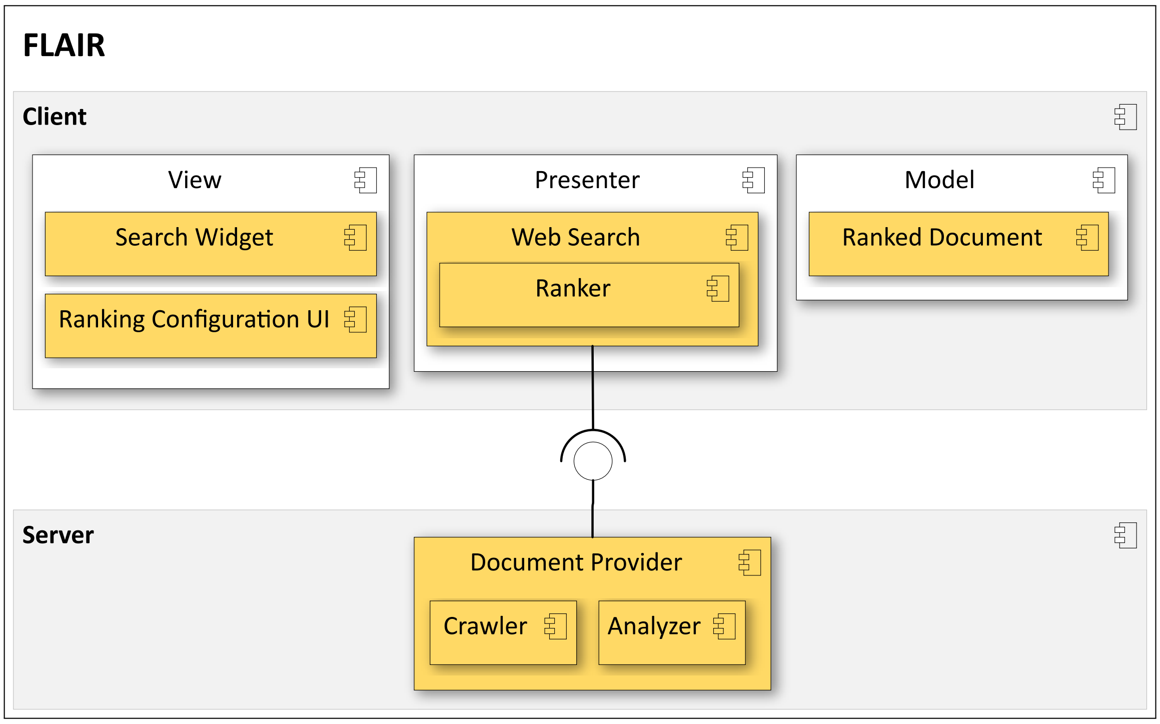


Figure 2.1.1: UML Component Diagram of the system architecture

FLAIR uses a highly modularized client-server architecture (*gray*) with a MVP structure (*white*) on the client side. The main module (*orange*) performs the web search and ranks the documents.

The server hosts all resource consuming operations. This includes primarily the

`Crawler`'s document retrieval and the `Analyzer`'s NLP analyses. The search functionality uses the Bing Web Search API[2] to identify documents that are relevant for the requested topic. The plain texts are then extracted from the retrieved documents with the highly efficient Boilerpipe[3] library (Chinkina et al., 2016). This algorithm extracts the main content of a web page, excluding boilerplate elements such as navigation and advertisements (Kohlschütter et al., 2010). To this purpose, it analyzes not only a range of HTML elements but also shallow text features such as average word and sentence lengths, and text density in terms of tokens per line. The thus extracted plain texts serve as input to FLAIR's NLP analysis which extracts all supported linguistic constructions. Constructions are identified based on POS tags, constituency trees and dependency relations. Linguistic annotations consisting of the start and end indices of the extracted occurrences within the plain text and their construction types are returned to the client along with the document plain text.

2.2 Exercise generation functionality

Thanks to FLAIR's modular structure, the new feature for exercise generation can be integrated as a mostly independent module with only minor changes to existing modules. This leads to the extended architecture depicted in Figure 2.2.1, where the exercise generation component operates without interfering with the existing modules.

Exercise generation comes into play after the documents have been retrieved and ranked. On the client side, the Presenter uses the linguistic annotations to dynamically display the `Exercise Configuration UI`, in which users can specify exercise settings, in such a way that it only offers those options which are applicable to the selected text. It then sends completed exercise configurations to the server which generates and returns an exercise based on those settings. The client offers the exercise file to the user for download.

On the server side, two new components are introduced. The first component downloads the document with preserved HTML markup. The second component gen-

[2]https://www.microsoft.com/en-us/bing/apis/bing-web-search-api
[3]https://github.com/kohlschutter/boilerpipe

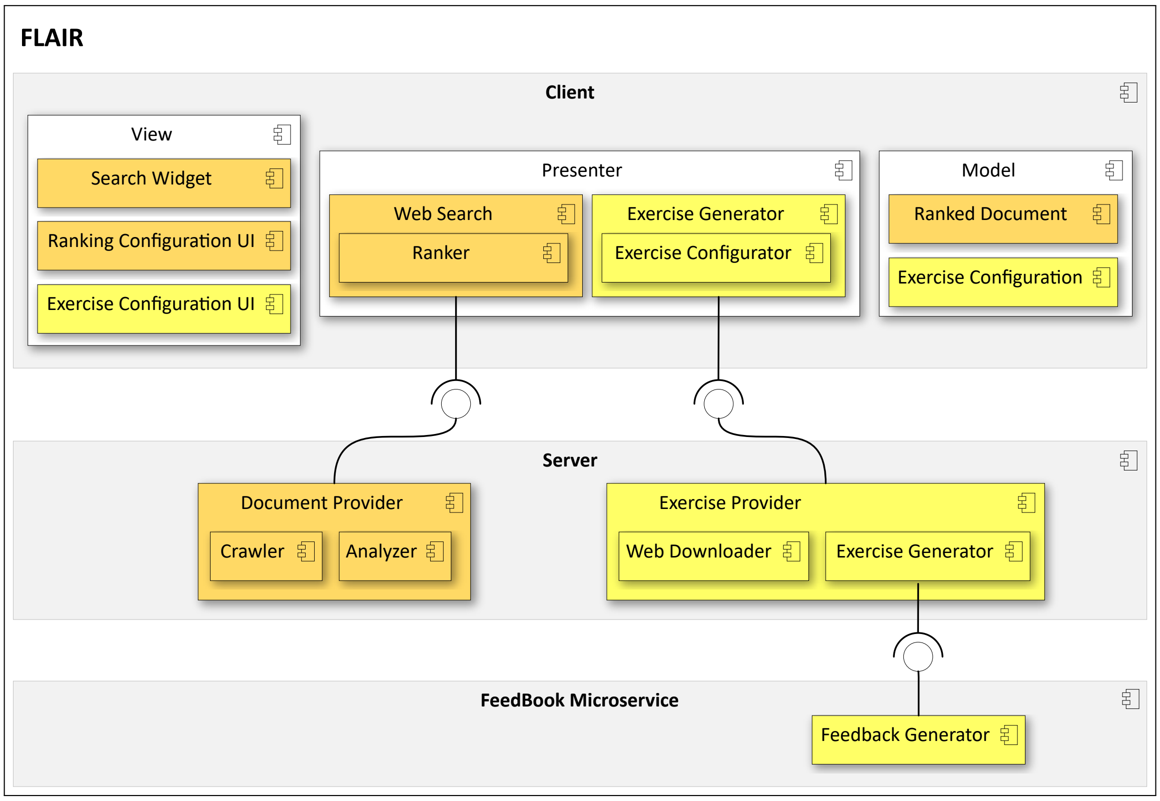
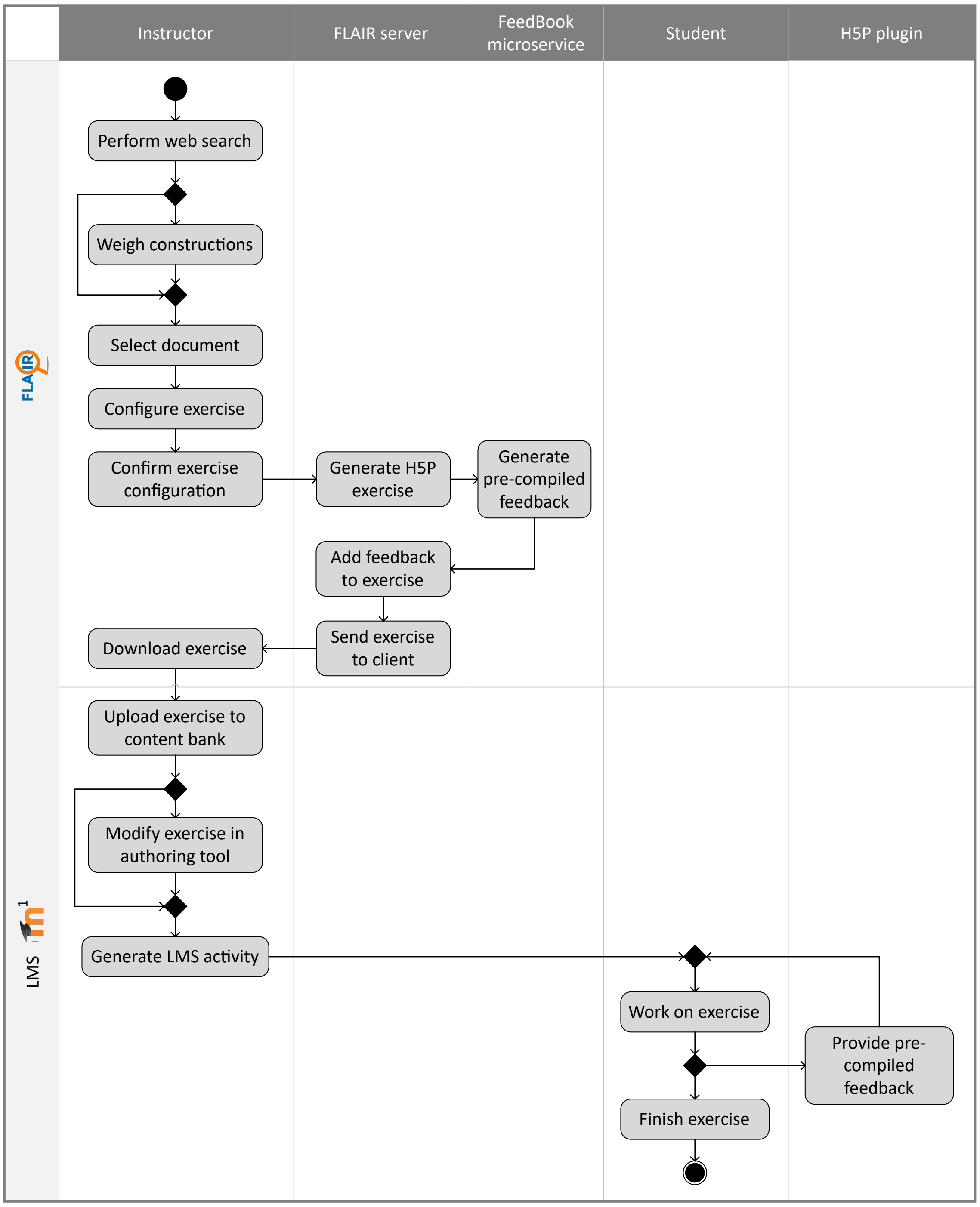


Figure 2.2.1: UML Component Diagram of the extended system architecture
The new component (*yellow*) can be integrated as a largely independent module. The existing modules (*orange*) are not impacted by the new functionality.

erates an exercise from the downloaded document. It first identifies the annotated constructions in the Document Object Model (DOM). Then it generates exercise components such as task descriptions and distractors. It adds pre-compiled feedback generated by the FeedBook microservice and converts the DOM into the format required by the JSON specification for the H5P package. Finally, it generates a H5P file populated with the thus created content.

2.3 Interaction of components

We aim at providing the exercises in a portable format which can be integrated into an existing LMS. The end user, typically a language instructor, will therefore interact with FLAIR on the one hand in order to generate an exercise and with the LMS on the other hand in order to provide the exercise to learners. The activity



[1] Julien Monty (2015)

Figure 2.3.1: Workflow Activity Diagram

The instructor configures a H5P exercise in FLAIR whose server then generates the exercise, also including feedback compiled by the FeedBook microservice. The instructor then post-processes and distributes the exercise in a LMS such as Moodle where the H5P plugin displays it to students.

diagram in Figure 2.3.1 outlines the prototypical scenario for using FLAIR$_{\text{ExGen}}$. The instructor first invokes FLAIR's web search. Weighting linguistic constructions to re-rank the results is optional. After choosing a document from the results, the instructor configures one or multiple exercises. When exercise generation is triggered, the configuration is sent to the server. The FLAIR server then generates the H5P exercise including pre-compiled feedback generated by the FeedBook microservice. The feedback, which maps a scaffolding message to each pre-compiled target hypothesis, is stored with the exercise in form of a dictionary. The H5P file is sent back to the client so that the instructor may download the exercise. The teacher then uploads the file to the content bank of a LMS such as Moodle where he or she may review the generated exercise in the H5P authoring tool and make adjustments where desired. The instructor thus represents the interface between FLAIR and the LMS. Although it is possible to add content such as textual elements, exercise targets or feedback, the primary focus of post-editing automatically generated exercises is on removing or modifying incorrect or unsuitable elements. In order to provide the exercise to students, the instructor may for example include it in a Moodle activity. This allows for post-generation sequencing of exercises as well as for including additional manually generated exercises. While working on the exercise, students will receive instant, dynamic feedback based on the feedback dictionary included in the H5P exercise until they complete the task.

In order to differentiate between operations performed in the two systems, we introduce the terms GENERATION RUNTIME to refer to tasks executed in FLAIR, whereas RENDERING RUNTIME denotes execution in the LMS.

Chapter 3

Implementation

The implementation is centered on two main aspects. The first one consists in modifying the H5P content types to fulfill the requirements for context-based exercises with intelligent feedback. The second aspect concerns extending FLAIR to host the exercise generation functionality. As the new architecture outlined in Chapter 2 suggests, this requires implementation effort both on the client side for configuration and on the server side for generation.

3.1 H5P exercises

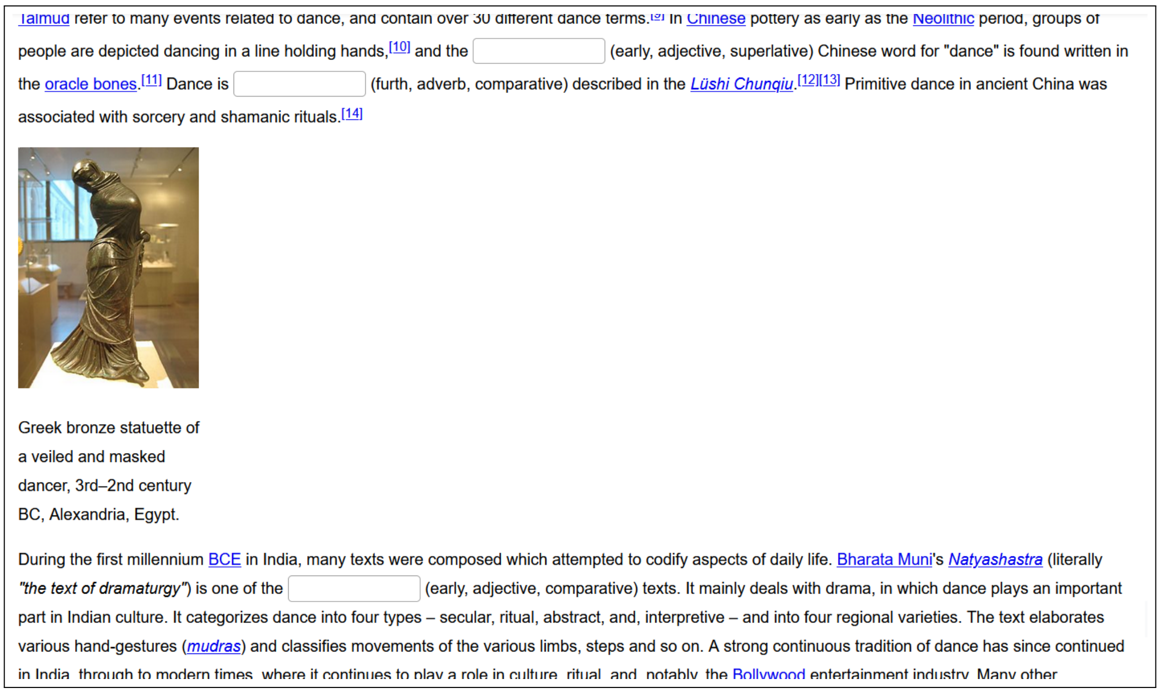
All H5P content types are based on JavaScript implementations and CSS style formatting. The authoring interface provided by the H5P environment is available for all content types. Its particular outline is defined in the `semantics` JSON specification as part of an exercise definition. It is made up of re-usable widgets and populated dynamically with exercise-specific information specified in the `content` JSON file. This file constitutes the variable component of any H5P exercise. Its structure needs to comply with the `semantics` format and contains the exercise context as well as all information on target constructions, including feedback for wrong answers. In addition, resources such as images may be added to the H5P file, although allowed file types are limited to just over 40 formats recognized by the framework (Joubel, n.d.a).

The existing content types cover the range of exercise types we want to offer, but they do not fulfill our requirement of preserving HTML context. We therefore created custom content types which are derived from existing ones. As a naming convention within our project, we prefix the names of the original content types with a *X* in order to designate our types as extensions of those types. The necessary modifications concern the JavaScript implementations for the content types as well as the `semantics` files. When deriving custom H5P content types from existing ones, the benefit of profiting from improvements and bugfixes to those content types is lost. They need to be maintained and updated manually instead. However, considering that the existing content types are not applicable to our use cases, which aim at including HTML context and intelligent feedback, continual maintenance of the content types constitutes a justifiable effort.

A requirement shared by all our custom content types consists in the ability to process text embedded in HTML elements. However, the H5P framework supports only a very limited number of HTML tags and removes any other XML tags from textual input before the content becomes available in the JavaScript code. Our custom content types therefore expect HTML content to be pre-processed in a way that HTML special characters are replaced with dummy strings. These strings were composed with the aim of minimizing their likelihood of occurring in any downloaded document so that escaping them in the original texts is not necessary. They are then converted back into the HTML characters at rendering runtime. The result are exercises embedded in marked-up pages such as the excerpt shown in Figure 3.1.1. The document's layout is mostly preserved, including resources such as images. Target constructions are replaced with the interactive widgets used by the exercise type, such as input fields for FiB exercises.

An additional challenge resides in keeping the editing interface manageable. If the HTML content was included in the editable text, manipulating the generated exercises would be unfeasible for the average language instructor as the plain text would be strewn with markup tags. We therefore separate HTML content from plain text content so that users may manipulate the plain text without getting distracted by interfering markup elements. At rendering runtime, the plain text fragments thus need to be inserted into their original position within the HTML content. In the specification, they are therefore embedded within HTML elements which are supported by the H5P framework. These elements are assigned attributes with unique

IDs which are referenced in placeholder elements inserted into the HTML text. This allows our JavaScript code to correctly reconstruct the document at rendering runtime by replacing the placeholders with the correct plain text fragments. Although the HTML elements need to be provided in the `content` JSON file, we do not want to expose instructors to them. Since H5P does by default not support elements which are contained in the JSON file but not displayed in the authoring interface, we created a new widget to this purpose. This widget implements the editor interface provided by H5P and prevents the associated elements specified in the `content` JSON file from being displayed in the authoring tool.



Talmud refer to many events related to dance, and contain over 30 different dance terms.[9] In Chinese pottery as early as the Neolithic period, groups of people are depicted dancing in a line holding hands,[10] and the (early, adjective, superlative) Chinese word for "dance" is found written in the oracle bones.[11] Dance is (furth, adverb, comparative) described in the *Lüshi Chunqiu*.[12][13] Primitive dance in ancient China was associated with sorcery and shamanic rituals.[14]

Greek bronze statuette of a veiled and masked dancer, 3rd–2nd century BC, Alexandria, Egypt.

During the first millennium BCE in India, many texts were composed which attempted to codify aspects of daily life. Bharata Muni's *Natyashastra* (literally *"the text of dramaturgy"*) is one of the (early, adjective, comparative) texts. It mainly deals with drama, in which dance plays an important part in Indian culture. It categorizes dance into four types – secular, ritual, abstract, and, interpretive – and into four regional varieties. The text elaborates various hand-gestures (*mudras*) and classifies movements of the various limbs, steps and so on. A strong continuous tradition of dance has since continued in India, through to modern times, where it continues to play a role in culture, ritual, and, notably, the Bollywood entertainment industry. Many other

Figure 3.1.1: Excerpt of a generated H5P exercise in the LMS

The exercise was generated from a web article selected in FLAIR (Wikipedia contributors, 2021c). Formatting and resources are mostly preserved. The identified targets were replaced by exercise components which, for the FiB exercise, consist of blanks input fields. Brackets behind the blanks were inserted into the text of the web page.

Another necessary replacement performed in the JavaScript code concerns image paths. Since path management is regulated by the H5P framework, it is not possible to specify relative paths at generation runtime. We therefore specify only the unique file names assigned to the downloaded resources in the exercise specification and transform them into full, valid paths at rendering runtime by means of H5P functions provided by the framework.

Since removing plain text segments in the authoring tool does not affect corresponding HTML elements, such deletions may result in empty HTML elements still contained in the DOM. While most HTML elements have formatting and layout functionalities, a couple of elements such as list items do manifest some visible display even if they do not contain any text content. We therefore manipulated the JavaScript implementation to remove any such elements without text content.

In order to specify incorrect answers and corresponding feedback, the existing H5P content types generally pursue one of two conceptually different approaches. The first approach constitutes an inline specification in which all answer and feedback information is specified as meta-information within the exercise plain text. This requires the adherence to some defined syntax differing form one content type to another. In addition, special characters of that syntax need to be escaped if used as ordinary characters. Such a strategy allows to maintain all relevant information concerning a target in one place. However, with large amounts of incorrect answers the exercise text gets cluttered with meta-information, making it hard to make adjustments to the text itself. The second approach uses separate fields in the JSON specification to define correct as well as incorrect answers and feedback. The plain text only contains a placeholder symbol which also needs to be escaped in the rest of the text if used as an ordinary character. The JSON fields for the exercise targets are mapped to the placeholder symbols in order of appearance. While this has the advantage of keeping the exercise text clean and manageable, the task of keeping track of correspondences between in-text placeholders and specification fields quickly becomes unfeasible as the number of targets increases. Since large amounts of both target forms and incorrect answers per target are frequent features of our generated exercises, neither of these two approaches is suitable. Instead, we opt for a hybrid approach which specifies incorrect answers with corresponding feedback as separate JSON fields. At the same time, it gives the correct answer, along with a reference ID to the corresponding field of incorrect answers, in the exercise text. This meta-information is enclosed in asterisks. As some feedback messages contain hyperlinks to web pages providing additional information on technical vocabulary, feedback may also contain markup elements. Since such reference material constitutes an important component of helpful, scaffolding feedback, we want to preserve the links in the H5P exercises. The resulting problem is in its core similar to that of HTML elements in the exercise text: H5P does not support `link` tags. However, as the amount of these HTML elements is rather manageable, we opt to keep them within the feedback messages in the authoring tool. This has the additional advantage of allowing users to change the link to their preferred reference page. Since this approach does not separate HTML from plain text elements, no markup for referencing IDs is required. Therefore, the H5P plain text widget can be used for feedback messages. This widget interprets any text input as plain text so that not supported markup tags are not removed. Instead, HTML special symbols are replaced with their character entity references. We therefore adjust the JavaScript code to transform special characters of link tags back into HTML characters at rendering runtime. The feedback messages do not, however, need to be altered at generation runtime.

These modifications are required for all content types with the exception of the sequencing `Quiz` content type. As it constitutes a wrapper type, the adjustments are already contained in the enclosed exercises. The following sections discuss other modifications that concern only individual types.

3.1.1 Fill-in-the-Blanks exercises

FiB exercises are based on the `Advanced fill the blanks` content type. Blanks are here implemented as inline input fields into which the learner needs to type the target answer. Feedback is provided in popup windows either instantly after typing an answer or on pressing the `Check` button. Which of the two feedback strategies is applied may be defined in the authoring tool. Although a fuzzy matching approach allows for more flexible matching of learner answers to target hypotheses, the H5P implementation might result in unforeseen and undesired side effects when applied to our exercises. Since feedback messages are associated with a particular error type, fuzzy matching might result in incorrect associations so that misleading feedback might be displayed. We therefore disable it by default, yet keep this configuration option in the authoring interface so that instructors can enable it if they so desire. For cases where the learner's input does not match any target hypothesis, we added a default feedback to the implementation alerting the learner that the answer is incorrect.

The original `Advanced fill the blanks` content type uses triple underscores to mark blanks. Since it is not unlikely that such a string might occur in an arbitrary web document (e.g. Kirka and Gannon, 2021) and also to keep our target specification uniform across content types, we replaced this with the asterisk notation. Another necessary adjustment relates to using this content type in Quizzes. The original implementation uses IDs for HTML elements which are unique within a single task instance but not when multiple tasks of that content type are included in the same Quiz instance. We therefore modified the code to use globally unique IDs.

3.1.2 Single Choice exercises

Since `Advanced fill the blanks` exercises also support dropdown blanks, the custom content type for FiB exercises can be re-used for SC exercises without further modifications. By setting the exercise mode to `selection` in the `content` JSON file, the content type uses dropdown fields from which the learner needs to select the correct answer instead of input fields.

3.1.3 Mark-the-Words exercises

Based on the `Mark the Words` content type, these exercises consist of texts in which all space-delimited words constitute a clickable item. Colour feedback is given on activating the Check button by highlighting all correctly marked target words in green and all incorrectly marked non-target words as well as not marked target words in red.

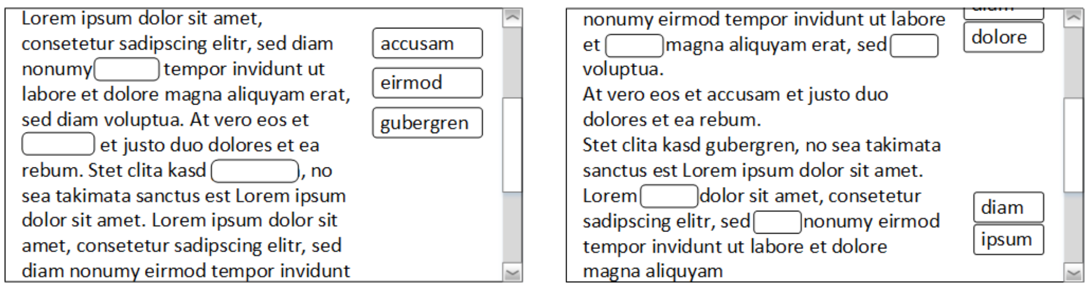
Unlike the other content types we use, `Mark the Words exercises` do not provide instant feedback functionalities. Instead, only accumulated feedback after activating the `Check` button is available. We therefore added an implementation for instant feedback as well as the configuration controls in the authoring interface to choose between the two feedback mechanisms.

3.1.4 Drag and Drop exercises

The H5P `Drag and Drop` content type provides a text in which the targets are replaced with non-editable input fields. They are instead displayed on the right margin of the page in random order. If the learner drags a target to the correct input field, green colour highlighting indicates the correctness of the assignment. Incorrectly placed targets are marked in red.

Apart from colour highlighting, this content type does not support feedback provisioning. However, it does have an information icon which is displayed next to each target construction for which a hint has been specified in the `content` JSON file. Clicking on the item then displays a tooltip with the configured text. We changed the purpose of this icon into providing feedback by only displaying it when feedback needs to be shown. Since the hint was intended to be used with a static text message, we also needed to add an implementation to select the correct feedback corresponding to the dropped item. This consists in a string comparison between the draggable and the target hypotheses stored with the droppable. In addition, although the original implementation uses asterisks to mark target constructions, it does not support escaping asterisks used as normal characters in the text. In order to correctly process our documents, we included an implementation analogous to that for FiB exercises.

We want to support the two types of DD exercises illustrated in Figure 3.1.2: (1) tasks consisting of a single exercise spanning the entire document and (2) tasks containing multiple sub-exercises, each spanning only a sentence or short paragraph. This results in two custom content types which need some additional individual adjustments. For multiple DD exercises within a single document, we modified the content type to support sub-exercises. To this purpose, we mark all text fragments belonging to a sub-exercise with the same sentence ID in the `content` JSON file. For each group of fragments with the same sentence ID, the modified code dynamically creates an exercise widget at rendering runtime. Each exercise widget consists of the text context of the contained fragments, target blanks and a bag of words containing the draggable elements which are displayed next to the sub-exercise paragraph. All other fragments are displayed before and after these sub-exercises without further modifications.


Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy ___ tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et ___ et justo duo dolores et ea rebum. Stet clita kasd ___, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt

accusam
eirmod
gubergren

(a) Single, document-spanning exercise

nonumy eirmod tempor invidunt ut labore et ___ magna aliquyam erat, sed ___ voluptua.
At vero eos et accusam et justo duo dolores et ea rebum.
Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.
Lorem ___ dolor sit amet, consetetur sadipscing elitr, sed ___ nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam

dolore
diam
ipsum

(b) Multiple sub-tasks

Figure 3.1.2: Types of DD exercises

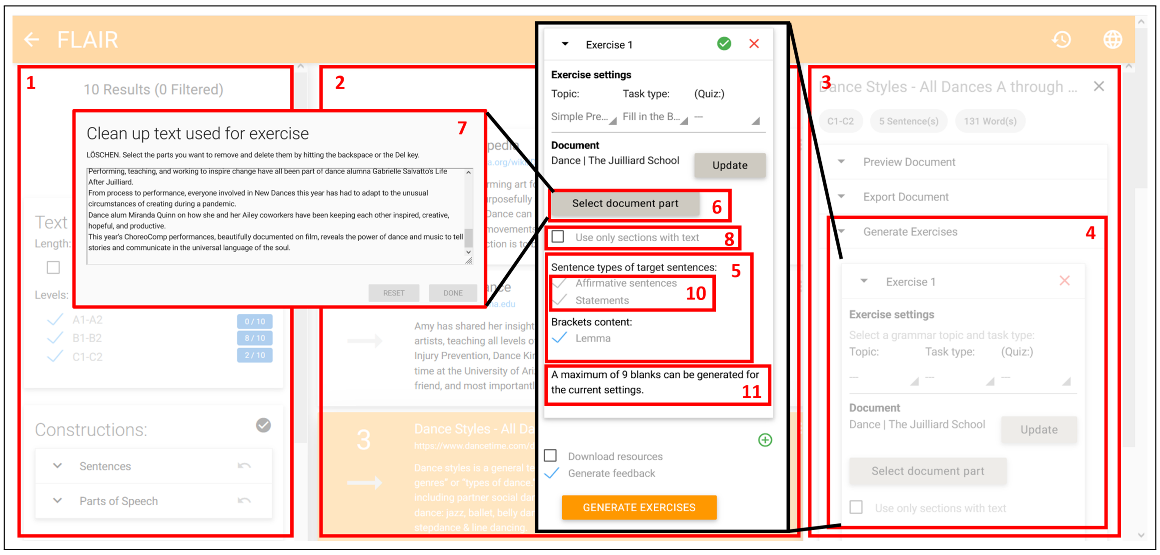
Concerning tasks consisting of a single exercise, the original implementation is not suitable for long texts. It displays all droppable items at the top right corner of the document so that they are no longer visible when the user scrolls down the page. While this may be sufficient for typical H5P exercises, our use cases usually consist of long web pages where scrolling is indispensable. We therefore enabled the draggable elements to scroll with the page so that they always float in the top right corner of the visible page content.

3.1.5 Quizzes

The only adjustment required for this content type consists in adding the package files of our custom content types and specifying the dependencies in the `semantics` JSON file in order to make them available in Quizzes. Instead of replacing the originally included content types with our custom types, we maintain support for those as well. This allows users to manually add such exercises to an automatically generated Quiz.

3.2 Exercise configuration

The generated H5P exercises are customizable in terms of the exercise type, the grammar topic and other parameters such as restrictions on target constructions which allow to vary exercise complexity. These customizations can be managed in the client's new module for exercise configuration, which is presented in Figure 3.2.1, and result in task-specific exercise settings. The indices given in the Figure will be referenced in the following description of the web interface.



Clean up text used for exercise

LÖSCHEN. Select the parts you want to remove and delete them by hitting the backspace or the Del key.

Performing, teaching, and working to inspire change have all been part of dance alumna Gabrielle Salvatto's Life After Juilliard.
From process to performance, everyone involved in New Dances this year has had to adapt to the unusual circumstances of creating during a pandemic.
Dance alum Miranda Quinn on how she and her Ailey coworkers have been keeping each other inspired, creative, hopeful, and productive.
This year's ChoreoComp performances, beautifully documented on film, reveals the power of dance and music to tell stories and communicate in the universal language of the soul.

A maximum of 9 blanks can be generated for the current settings.

Figure 3.2.1: FLAIR web interface

The exercise generation UI is integrated into FLAIR's web interface. Indices of the highlighted areas are referenced in the text. The exercise generation panel is magnified in order to display it entirely. The popup window (index 7) is only displayed after pressing the corresponding button (index 6).

The new UI functionalities for exercise generation (4) are integrated into FLAIR as a sub-module of the side panel (3) for document previews. This way, a generated exercise is automatically associated with a text selected from FLAIR's search results (2).

The configuration parameters (5) consist mostly of linguistic properties such as *irregular forms* or a specific tense. Other parameters include the content of brackets for FiB exercises and distractor properties for SC tasks. If it makes sense for a grammar topic to offer DD exercises both per sentence and for the entire text, the user may choose between these two options.

Many configuration parameters are shared across exercise types or grammar topics so that they need to be available for multiple type-topic constellations. The options to include constructions in *simple past*, *present perfect* and *past perfect* tense, for instance, are available for Past tense as well as for Passive exercises of all supported exercise types. Although it would be possible to develop a distinct widget for each grammar topic and exercise type, these shared components would then have to be implemented redundantly in all widgets in which they are required. This also implies that by changing the exercise type of a task, previous settings which might still be relevant to the newly selected type would be lost unless transferred from the template of the previous selection with considerable effort. For these reasons, we developed only a single settings widget which contains configuration controls for all possible settings parameters. The visibility of the controls is then managed dynamically by the Presenter depending on the selected topic and exercise type so that only those parameters are displayed which are relevant to that type and topic.

Since exercise targets are determined by a number of parameters, they constitute items which combine multiple linguistic properties. Although FLAIR identifies items with these properties in isolation, it does not make a fine-grained distinction into combinations of the properties. In order to determine target constructions for exercise generation, we need to identify those items which correspond to the intersection of constructions identified by FLAIR which comply with all selected settings parameters. This intersection needs to be determined individually for each combination of settings parameters since the constructions targeted by different parameters of an exercise target are not always congruent. While Example 3a constitutes a case where the adverb and comparative constructions perfectly overlap, this does not apply to Examples 3b–d. Example 3b illustrates that questions always contain the entire verb construction, yet the verb construction does not usually cover the entire question. As Example 3c highlights, the negation is not always part of the identified verb construction but may be adjacent to it. In order to determine whether the verb is being used in a negated context, we thus also need to examine preceding and succeeding tokens. In Example 3d, the passive construction and the future tense construction overlap, neither of them entirely containing the other construction.

(3) a. My brother eats ice cream $\boxed{\boxed{\text{faster}}}$ than I.
Constructions: Adverb, Comparative

b. $\boxed{\boxed{\text{Does}}\text{ he love ice cream?}}$
Constructions: Question, Simple present

c. He $\boxed{\text{does}}$ $\boxed{\text{not}}$ love ice cream.
Constructions: Simple present, Negation

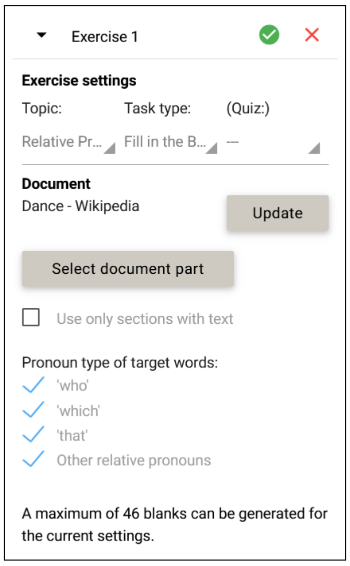
d. You $\boxed{\text{'ll}}$ $\boxed{\text{be}}$ $\boxed{\text{punished}}$ for eating all the ice cream.

Constructions: Future simple, Passive

The web interface also allows users to restrict the text used for exercise generation to only part of the document. To this purpose, a button (6) opens a popup window (7) containing the plain text of the selected web page, in which the user may delete parts of the document. Adding content is not possible. By resetting the text selection, the user may revert all modifications to the initial state which uses the entire document for exercise generation. It is still possible to further modify the selected text later in the H5P authoring tool of the target platform. However, as outlined in Section 3.1, removing plain text in the H5P authoring tool may result in empty HTML elements. The described workaround to remove empty elements should only be considered an approximation of the intended rendering. It is thus recommendable to select the desired part in FLAIR already. This has the additional benefit of reducing processing time as well as the size of the generated file.

We also offer an option to use only those parts of the document which contain text elements extracted by FLAIR (8). While this approach preserves local markup related to the plain text, global page elements such as navigation bars or footers are removed. Such elements are not relevant to language exercises and often use absolute positioning. Removing them thus has the additional benefit of decreasing the likelihood that absolutely positioned elements of the document overlap with H5P exercise components such as the task description or navigation elements in `Quizzes`. Since exercises are not usable if the exercise components are not visible, we display the task description as top-level element. However, this implies that parts of the exercise document are concealed if overlap issues occur. They should therefore be avoided.

In order to make sure that only valid exercises are configured, only those topics are offered to instructors in the configuration panel for which the FLAIR ranking mechanism has identified corresponding grammatical constructions. It would also have been possible to only enable exercise generation after the instructor has weighted the constructions he or she wants to target (1). The topic for the generated exercises could then have been determined automatically based on the construction


Exercise 1

Exercise settings

Topic: Task type: (Quiz:)

Relative Pr... Fill in the B... —

Document

Dance - Wikipedia

Update

Select document part

☐ Use only sections with text

Pronoun type of target words:

✓ 'who'

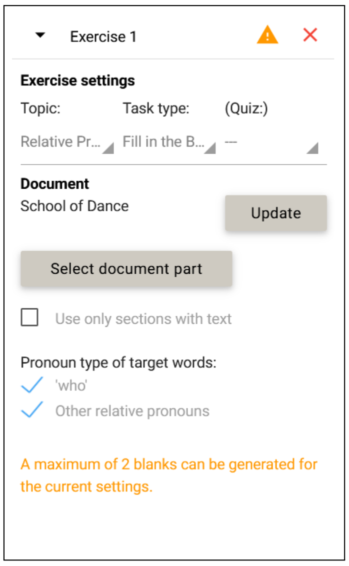
✓ 'which'

✓ 'that'

✓ Other relative pronouns

A maximum of 46 blanks can be generated for the current settings.

(a) Valid exercise configuration


Exercise 1

Exercise settings

Topic: Task type: (Quiz:)

Relative Pr... Fill in the B... —

Document

School of Dance

Update

Select document part

☐ Use only sections with text

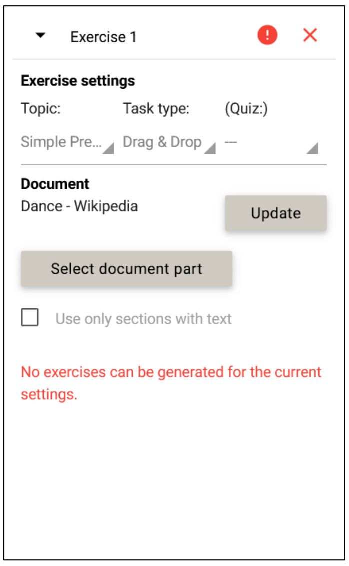
Pronoun type of target words:

✓ 'who'

✓ Other relative pronouns

A maximum of 2 blanks can be generated for the current settings.

(b) Critical exercise configuration


Exercise 1

Exercise settings

Topic: Task type: (Quiz:)

Simple Pre... Drag & Drop —

Document

Dance - Wikipedia

Update

Select document part

☐ Use only sections with text

No exercises can be generated for the current settings.

(c) Invalid exercise configuration

Figure 3.2.2: Visual display of exercise validity

weighting. However, allowing the user to choose the topic and offering all possible topics for exercise generation facilitates the use of the exercise generation functionality. It also allows users who do not wish to bother with weight configurations to generate exercises for a selected document. Additional precautions to prevent invalid exercise configurations include only displaying those configuration parameters for which corresponding constructions have been found in the selected document part. Configuration parameters which need to be selected in order to generate any exercise at all regardless of other parameter settings are disabled (10) so that they cannot be deselected. Since combinations of parameter settings can still lead to invalid exercises which do not contain any target constructions, the number of targets for the current configuration is re-calculated and displayed at the bottom of the configuration panel (11) on each configuration modification. This number constitutes an estimate based on the exercise targets determined from overlaps of the extracted constructions. Invalid exercises such as the one displayed in Figure 3.2.2c are marked with an error symbol. As it is likely that not all identified targets can be used in the generated exercise, an additional warning symbol alerts the user if the specification may fail to produce an exercise due to low target occurrence. This is the case for the example given in Figure 3.2.2b. The threshold for this is currently set to less than five occurrences. Valid exercises receive a green icon as illustrated in Figure 3.2.2a

If the instructor selects another document from FLAIR's search results, the text on which the configured task is based is not automatically updated. This allows a user to configure multiple exercises for a range of different documents on the same or various topics. The text is only updated if and when the Update button of a task configuration is pressed.

Defining multiple exercises within the same session is most useful if the generated exercises are to be bundled in a `Quiz`. In order to specify whether to generate a Quiz and, if so, which exercises to bundle together, the user may select a Quiz from the dropdown field. The dropdown always offers all Quizzes defined in any of the configured tasks as well as an additional one to start a new Quiz. If no Quiz is selected, the exercise is generated as a simple exercise. Such exercises may still be sequenced later through the means provided by the target LMS.

A global configuration option for all tasks allows instructors to specify whether resources of the document, in particular images, are to be downloaded and included in the H5P package. Since this can result in very large files, especially if the user does not restrict the used document parts, the default is to not download the resources. Disabling the download checkbox does not, however, remove them from the document altogether. Instead, the paths specified in the HTML and CSS definitions of the web page are set to absolute paths. The resources can therefore only be displayed as long as they are accessible on the original site, but processing time and file size are reduced.

Another global setting applied to all exercises concerns feedback generation. This option is only provided if at least one of the configured exercises supports automatic feedback generation. If this is the case, the default setting includes feedback in the generated exercises. The instructor may, however, choose to disable the feature in order to reduce processing time.

Once the instructor is satisfied with the task configurations and has managed to configure at least one valid (or critical) task, a button click initiates sending all valid configurations to the server. FLAIR's existing client-server architecture with thread pooling on the server allows for minimal processing time for client requests. If only a single exercise was configured or if all exercises were assigned to the same Quiz, the server returns the generated H5P file. If multiple tasks need to be generated, they are returned in a zipped format. After an exercise has been generated successfully, a `Download` button is displayed at the bottom of the configuration panel. Pressing it triggers the browser's file download for the generated file. The implementation makes use of JavaScript injection in order to provide the file, which is stored in-memory, to the browser. The file names of the generated H5P files correspond to the names of the specified tasks or, if they represent a Quiz task, to the name of the Quiz, so that the instructor may associate them with their corresponding configuration prior to opening them on the target platform. In order to verify the suitability of the exercise within FLAIR$_{\text{ExGen}}$, a preview of the exercises can be opened when they have been successfully generated. While the preview does not contain interactive elements such as distractors or feedback, it displays the exercise, including the HTML context, and highlights the exercise targets in green colour. Instructors can therefore determine at a glance whether the document is suitable, and identify passages which they might wish to remove from the exercise.

An instructor may also abort a running exercise generation. Configurations based on very long documents, containing large amounts of targets, and requesting resource download and feedback generation can result in very long processing times. If a user thus decides to modify the configurations to facilitate the process, the `Cancel` functionality allows to stop the running exercise generation and start a new one with the modified settings. As the `Generate exercise` button is disabled while an exercise is being generated, the instructor would otherwise be forced to reload the page and specify the configurations from scratch in order to issue a new exercise generation request.

3.3 Exercise generation

Although FLAIR is highly compatible with our requirements, some adjustments to existing modules were necessary in order to adequately support the grammar topics intended for exercise generation. These adjustments are outlined in Section 3.3.1. The main implementation effort is, however, centered on the new exercise generation functionality which is presented in Sections 3.3.2–3.3.6.

3.3.1 Adjustments to existing modules

FLAIR's base implementation only supports identifying relative clauses, but not relative pronouns for English. As Example 4 illustrates, relative pronouns are highly ambiguous in English. While the first occurrence of *that* in Example 4a constitutes a conjunction, the second occurrence is a determiner and in the fourth occurrence, the same token is used adverbially. Only the third occurrence is indeed a relative pronoun. Although not used quite as diversely as *that*, Example 4b shows that other relative pronouns like *who* can also be easily confused with other parts of speech like, in the first occurrence, an interrogative pronoun. Simple string matching is therefore not suitable to identify the pronoun of a relative clause as it may produce incorrect results. A more sophisticated and less error-prone solution implies extending the server back-end to also support identifying relative pronouns. Since this has already been implemented for German, the infrastructure was in place and we only needed to supply the parsing logic to identify the constructions.

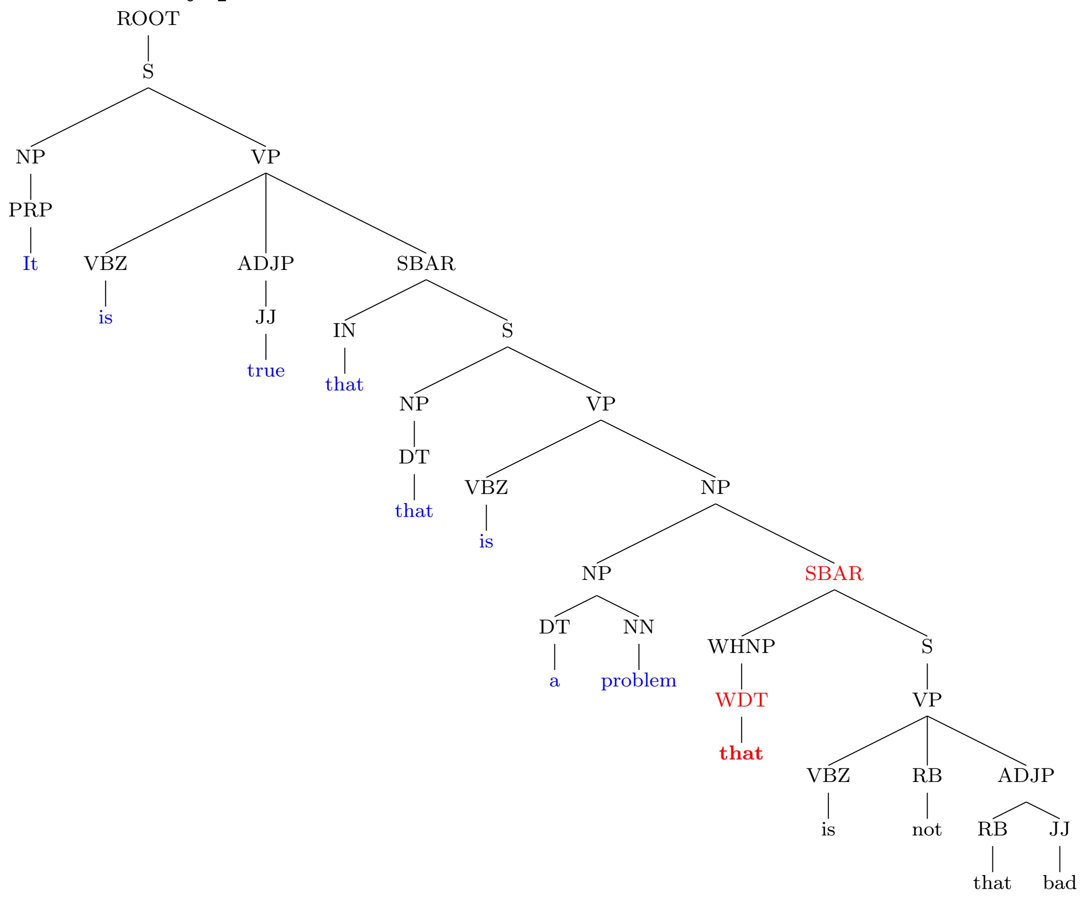
(4) a. It is true **that that** is a problem **that** is not **that** bad.
 b. **Who** is the guy **who** caused the problem?

As the Stanford POS tagger used for English does not support a tag for relative pronouns, the constituent parser is best suited to identify them. To this purpose, the tags of the individual tokens as well as those of higher-level non-terminal symbols in the parse tree need to be considered. Tree Regular Expressions (TRegEx) allow to efficiently identify items with the required tag and ancestors in the tree structure. However, this structure does not preserve start and end indices of terminal symbols. Therefore, once the terminal symbols which constitute the construction have been determined, their indices need to be extracted from the list of POS tagged tokens. As illustrated in Figure 3.3.1, the index of the correct item in that list corresponds to the amount of left leaf sisters of the identified token in the constituent tree.

3.3.2 Web download

Although downloading the document's DOM is a trivial task in itself, we need to account for a number of technical peculiarities in order to be able to integrate the downloaded HTML into a H5P environment. This includes avoiding cross-domain

Constituency parse



POS tagged tokens

It	is	true	that	that	is	a	problem	that	is	not	that	bad
[0,2]	[3,5]	[6,10]	[11,15]	[16,21]	[22,24]	[25,26]	[27,34]	**[35,39]**	[40,42]	[43,46]	[47,51]	[52,55]

Figure 3.3.1: Indexing of relative pronouns

The constituency graph is based on the output generated in the web tool CoreNLP[1]. The pronouns are identified in the constituency tree through the tags assigned to them and their ancestors. The terminal symbol *that* is identified as a relative pronoun based on the nodes marked in red. *SBAR* indicates a clause, *WDT* represents WH-determiners according to the Peen Treebank tagset[2]. Since the relative pronoun has 8 left leaf sisters (blue) in the parse tree, it appears as the ninth token (index 8) in the POS tagged list. From that list, the corresponding start and end indices (35 and 39 respectively) are extracted.

issues with IFrames as well as modifying relative resource paths which will no longer be valid on the platform hosting the H5P exercise.

[2]http://corenlp.run/
[2]https://gist.github.com/nlothian/9240750

We therefore download IFrames recursively analogous to downloading the main web page and integrate the downloaded contents into content IFrames which replace the reference IFrames. Since CSS files are not supported as resource files of H5P packages, they cannot be downloaded and added to the package as files. Instead, their content is transferred to HTML style elements.

As the configuration interface in FLAIR allows to remove parts of the text, some resources specified within those segments will also be removed. It is consequently not necessary to download all resources of the original DOM. However, the parts removed from the plain text need to be indexed in the HTML structure in order to determine which resources are no longer relevant. Downloading resources is therefore deferred to a later processing step in order to avoid download overhead. URIs which do not correspond to downloadable resources such as links to other web pages, however, are already set to absolute paths when the document is downloaded.

3.3.3 Post-processing of exercise elements

Since we use the linguistic constructions extracted by FLAIR as exercise targets, most of the relevant NLP processing has already been completed. Some post-processing is still required for a number of topics and exercise types. We use the Stanford CoreNLP tools also used to extract the constructions in order to maintain as much consistency as possible in the linguistic analysis between document selection and exercise generation.

Construction adjustments Whenever the exercise configuration expects targets whose token span differs from that of the extracted constructions, the construction indices need to be adjusted.
Concerning conditionals, FLAIR only extracts the entire conditional sentence. Since our exercises focus on the verb constructions of the clauses, we need to extract them from the sentence and associate them with either the main or the if-clause. We use a constituency parser to determine the clauses by means of TRegEx specifications and identify the verb targets through POS tags. Since the clauses are also required to determine the brackets content, we store their indices and the clause type with the exercise settings to make them globally available for later processing steps.
For FiB exercises on passives where an active sentence is specified in brackets and

learners need to give the passive sentence, the target needs to be extended to encompass the entire sentence. Post-processing is also needed with this topic for DD exercises for which participles are to be used as individual targets separate from the rest of the verb construction. For these tasks, the participles are identified and isolated based on POS tags.

For both types of DD exercises, the droppable items can only be displayed on the right-hand border of the web page if they are not too long. Otherwise, they are all accumulated underneath the (sub-) task. This does not pose a problem for small texts so that it is not an issue for DD tasks with multiple exercises as each sub-task consists of only one or few sentences. It does, however, defeat the purpose of the added scroll mechanism for DD exercises spanning the entire document. If the draggables are displayed at the bottom of the page, they are, once again, not visible after scrolling. Relative pronouns do not tend to be very long, yet verb constructions of present and past tenses may easily reach problematic dimensions. For these tasks, it is therefore likely that the draggable items cannot be displayed on the page's border. We thus try to limit the length of the targets for exercises on tenses by removing arguments of the verb which are sometimes extracted by FLAIR as part of the constructions and keep only the verb cluster. If this still results in long targets, we use only the main verb as exercise target. The threshold to determine what qualifies as a long target is currently set to 30 characters.

An additional requirement for DD exercises which are made up of multiple sub-exercises is that they need to define the scope of such a sub-task. This scope generally corresponds to a single sentence. We therefore determine the indices of all sentences which contain at least one exercise target by means of the sentence segmenter of the NLP pipeline.

Task description generation Instructions in terms of a task description need to contain all the information required by a learner of sufficient proficiency to successfully complete an exercise. This includes any information on the exercise targets not given in the text context of the exercise itself or, in the case of FiB exercises, in brackets.

While some necessary information such as the lemma of the target construction is target specific, other properties such as the topic in terms of *Relative Pronouns*, *Simple Present*, etc. apply to the entire task and still others such as the tense of target

constructions may or may not be shared across all targets. We therefore rely on the user's configuration generated in FLAIR's web interface to determine whether information on target constructions of FiB exercises should be given in brackets with the target constructions or globally for the entire exercise. The information in brackets often determines what kind of linguistic constructions are targeted and is therefore relevant for successful completion of the exercise. If the user has de-selected a configuration parameter for brackets, the brackets in the generated exercise will not contain the corresponding information, yet it still needs to be transmitted to the learner. It is thus provided as global information in the task description which applies to all exercise targets.
Explicitly specifying instructions for each settings constellation would be cumbersome and poorly maintainable. If context specific elements such as the lemmas of the target constructions are to be included in the instructions, such an approach becomes outright impossible. We therefore introduce an instruction template for each type-topic combination which is dynamically enriched at exercise generation runtime with variable content of the configured exercise settings. While meta-information such as tense, voice or aspect can be directly inferred from the construction type, some variables like lemmas require additional NLP processing of the exercise targets.

Brackets for FiB exercises Since many instruction components may be specified either in the task description or in brackets, post-processing for brackets manifests some parallels to that required for task descriptions. This does, however, not imply that the same information is extracted redundantly. It rather allows for the relevant code to be executed at the place where the information is needed. This place, either for task description generation or for brackets compilation, is defined by the exercise configuration.
For ***Comparison*** tasks, the lemmas of the adjectives or adverbs need to be given. Since the lemma provided by the Stanford CoreNLP lemmatizer for comparatives and superlatives is identical to the token form instead of the positive form of the adjective or adverb, we use a dictionary-based lemmatizer of the Apache OpenNLP[3] library instead with a dictionary file compiled by Richard Willy (2013). As both the dictionary file and the CoreNLP POS tagger employ the Penn Treebank tagset, we can use the output of the existing NLP pipeline as input for the lemmatizer and

[3]https://opennlp.apache.org/

thus obtain the lemmas with minimal additional processing effort.
For tasks on ***Passives***, required post-processing depends on the brackets content. If the lemma is needed in the brackets, the main verb of the verb construction needs to be determined and its lemma extracted from the NLP pipeline. If instead the active sentence is required as brackets content, the passive sentence needs to be transformed accordingly. This is achieved by determining the verb construction, the subject of the passive sentence and, if available, the by-clause through dependency relations of the dependency parser. In order to generate the active sentence, we use the natural language generator library SimpleNLG[4]. By providing the lemma of the verb, the subject of the passive sentence as object of the active sentence and the agent of the by-clause (or a default *someone or something* if none is given) as the new subject, the active sentence can be generated. Any additional parts of the passive sentence remain unchanged so that only the part consisting of the agent, the patient and the verb are replaced. In authentic texts, by-clauses are often omitted. The variant with agent, verb and patient in brackets suggested in Section 1.4.2 is not applicable in such cases. FLAIR extracts any passive sentences regardless of whether they contain a by-clause or not and does not provide weighting options for by-agents. Whether passive sentence are suitable for exercises with such bracket content can therefore only be determined at exercise generation time, yet is unknown during exercise configuration. In order to prevent a frustrating user experience where exercise configurations fail to result in successful exercise generation, we do not use this suggested variation for brackets in our exercises.
Analogously to extracting the main verb and lemma of passive constructions, the verb lemma is also determined for ***Tense*** exercises. By lemmatizing the entire verb construction and including any contained non-verb tokens such as negation or personal pronouns, we make sure that the brackets contain all necessary information.
For tasks on ***Conditionals***, we also extract the clause type from the settings and determine the modals of main clauses if they are to be given in brackets.

Distractors for Single Choice exercises The challenge of generating appropriate distractors has received considerable attention in the field of Computer-Assisted Language Learning (CALL). Approaches for vocabulary, grammar and semantic language exercises range from identifying tokens with similar properties as the target

[4]https://github.com/simplenlg/simplenlg

construction in the text or in external resources such as lexical databases (e.g. Coniam, 1997; Brown et al., 2005; Sumita et al., 2005; Liu et al., 2005; Karamanis et al., 2006; Smith et al., 2010; Knoop and Wilske, 2013; Chen et al., 2015; Jiang and Lee, 2017; Susanti et al., 2018) to using common learner errors (Sakaguchi et al., 2013) as well as applying machine learning techniques (e.g. Liang et al., 2017; Yeung et al., 2019).

Since we aim at allowing users to vary exercise complexity by, inter alia, specifying characteristics of distractors, these approaches are not suitable for FLAIR$_{\text{ExGen}}$. Instead, we employ a rule-based approach to derive ill-formed as well as task-inappropriate variants, such as incorrect tenses in past tense exercises, from the target and use them as distractors. This process has notable parallels to FeedBook's generation of target hypotheses. It would, in principle, be possible to use FeedBook's feedback generation algorithm not only for feedback generation, but also for distractor generation. However, this algorithm applies all applicable transformation rules for generating target hypotheses to the target forms which results in a large number of incorrect forms. Since this can be pedagogically problematic, we implement our own distractor generation logic which applies a limited set of rules manually defined for each topic. This gives us more control over the proportion of ill-formed as opposed to well-formed but context-inappropriate distractors and allows us to control the degree to which an ill-formed distractor deviates from the correct form.

In order to generate well-formed but context-inappropriate distractors, we again employ the SimpleNLG generator. For ill-formed variants, we introduce rules explicitly defined for each topic to generate forms representing common mistakes for that topic. Examples 5–7 illustrate prominent errors, marking incorrect forms with an asterisk (*). For ***Comparison*** forms, ill-formed variants include neglecting to substitute a trailing *y* with an *ie* (Example 5a) or to double a trailing consonant (Example 5b) or else wrongfully doubling it (Example 5c) before appending the suffix *er* or *est* for comparative or superlative forms respectively.

(5)	a.	**pretty:**	prettier	*prettyer
	b.	**big:**	bigger	*biger
	c.	**smart:**	smarter	*smartter

For ***Simple present***, incorrect forms target wrong question and negation formation such as illustrated in Examples 6a–b, as well as incorrect suffix formation. In order not to introduce too many errors within a single distractor, we apply incorrect suffix formation only on affirmative statements. Such forms may miss an *e* (Example 6c) or feature an additional *e* before the *s* suffix (Example 6d) or else, similar to comparison forms, fail to substitute a trailing *y* with *ie* (Example 6e).

(6)	a.	**go (he, interrog.):**	Does he go?	*Goes he?
	b.	**go (he, neg.):**	He doesn't go.	*He not goes.
	c.	**go:**	goes	*gos
	d.	**say:**	says	*sayes
	e.	**fly:**	flies	*flys

Since ***Past tense*** exercises may also include forms of other tenses, generating ill-formed variants needs to be possible for all past, present and future tenses in perfect as well as non-perfect aspect and simple as well as progressive forms. Perfect and progressive forms mainly target incorrect past and present participle formation respectively, but also incorrect auxiliary use in terms of wrong person or lemma or else incorrect question or negation formation. Incorrect past participle forms may, as illustrated in Example 7a, use an incorrect suffix such as *d*, *t* or *en* instead of the regular *ed*. Incorrect present participle forms may for instance neglect to double a trailing consonant (Example 7b) or to drop a trailing *e* (Example 7c) before appending the *ing* suffix. Simple past uses the same logic applied to past participles since the rules for generating regular forms are identical.

(7)	a.	**talk (past):**	talked	*talkd	*talkt	*talken
	b.	**plan (pres.):**	planning	*planing		
	c.	**come (pres.):**	coming	*comeing		

In order to provide the requested amount of distractors, a sub-sample needs to be selected from the pool of generated distractors in case their number is higher than that specified in the configuration interface. The simplest approach consists in taking a random selection of the generated forms. However, the amount of generated ill-formed variants usually exceeds that of inappropriate forms. Since presenting incorrect forms to learners should be moderated so as not to expose them to an abundance of ill-formed language, we need to control their use as distractors. Another factor to consider in the selection of distractors concerns the availability of feedback. As the logic for generating distractors may differ from the one for generating target hypotheses for feedback, not all distractors might be assigned a corresponding feedback. In order to provide feedback for as many forms as possible, we prioritize distractors with associated feedback over those without in the selection process. Distractors can therefore only be chosen after feedback has been generated.

Target constructions can only be used for exercise generation if all necessary exercise components could be determined. Post-processing may not be successful for all targets if, for instance, the clauses of conditional sentences cannot be identified. This step may therefore result in a reduction of exercise targets.

3.3.4 HTML indexing

FLAIR operates on extracted plain texts, subsequently referred to as FLAIR PLAIN TEXT, without preserving any HTML context. In order to integrate exercises in the original HTML context, the identified constructions need to be annotated in the plain text extracted from the HTML DOM, subsequently called HTML PLAIN TEXT. Although it would be possible to repeat the NLP analysis on the HTML plain text, this would entail redundant processing. Even more importantly, this approach cannot account for parts of the document deleted by the user in the configuration interface. It is therefore not suitable for our purposes.

Instead, we pursue an alternative approach which consists in mapping the FLAIR plain text indices of the constructions, as well as those of the removed document parts, to the HTML plain text. Considering that both plain texts are extracted from the same web page and only deletions but no insertions are allowed on the FLAIR plain text, the FLAIR plain text generally constitutes a subset of segments of the HTML plain text, without additional segments. However, this is not an absolute given. The reason resides in the discrepancy in plain text extraction tools and methodologies between the two plain texts, which results in slightly different plain texts. While the construction extraction pre-processing uses Boilerpipe for performance reasons, $\text{FLAIR}_{\text{ExGen}}$ parses the HTML DOM with the jsoup[5] parser in order to also keep the HTML tags instead of only extracting plain text elements.

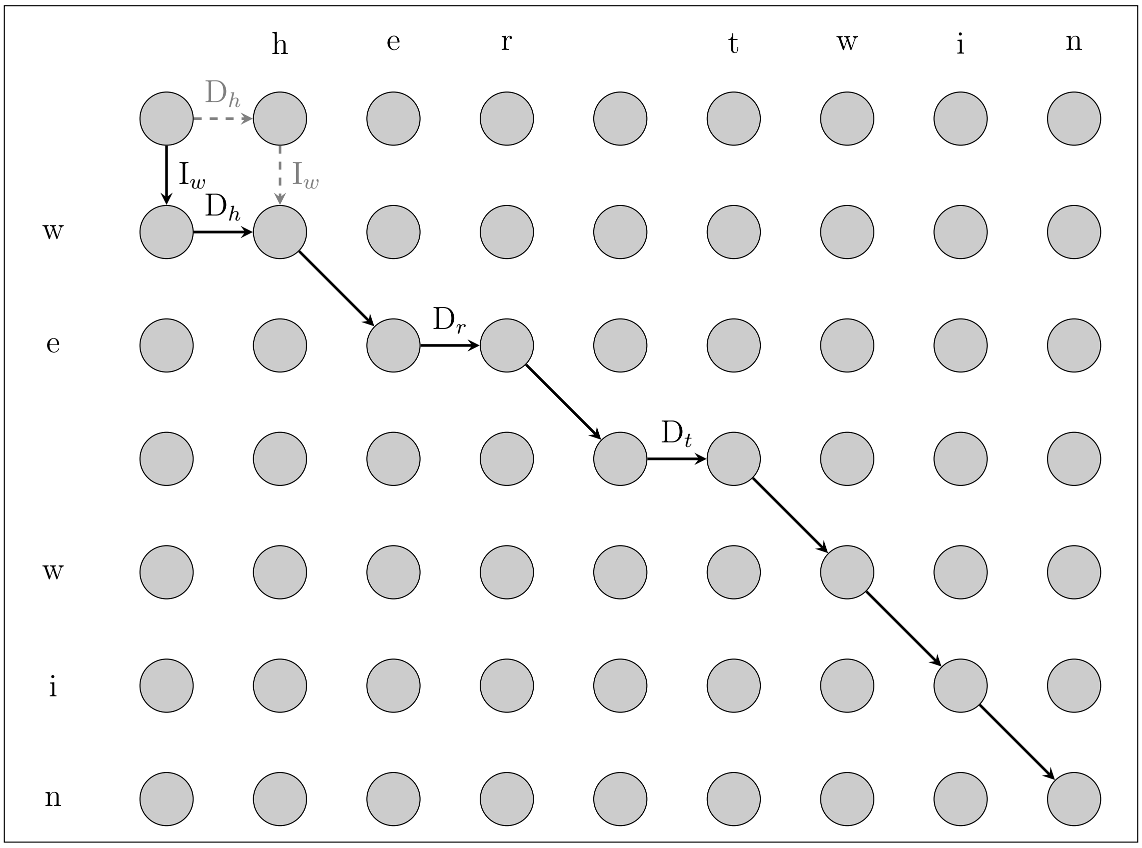


Figure 3.3.2: Edit graph for operations supported by the Myers algorithm

The text on the vertical axis represents the HTML plain text, the text on the horizontal axis the FLAIR plain text. The Myers algorithm supports only insertions (I), denoted by downward arrows, and deletions (D), denoted by arrows to the right. The affected letter is given as subscript of the operation. Diagonal arrows indicate that both plain texts are identical. Dashed arrows in gray denote alternative paths considered equally likely by the algorithm. The graph only illustrates the basic principle for applying the supported edit operations. Optimizations implemented by the Myers algorithm are not considered.

Since such mapping problems are also central to difference algorithms used for example in version control systems, we examined these algorithms for applicability to our purposes. Some algorithms such as `Histogram` are optimized for code comparison where many lines tend to be made up of certain elements like brackets (Nugroho et al., 2019). Other algorithms such as `Myers` have a broader application scope.

[5]https://jsoup.org/

This algorithm is optimized to identify differences between two texts by determining the shortest path in an edit graph (Myers, 2005). Figure 3.3.2 illustrates how the algorithm finds differences between two strings by applying one of its two operations `Insert` and `Delete` whenever differences are detected. Since this algorithm has been implemented in the Apache Commons library[6], it can be easily integrated into FLIAR$_{\text{ExGen}}$ to allow efficient indexing.

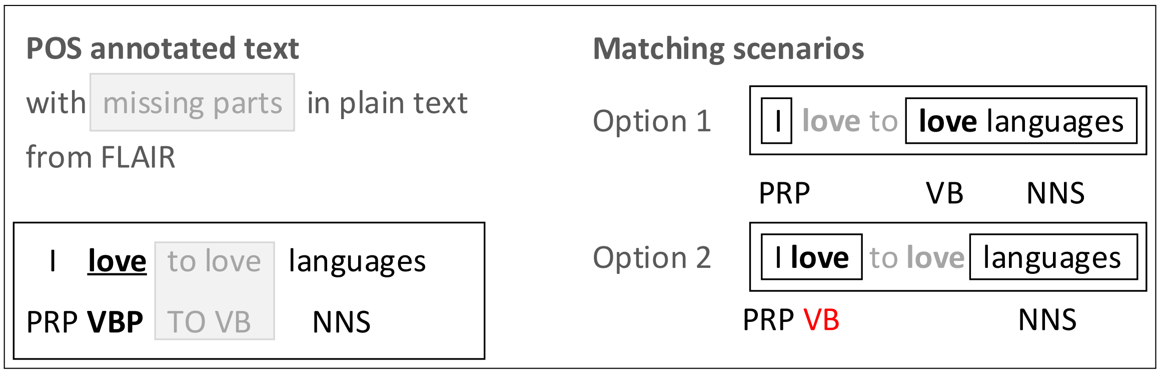


Figure 3.3.3: Ambiguity in plain text matching

The FLAIR plain text lacks a to-infinitive contained in the HTML plain text. The matching allows two possible scenarios regarding the deleted part in which the token *love* is once at the left-hand border (Option 1) and once on the right-hand border (Option 2) of the removed segment. While Option 1 reflects the correct mapping, Option 2 leads to a mapping where the ambiguous token is wrongly assigned the Peen Treebank tag *VB* for base verbs instead of the present tense verb tag *VBP*.

While the algorithm is very effective and efficient in identifying the most likely changes, it cannot account for ambiguities in the matching. Figure 3.3.3 illustrates why such ambiguities can be problematic: If some sequence is deleted from the original text that ends with a token which also makes up the rightmost element of the sequence to the left of the deleted part, this token might be matched to either of the two occurrences in the original text. The same holds for tokens on the left of the deleted sequence which also occur at the left border of the part to the right of the deleted segment. If the two occurrences have different grammatical functionalities, only one of them might constitute an exercise target. By assigning the ambiguous token to the wrong element in the original text, the remaining occurrence might thus wrongly be assigned the target construction annotation of the deleted token. Since such a scenario is highly unlikely considering that both plain texts are extracted

[6]https://commons.apache.org/

from the same web document so that the main text containing constructions should be included in both, we do not attempt to further avoid this kind of error. It will be up to the instructor to revise the generated exercises and remove targets as needed in the H5P authoring interface before presenting the exercises to learners.

Since the Myers algorithm allows deletions as well as insertions, the output may result in a mapping of the two plain texts where not all parts of the source text, in our case the FLAIR plain text, are matched to a part of the target text, our HTML plain text. If such a part contains a linguistic construction, it cannot be used for exercise generation as it cannot be located in the HTML DOM. Example 8 highlights that even if only part of the construction (*bold*) of the FLAIR plain text given in Example 8a is missing (*red*) in the HTML plain text shown in Example 8b, the construction is unsuitable as an exercise target. The indexing may thus lead to an additional reduction of the amount of exercise targets.

(8) a. He **is** **being** nice.
    b. He is nice.

3.3.5 Feedback Generation

We use FeedBook's microservice to provide scaffolding feedback with our generated exercises. While FeedBook's offline component generates multiple pre-compiled feedback messages for each exercise target at generation runtime, the online component identifies the one best suited for a learner's input at rendering runtime. For offline processing, the feedback generation takes the correct target answer, which can be processed by standard NLP tools, as a starting point. The algorithm iteratively transforms it into well-formed but task-inappropriate as well as ill-formed target hypotheses. The iterative nature of the algorithm allows it to associate errors with transformation rules, thus enabling the tool to provide meta-linguistic information on the error characteristics. Each target hypothesis is associated with one of the tool's manually defined feedback templates. The templates are populated based on the transformation rules that are applied to generate the variant. In order to provide the feedback that fits best to a learner's answer, the online component enables the matching algorithm to, for example, gloss over additional words inserted by the learner which are not incorrect, but not foreseen in the target answer. This functionality is provided by the search algorithm of the Lucene[7] search library, enhanced by FeedBook's implementation for task-specific term weightings to take into account the task context. The enhancement allows the algorithm to prioritize hypotheses based on task-relevant transformations over those based on transformations not relevant to the task (Ziai et al., 2018). For the exercise and the student's answer (*red*) given in Example 9, the implementation with this enhancement will rate message 9b as more relevant since the task description explicitly asks for present progressive. The learner is thus directly pointed to the main issue.

(9) Give the correct present progressive form:
He *go* (go) to school.

a. Simple present: He, she, it the s must fit.

b. This is the simple present. You need to use the present progressive here.

Considering that the microservice development constitutes work in progress, the current implementation supports only offline feedback generation. We therefore do not presently integrate online feedback into FLAIR$_{\text{ExGen}}$. Providing feedback for MtW exercises is not possible without online analysis of the learner's answer and DD tasks are not supported by FeedBook's algorithm. Exercise types for which feedback generation is already supported thus encompass FiB and SC exercises for which online processing is not essential. Integrating feedback into FLAIR$_{\text{ExGen}}$ consists of assembling the exercise information required by the microservice, calling the microservice and compiling the returned feedback into the format required by the H5P exercises.

Feedback generation is only triggered if the according checkbox in the exercise configuration interface has been activated. For exercise types which do not support feedback generation, i.e. MtW and DD exercises, the feedback generation logic is automatically bypassed. The microservice does not support passive transformation exercises which provide the active sentence in brackets and expect the corresponding passive sentence as target answer. Support for this exercise type is, however, currently being developed. We therefore enable feedback generation for such exercises. Suitability of the generated feedback for those tasks can then be evaluated by the

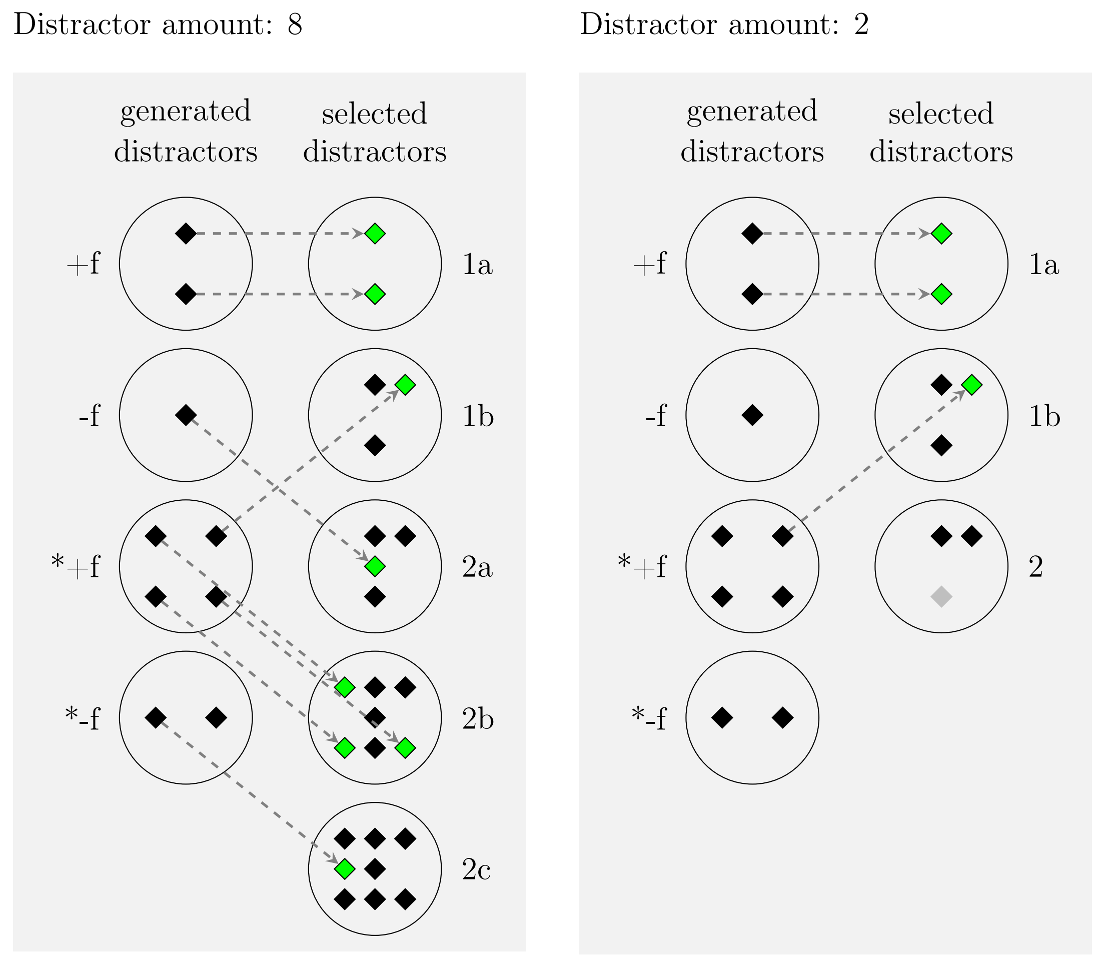
[7]https://lucene.apache.org/

instructor in the authoring tool before presenting it to learners.

If the configured exercise requires feedback, a POST request is sent to the FeedBook microservice. Since the microservice is not designed to process large requests, we split the exercise items into multiple messages. Batch size is currently set to 20 items per request. Each item consists of the sentence containing the target as well as the previous sentence in order to provide sufficient context to perform NLP analyses. In addition, the exercise target needs to be specified by giving its start and end indices within the containing sentence. If a sentence contains multiple target constructions, each target is processed as an individual item, treating other targets as regular plain text elements of the text context. Although information on exercise topic and type is not evaluated by the feedback algorithm, it is added to the request for possible future processing and logging purposes.

The response returned by the microservice contains a dictionary of target hypotheses with associated feedback for each item extracted from the request. The feedback algorithm may determine multiple possible feedback messages corresponding to a target hypothesis. However, only the most likely one according to a set of static rules defined in the feedback algorithm is returned in the response. Since we do not presently support online feedback, determining the most probable feedback at generation runtime suits our current implementation.

As the feedback generation was originally developed exclusively for the FeedBook system, some feedback messages contain elements specific to that application such as links to system-internal HTML pages and references to pages in the accompanying paper text book. Although we cannot filter out the latter in our system, it is possible to identify critical links. We thus replace those with publicly available alternatives manually specified in resource files of FLAIR$_{\text{ExGen}}$. Other than that, no further processing is required for FiB exercises before storing the generated target hypotheses with the respective target. They are added to the target information, along with the associated feedback, as incorrect answers. For SC exercises, only those target hypotheses which correspond to generated distractors are used and their feedback is stored with these distractors. Since the feedback generation algorithm might not cover all distractors, we consider this in the distractor selection as illustrated in Figure 3.3.4. The algorithm regulates the proportion of ill-formed distractors and prefers forms with associated feedback over those without. To this purpose, it



(a) Distractor padding (b) Distractor reduction

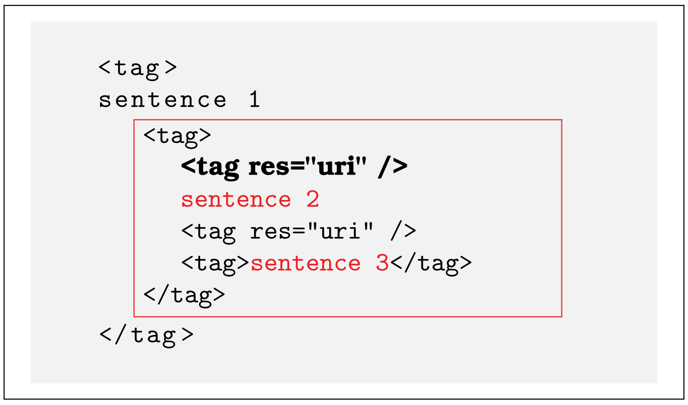
Figure 3.3.4: Distractor selection

The circles on the left in the figures indicate groups of distractors with either ill-formed (*) or well-formed but task-inappropriate variants and either with (+f) or without (-f) associated feedback. The circles on the right contain the currently selected distractors at each represented processing step 1a through 2c in Figure 3.3.4a and 1a through 2 in Figure 3.3.4b respectively. Distractors are depicted as diamond shapes. Newly added distractors in a processing step are marked in green, distractors removed in a step are depicted in light gray. Steps 1a and 1b are always executed. If the number of selected distractors after these steps is lower than the required amount, the set of selected distractors needs to be padded from the remaining distractor groups. If the number of distractors exceeds the required amount, the set of selected distractors needs to be reduced.

generates a pool of possible distractors consisting of all well-formed variants with corresponding feedback and half as many ill-formed variants as well-formed ones in a first step. Only if the required overall amount of distractors is not met will the pool be supplemented by additional forms in the second step, as demonstrated in Figure 3.3.4a. These consist of well-formed variants without feedback at first priority, ill-formed variants with feedback at second priority and ill-formed variants without feedback at third priority. Unless all generated forms of one group are used as distractors, the forms are chosen randomly. The amount of distractors may fall short of the requested number if not enough variants could be generated. If the pool of options determined in the first step contains more distractor candidates than desired, as is the case for the example in Figure 3.3.4b, the second step consists in determining a sub-sample of the candidate pool instead of enriching it with additional forms. A random selection of as many distractors as specified in the web interface is thus taken.

3.3.6 DOM manipulations

Once the DOM has been indexed and all exercise components have been generated, the HTML structure can be transformed into the format required by the H5P `semantics` specification.



```
<tag>
sentence 1
   <tag>
      <tag res="uri" />
      sentence 2
      <tag res="uri" />
      <tag>sentence 3</tag>
   </tag>
</tag>
```

Figure 3.3.5: Deletion of removed sections from the DOM

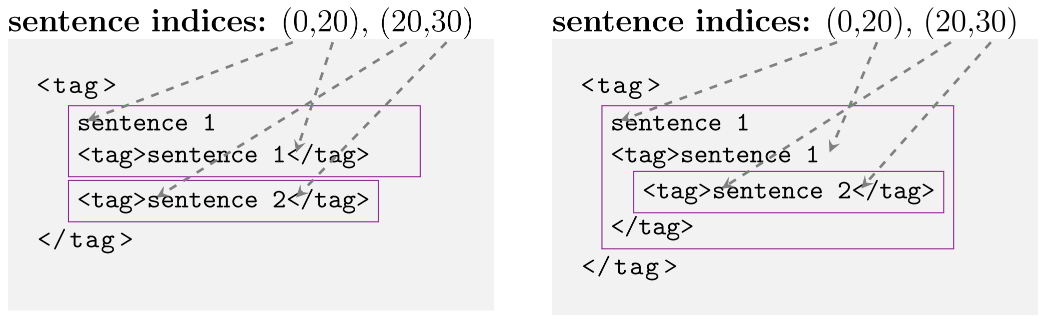
The *tags* may refer to any HTML tag. Similarly, the *res* attribute may constitute any HTML attribute referencing a resource. The plain text colored in red has been removed by the instructor in the configuration panel. The elements within the red rectangle are removed by FLAIR$_{\text{ExGen}}$. The resource marked in bold is also removed even though it is not inside the deleted text section but at its boundary.

The parts of the plain text that were deleted by the instructor are identified and removed along with any enclosing HTML elements without preserved plain text content. As illustrated in Figure 3.3.5, the removed elements might contain resources at the plain text boundaries which the user may or may not have intended to delete. However, removing all enclosing HTML elements ensures that empty HTML elements with some visible manifestation are removed along with the text.

The remaining DOM is searched for resources. If the instructor has not enabled resource downloading in the exercise configuration, the paths are set to absolute URIs. Otherwise, processing depends on whether the file type is supported by H5P or not. If it is supported, the resource is downloaded into FLAIR's working memory, assigned a name consisting of the string `tempResourceName` suffixed with a unique sequential number, and the resource path is set to that name. If the file type is not supported, any relative URIs are replaced with absolute paths. It would also be possible to download such resources and assign them some supported default file extension like *jpeg*. However, while the MIME type, not the file extension, is supposed to be considered in order to determine the file type, some browsers may apply an opposing strategy (Mozilla, 2005; The MITRE Corporation, 2020). Files with manipulated file extensions would therefore not always be processed correctly. FLAIR$_{\text{ExGen}}$ also searches all style elements for resources by means of Regular Expressions. The identified items are then processed analogously.

The next step requires replacing exercise targets with the widgets for the selected exercise type. Since the widgets themselves are defined by the H5P content types and inserted at rendering runtime, the exercise targets are merely replaced at generation runtime with the placeholder elements expected by the H5P implementations. As we have modified our content types to use inline target specifications with referenced feedback fields, the placeholders consist of the correct answer enriched with the reference ID, all enclosed in asterisks. The reference ID is only given if incorrect answers in terms of target hypotheses for FiB exercises or distractors for SC tasks have been specified. The incorrect answers and corresponding feedback are accordingly bundled into JSON objects. MtW exercises impose some additional requirements on target specifications. Although according to the `Mark the Words` type description, only single-word items can be used as targets, the implementation uses spaces to determine word boundaries. We therefore replace any space occurrences within an exercise target with non-breaking spaces which have the same surface form but a different ASCII representation. This way, `Mark the Words` exercises can use multi-word expressions as exercise targets without further modifications. An additional requirement of this content type is for asterisk-enclosed target words to be surrounded by spaces. For FiB exercises, the generated brackets also need to be included in the DOM. They are inserted as simple text elements right after the placeholder specification of the corresponding target word.

The last challenge in this vein consists providing the HTML string in the form expected by our custom H5P content types. To this purpose, we replace the relevant HTML special symbols such as angled brackets, ampersand and quotation marks with the corresponding replacement string. All plain text elements are extracted from the HTML string, placed inside HTML elements to which we assign a unique ID and replaced in the DOM with a placeholder referencing that ID. Plain text is thus cleanly separated from HTML elements.



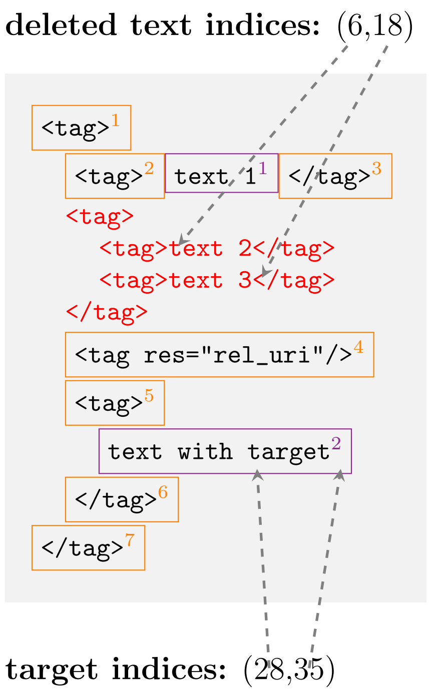
(a) Non-overlapping sentences

(b) Overlapping sentences

Figure 3.3.6: Merging of overlapping sentences

The tags in the XML structures may refer to any HTML tag. The arrows point to the positions in the HTML plain text corresponding to the respective indices. The boxes enclose all HTML tags of the sentences necessary to display them as exercise instances.

For DD exercises with multiple sub-tasks, the exercise segments need to be assigned a sentence ID so that they may be represented as sub-exercises on the target platform. Such sub-exercises need to constitute valid, complete HTML elements. As illustrated in Figure 3.3.6, it is therefore possible that some segments have to be merged in order to fulfill this requirement. If the enclosing HTML elements of two sentences do not overlap, such as the illustration in Figure 3.3.6a, they are kept as individual exercises. If they do, however, such as is the case for the example in Figure 3.3.6b where the enclosing HTML elements of one sentence contain the other sentence, they are merged into a single exercise with the scope of the enclosing sentence.



```
<tag>1
  <tag>2 text 11 </tag>3
  <tag>
    <tag>text 2</tag>
    <tag>text 3</tag>
  </tag>
  <tag res="rel_uri"/>4
  <tag>5
    text with target2
  </tag>6
</tag>7
```

(a) HTML input

```
{
  "textElements":
     "<span data-id=\"1\">
     text 11 </span>
     <span data-id=\"2\">
     text with *target: f1 *2
     </span>",
  "htmlElements": [
    "<tag>1 <tag>2 ",
    1,
    "</tag>3
     <tag res="absolute_uri"/>4
     <tag>5 ",
    2,
    "</tag>6 </tag>7 "
  ],
  "feedback": [
  {
    "incorrectAnswers": [
      {
        "answerText": "hypothesis1",
        "answerFeedback": "message1"
      },
      {
        "answerText": "hypothesis2",
        "answerFeedback": "message2"
      }
    }
  ],
  "feedbackId": "f1"
}
```

(b) JSON output

Figure 3.3.7: Conversion of HTML into JSON

The tags in the HTML DOM may refer to any HTML tag. Similarly, the *res* attribute may constitute any HTML attribute referencing a resource file. The HTML elements marked in red are identified by their indices and deleted. Exercise targets, also identified by the corresponding construction indices, are enclosed in asterisks. Indices in colored boxes of the HTML input map to indices of the same color in the JSON output. The arrows point to the positions in the HTML plain text corresponding to the respective indices. Linefeeds within JSON strings are added in this representation for better readability.

In order to provide a more visual representation, Figure 3.3.7 summarizes the most relevant steps to obtain a JSON specification from the HTML document up to this point: The text parts removed by the instructor are deleted (*red*), and the remaining DOM is split into HTML tags (*orange*) and text elements (*purple*). The text elements receive markings for targets and are embedded into *span* elements with a unique ID (*green*). This ID is referenced in the list of HTML elements where it serves as a placeholder for the text element. The targets are enriched with a feedback ID (*blue*) which is referenced in the corresponding feedback field. Relative URIs in resource HTML tags are replaced.

Chapter 4

Evaluation & Discussion

In order to determine the effective contribution that FLAIR$_{\text{ExGen}}$ provides to the domain of automatic exercise generation, we assessed our system qualitatively as well as quantitatively.

4.1 Methodology

In our evaluation, we compare the scope of FLAIR$_{\text{ExGen}}$ with that of related work. In addition, we analyze the results of a technical analysis of our system's robustness and performance. As large-scale evaluations including student and teacher participants or subject matter experts are beyond the scope of this thesis, we did not evaluate such criteria.

4.1.1 Comparison with related work

We assessed a number of quality criteria highlighted by the developers of existing exercise generation systems as well as features considered relevant to such tools by the authors of related work. The evaluation criteria encompass features of versatility and coverage.

Versatility The more flexible the exercise generation tool and the generated exercises are, the more widely applicable they are to a range of learning scenarios.
The ***portability*** of the exercises determines their applicability to real-world settings.

By making the tasks independent of a specific platform, high portability increases the likelihood of the tool being adopted by users as the exercises may be integrated into an established e-learning system supporting the exercise format (Schwartz et al., 2004; Naiara Perez Miguel, 2017; Aldabe et al., 2006).
High ***configurability*** of the generated exercises allows to adjust exercise complexity in order to account for the learners' proficiency levels, strengths and weaknesses (Naiara Perez Miguel, 2017). In order to supply learners with a range of scaffolding exercises, allowing users to specify the sequence in which exercises are displayed should also be considered (Antoniadis et al., 2004). As pre-generation configurations may not always be fine-grained enough to produce the desired output, provided functionalities to post-edit generated exercises also factor into this criterion (Hoshino and Nakagawa, 2008).

Coverage of learning scenarios In order to be accepted by a broad public, the generated exercises need to be widely applicable.
The tasks need to support practice for a significant proportion of the requirements established in the curriculum a learner follows. Since we focus on learners at beginner and lower intermediate levels of English, we measure the coverage of the ***pedagogical targets*** our system supports (Perez-Beltrachini et al., 2012; Antoniadis et al., 2004) against those covered by the 7th and 8th grade curriculum in Baden-Württemberg high schools.
The ***skill acquisition perspective*** determines whether the exercises aim at developing declarative or procedural knowledge. Declarative knowledge is typically acquired first through explicit explanations to then be automated into procedural knowledge (Ortega, 2014). While exercises may in general target both types of knowledge, instructions to acquire declarative knowledge do not need to be available in great variety. Providing large amounts of practice material for proceduralization, on the other hand, constitutes an important bottleneck. It should therefore constitute the primary target of automatic exercise generation.
While supported ***languages*** are of little consequence to the prototypical user interested in a single language, this criterion affects overall popularity of the tool and acceptance from users working with multiple languages (Naiara Perez Miguel, 2017).
The range of exercise types influences the targeted ***language use*** in terms of production and comprehension (Aldabe et al., 2006). This criterion is usually considered

in tandem with the support for practice of spoken and written ***modalities*** (Bodnar and Lyster, 2021). The 7-8th grade curriculum in Baden-Württemberg requires the entire spectrum of comprehension and production of written and spoken language to be covered (Ministerium für Kultus, 2016).
Since support for building a ***learner model*** allows to monitor the learner's progress and thus provide material adapted to the learner profile, this is considered a quality criterion by various authors (e.g. Hoshino and Nakagawa, 2008; Volodina et al., 2014).

4.1.2 Technical evaluation

The assessment of robustness and performance relies on objective measurements. They were carried out on exercises generated for samples of representative documents. Since sampling of web pages varied slightly depending on the feature being assessed, the methodological details are outlined separately in the following descriptions of the evaluated features.

Robustness and correctness The more reliably all features of the exercises can be generated correctly, the less manual post-processing effort is required. High reliability will also allow use case scenarios where this manual step is omitted, thus opening opportunities for full automation and independent, user-adaptive learning. The definition of ***grammaticality of the generated exercises*** differs from one author to another depending on the focus of the assessed system. Perez-Beltrachini et al. (2012) determine whether the generated sentences are syntactically and morphologically correct. Hoshino and Nakagawa (2008) assess the discrimination value which allows to discriminate between strong and less proficient learners based on an exercise, as well as the difficulty level of the generated distractors. Both EWR and Seneff (2007) and Lee et al. (2016) consider exercises incorrect if at least one of the distractors generates a correct sentence. Since Lourenco (2015) generates her passive transformation exercises by transforming a passive into an active sentence, incorrect tasks constitute incorrectly formed active sentences. As we focus on targets ranging from simple to more complex linguistic constructions, the most crucial aspect of correctness for FLAIR$_{\text{ExGen}}$ concerns the correct identification of target constructions. While this depends to a large extent on the precision at which

FLAIR identifies constructions, an additional aspect to consider consists in the performance of our strategy to identify exercise targets based on multiple, overlapping constructions extracted by FLAIR. In order to assess this criterion, we sampled up to 10 occurrences for each type of exercise target from 100 arbitrarily selected web pages. Identical occurrences of target constructions were not considered and only web pages which contained at least one construction were taken into account. Recall is not essential to exercise generation as an exercise text may well contain occurrences of the targeted constructions which are not turned into exercise targets but merely part of the textual context. We therefore only determined precision values for the identified constructions.

Another dimension of robustness concerns the identification of adequate documents. Authentic texts need to contain a reasonable amount of target constructions in order to be suitable for exercise generation. While Perez-Beltrachini et al. (2012) focus on the number of generated targets in their evaluation, it makes sense to break this down into two components for our application: the number of identified documents suitable for exercise generation and the ratio of successful target generation from constructions. The assessment of ***document suitability*** comprises two aspects. The first aspect concerns the rendering of the document's markup on the target platform. While we aim to preserve as much of the original HTML markup of the web texts as possible, not all web documents are equally well displayed. However, as long as the exercises are usable, we consider the documents suitable for our application. The range of supported documents influences the usability of the tool for possible use case scenarios as the user may or may not be restricted to particular documents, corpora or document formats (Knoop and Wilske, 2013; Antoniadis et al., 2004; Schwartz et al., 2004). Wu et al. (2009) consider journals and newspapers to provide the most suitable contents for language learning activities due to the editorial process to which they are subjected so that they constitute sources of generally correct language material. We therefore determined whether the most popular English news sites can be used with FLAIR$_{\text{ExGen}}$. To this purpose, we evaluated articles published by the most frequently used online news sites according to a recent report by the web analytics provider SimilarWeb[1] (Majid, 2021) and the most trusted news sites by users according to a report by the consumer

[1]https://www.similarweb.com/de/

profiler GlobalWebIndex[2] (Global Web Index, 2019). We considered the 5 highest-ranked English sites from both reports and sampled three main page articles from the day of the evaluation. We also included Wikipedia as a non-scientific document source (Knight and Pryke, 2012) as, largely due to their length, they tend to appear among the highest-ranked documents in FLAIR's construction-weighted search results. In addition, we assessed the applicability of FLAIR's file upload to exercise generation. The second aspect in the evaluation of suitable documents concerns the occurrence of targeted grammatical constructions. For this assessment, we considered those pedagogical goals covered by the 7th and 8th grade curriculum with the aim to determine whether our exercise generation can, in combination with FLAIR's document ranking, identify documents for which according exercises may be compiled. We examined the highest-ranked results returned for the search term *Euro 2020*, referring to the currently popular topic of the 2020 UEFA European Football Championship, both without restricting the search sites and on the popular news site Reuters.

The ***target generation ratio*** is here defined as the ratio of the number of target constructions in the generated exercises to the number of target constructions predicted in $\text{FLAIR}_{\text{ExGen}}$'s exercise configuration panel. A perfect ratio of 1 indicates that all predicted target constructions could be turned into an exercise target. If indexing of the HTML plain text or post-processing of the constructions renders an occurrence unusable for the exercise, the ratio decreases. In this evaluation, *Euro 2020* served as search topic once more with search settings set to 50 results and no site restrictions. Construction weighting was adjusted for each exercise topic in order to emphasize those linguistic constructions which are targeted by the respective topic. The generation ratio was calculated for all supported exercise topic-type combinations.

Perez-Beltrachini et al. (2012) argue that the ***variability of exercises*** in terms of diversity in target constructions and in syntactic and morphosyntactic contexts is crucial in order for learners to master a grammar topic in a flexible manner. We focus on the variability of target constructions which can be measured objectively by calculating the type-token ratio (TTR) of the exercise targets. We define this measure as the number of distinct targets per overall exercise targets. A value of 1 indicates that all occurrences are distinct from each other, thus providing maximum

[2]https://www.gwi.com/

variety in the targeted forms. The ratio decreases as the number of non-unique exercise targets increases. For the evaluation, we generated exercises for all exercise topics and the same sites that we used to assess supported web sites. All exercise configurations were set to the default settings. The documents were selected based on the searches for *Euro 2020* with construction weights set accordingly in order to assess documents with highest possible numbers of exercise targets. TTRs were calculated for each generated exercise.

Robustness is also an important factor in evaluating the ***generated feedback*** (Meurers et al., 2010; Naiara Perez Miguel, 2017; Lourenco, 2015). The ratio of exercise types and topics supporting feedback constitutes an objective measure. In addition, the number of items per exercise for which feedback can be generated needs to be considered. To this purpose, we used the FiB exercises that we generated for the *Euro 2020* evaluation topic, thus covering all exercise topics. For each generated exercise, the number of targets with automatically generated feedback was compared against the overall number of targets.

Usability Since end users of our tool comprise both instructors compiling exercises and learners working on them, an assessment of user experience needs to take into account all components of the workflow that require the active involvement of the instructor or the learner.

In order to determine whether the performance is acceptable, measuring the average ***execution time*** to generate an exercise from configured settings for a pool of sample exercises provides an objective quality feature (Aldabe et al., 2006). We first examined the effect size of using different exercise types by comparing the execution times of our four task types on the same document and exercise topics. Since differences were in the millisecond range, measurements of execution times for further assessments were consequently performed only on FiB exercises as they are supported by all exercise topics. We again generated exercises for Reuters, BBC and CNN articles on *Euro 2020*, yet with resource downloading as well as feedback generation activated.

4.2 Results

The following sections outline the results of our evaluation concerning versatility, coverage of learning scenarios, robustness and correctness, and usability.

4.2.1 Versatility

Portability Thanks to using the popular H5P output format, the generated exercises can be integrated into any platform supporting H5P. Although integration via LTI or a plugin is currently limited to 6 platforms, H5P content may also be embedded into or linked to from any custom web page (Joubel, n.d.b). If users do not already employ a target platform, the freely accessible H5P platform[3] may be used by instructors to edit the generated exercises as well as by students to work on them. Building a learner model, however, will not be able in this case.
Of the existing exercise generation tools reviewed in Section 1.1, only the web plugins VIEW and CLozeFox, and KillerFiller and MIRTO provide the exercises in a portable format. The plugin approach has, however, the insurmountable drawback of not allowing post-editing of the generated tasks. The format used by MIRTO only supports Cloze tasks. KillerFiller fulfills our requirements on the output format best as it conforms to the SCORM standard. However, while there are tools to convert H5P files into SCORM content, they do not support an inverse conversion from SCORM into another format. H5P is therefore not only better manageable than SCORM but also more widely applicable.

Configurability Since we have extended FLAIR's web interface to encompass a configuration panel for exercise generation, the exercises are highly customizable. Configuration options concerning exercise targets include fine-grained specification of properties of the targeted linguistic constructions such as POS, comparison form and comparison level for Comparison exercises or regularity of formation, tense and aspect for Past tense exercises. Bracket contents for FiB exercises and distractor properties for SC tasks can also be determined by the user. Concerning the textual exercise context, features of the embedding sentence such as interrogative or negated qualities can be configured. Although FLAIR$_{\text{ExGen}}$ does not directly re-

[3]https://h5p.org/

late configuration settings to complexity measures, users can thus manipulate the difficulty of exercises. In addition, general configuration features allow to remove undesired parts of the plain text. The resulting exercise may thus be tailored to contain only those sections of the document which target the topic of interest. In case an instructor wishes for even more fine-grained specifications, the authoring interface inherent to all H5P exercises allows post-editing of the generated tasks so that context or target elements may still be removed or edited once an exercise has been generated. The `Quiz` content type allows to sequence exercises at configuration time. The order of the exercises within this content type corresponds to the order in which they were configured in FLAIR$_{\text{ExGen}}$'s web interface. It cannot be manipulated by an instructor in the exercise configuration panel, but may be altered in the H5P authoring tool. There, exercises may also be removed and new exercises can be added to a Quiz. An alternative approach consists in using the sequencing mechanisms of the target platform such as Moodle activities which allow to flexibly bundle various exercises and other contents into a unit.
The most highly configurable applications of related work include MIRTO, Sakumon and the Language Exercise App. Since Sakumon is an assistant system, instructors have high influence on the target items and distractors of the MC tasks as they need to select them manually from the tool's set of suggestions. Considering that exercise generation has not yet been fully automated in that system, it is only marginally comparable to FLAIR$_{\text{ExGen}}$. MIRTO grants the instructor considerable influence on the targeted constructions and exercise components such as bracket contents of FiB exercises. In addition, it allows to specify interactive supportive elements such as links to lookup pages. The Language Exercise App also allows instructor to specify target constructions, bracket contents and distractors. MIRTO and the Language Exercise App are thus very similar to our approach in terms of configurability. However, neither of these two tools supports post-editing of the generated exercises so that it is not possible to manually adjust features which cannot be managed through pre-generation configuration.

4.2.2 Coverage of learning scenarios

Pedagogical targets Table 4.2.2 summarizes the curriculum topics scheduled for our target group with indications of whether they are supported by FLAIR$_{\text{ExGen}}$: Al-

Curriculum topic	Supported
(zero) article	
Conditional clauses I and II	•
Adverbial clauses	
Relative clauses	(•)
Question tags	
Contrasting present perfect and simple past	•
Present perfect progressive	•
Past progressive	•
Past perfect	•
Past perfect progressive	•
Conditional forms	(•)
Adverbs of manner and degree	(•)
Comparison of adverbs	•
Active voice	•
Passive voice	•
Reported speech	

Table 4.2.2: Support for curriculum topics

The curriculum topics for which FLAIR$_{\text{ExGen}}$ can generate exercises are marked with a dot. Dots in parentheses indicate topics that are only partially supported.

though no exercises to practice the use of articles can be generated, we offer extensive support for both types of conditional clauses through FiB, SC and DD exercises. Adverbial clauses are not supported. With respect to relative clauses, generated exercises can target the relative pronouns listed in the curriculum. However, the configuration options do not allow to distinguish between defining and non-defining clauses. Contact clauses are not supported at all as they do not contain any relative pronoun. Exercises for question tags cannot be generated either. However, the topic of Past tenses covers the entire spectrum of corresponding topics proposed in the curriculum including present perfect, simple past and past perfect in simple and progressive aspect. Thanks to the high configurability of the exercises, contrasting present perfect and simple past can also be supported by including only these two tenses in the generated exercises. Conditional forms may appear in exercises on Conditionals, yet without specifically targeting formation and not allowing for restriction to specific tenses. The focus of exercises targeting adverbs is on comparison forms. No fine-grained distinction into types of adverbs is made in the configuration

settings. Exercises specifically targeting adverbs of manner and degree will therefore need manual revision to remove other types of adverbs. They will only contain comparative and superlative forms as exercise targets but no adverbs in their positive form. Active and passive voice is well supported since FLAIR$_{\text{ExGen}}$ dedicates its own exercise topic to this pedagogical goal. As by-clauses are not very frequent in authentic web texts, however, no configuration option is given to specifically target passive sentences containing a by-agent. The last item on the curriculum's agenda, reported speech, is not included in the exercise generation.
Coverage of exercise topics in related work is highest if configurability of the system is high. MIRTO and the Language Exercise App are therefore most likely to support the curriculum topics. MIRTO relies only on a morphological tagger and a lemmatizer or stemmer in order to identify targets (Antoniadis et al., 2004). Complex linguistic constructions such as conditional sentences or relative clauses can therefore not be targeted by the exercises. Similarly, the Language Exercise App uses a POS tagger in addition to a morphological analyzer and thus faces the same limitations concerning complex constructions. This is a clear advantage of FLAIR$_{\text{ExGen}}$ since our tool supports such constructions not only in the exercise generation itself, but also by identifying documents containing them.

Languages The current implementation supports only English for exercise generation. However, as both FLAIR and the FeedBook microservice can also process German input, extending FLAIR$_{\text{ExGen}}$ to that language is possible without cutting back on the features these external components offer. Since English, with 1.5 billion learners worldwide, is by far the most studied language, our tool addresses a sizeable target group already (Noack and Gamio, 2015; ICEF, 2019). By extending FLAIR$_{\text{ExGen}}$ to German, we would incorporate the fourth most prominent language with an additional 14.5 million learners.
Existing exercise generation systems support up to 22 languages in the case of Killer-Filler. It is not surprising that this should be the tool covering the largest number of languages since it focuses on vocabulary practice. Extending the NLP processing required for such tasks to other languages is considerably less complex than the implementation effort necessary with grammar exercises.

Language use and modalities The generated exercises do not support audio components, either speech or listening comprehension. Although integrating such elements into a H5P content type is possible, the focus is currently on written language. The range of supported exercise types of our tool does, however, cover both comprehension and production practice. While production of language is required for FiB exercises, the other three types target language comprehension (Schuetze, 2018). Moreover, as the exercises are embedded into authentic text contexts, comprehension may also be required for FiB exercises in order to successfully disambiguate. FiB exercises are available for all topics, SC exercises for all but tasks on Passive, DD exercises support all topics except for Simple present and MtW exercises can be generated for all topics but Conditional and Passive. All topics thus support at least one exercise type with focus on comprehension practice and one focusing on production.
Since most of the applications reviewed in the related work use authentic contexts to some degree, production and comprehension both play an important role in the exercises generated by most existing tools. Supported exercise types vary greatly, yet SC and FiB exercises, which FLIAR$_{\text{ExGen}}$ also incorporates, constitute the most widely adopted types.

Skill acquisition perspective As the purpose of our system, as well as that of related work, is to provide large amounts of practice exercises in varied contexts, the focus of the generated exercises is on acquiring procedural knowledge through ample practice opportunities. Since FLAIR$_{\text{ExGen}}$ providees scaffolding feedback with links to additional references, declarative knowledge may in addition be obtained while working on the exercises. This is especially valuable if lacking declarative knowledge is not directly related to the topic targeted by the exercise but to linguistic background knowledge necessary or helpful for successful exercise completion.

Learner model Since all H5P content types from which the custom types are derived generate xAPI statements on a range of learner actions, building a learner model is inherently supported. The recorded learner actions typically include answers given by the learner and the number of attempts. The task of processing the information sent in these xAPI statements resides with the target platform. Whether the learner model is actually populated with the data therefore depends

on the publishing platform.
Considering the tools of related work that have so far shown the greatest similarities to our application, MIRTO does not incorporate a learner model. Only Cloze exercises can be exported in Moodle CLOZE syntax in order to be integrated into Moodle and use the learner model of that platform. The Language Exercise App does not support a learner model at all.

4.2.3 Robustness and correctness

Grammaticality As illustrated by the boxplots in Figure 4.2.1, the precision of construct identification differed considerably between the exercise topics as well as within most topics.

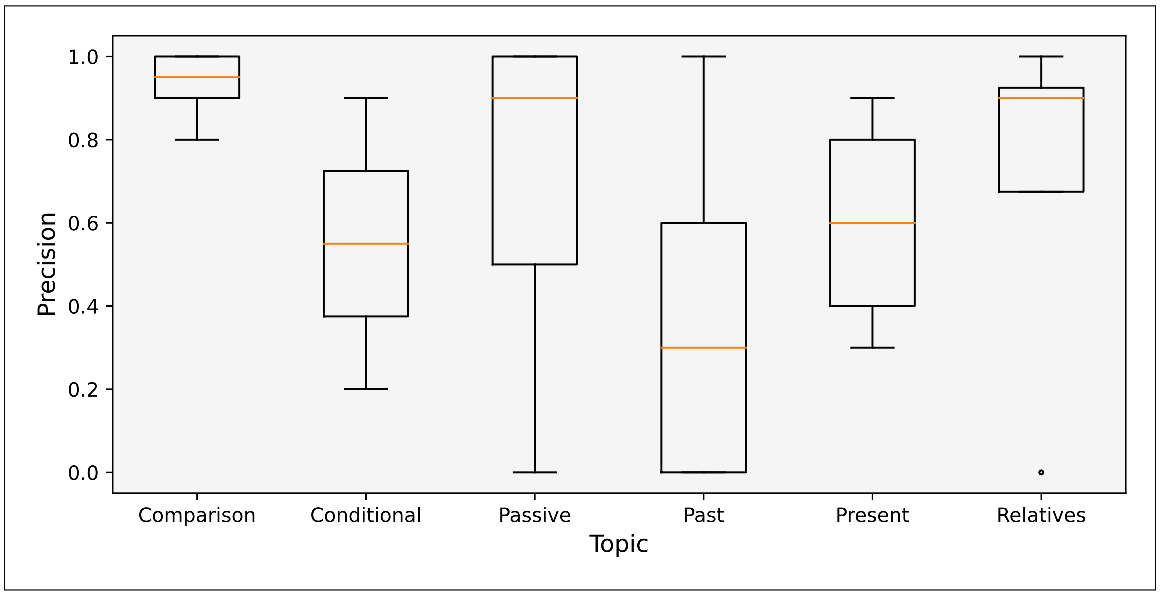


Figure 4.2.1: Boxplots for precision of construction identification

The boxplots were compiled from the evaluation results presented in Table A.0.2 in Appendix A.

Comparison constructions all obtained fairly high precision values. As the exercise targets of this topic correspond directly to constructions identified by FLAIR, errors are already introduced by FLAIR's existing construction extraction algorithm. They are mostly attributed to incorrectly assigned POS tags and occur for example with non-standard English such as the archaic English used in Example 10a or when adjectives and adverbs are confused such as is the case in Example 10b. Noticeable is also the shortage of findings for synthetic superlative adverbs. Almost all occurrences of these adverbs target the form *most*. Since construction weighting is based on overall occurrence counts of constructions and not on distinct occurrences, such excessively frequent forms may lead to high ranking positions of documents which introduce little variety in the targeted forms.

(10) a. "I can not dance unless thou **leadest**." (Wikipedia contributors, 2021c)
*Label: **Adjective (superlative, synthetic)***

b. "It differs from the much more famous "English Slow Waltz" by having much **faster** 180 beats per minute and was the first who introduced "closed hold" between performers." (Dance Facts, 2021)
*Label: **Adverb** (comparative, synthetic)*

The two ***conditional*** types are distinguished by FLAIR as well. While real conditionals were detected at high precision, most findings of unreal conditionals are in fact real conditionals, although they feature certain grammatical characteristics of unreal conditionals. While Example 11a clearly constitutes a real conditional in past tense, the main clause in Example 11b would indeed indicate an unreal conditional, were it not for the present tense used in the if-clause.

(11) a. "Balloons commonly had a crew of two, equipped with parachutes,[...] so that if there **was** an enemy air attack the crew **could** parachute to safety." (Wikipedia contributors, 2021g)
*Label: Conditional (**unreal**)*

b. "If Mr. Schumer **is** successful, it **would** advance the largest investment in roads." (Fandos, 2021)
*Label: Conditional (**unreal**)*

Performance with ***active and passive*** constructions differed considerably between the individual constructions. Tenses in simple aspect were rarely mislabelled for both active and passive voice. The rather high number of wrong labels for active simple present occurrences can be attributed to nouns which were tagged as verbs such as in Example 12a. The identified constructions for future simple in passive voice, such as the one given in Example 12b, all actually have active aspect with the infinitive consisting of the verb *be*, but no past participle present. As Example 12c illustrates, incorrect findings for future perfect consist of segments containing both a clause in present perfect and another one in future simple tense. For progressive aspect in combination with passive voice, out of the 9 occurrences that could be detected, only one was labelled correctly. While passive voice was attributed correctly for present perfect progressive, the occurrences did not have progressive aspect. For future perfect progressive, on the other hand, the tense was correct yet the findings were not in passive voice. With past perfect progressive passive, both scenarios occurred, the findings either having the correct aspect but incorrect voice or correct voice but incorrect aspect. Out of the 52 occurrences with progressive aspect for active voice, on the other hand, 43 were correct. Wrong labels were here only assigned in combination with perfect aspect. With past perfect progressive, none of the findings included the past participle of *be* and with future perfect progressive, the present participle was missing.

(12) a. "If two or more teams were equal on points on completion of the group **matches**, the following tie-breaking criteria were applied:" (Wikipedia contributors, 2021f)
*Label: **Present simple** (active)*

b. "Also **featured will be an illustration** of Pride and Prejudice's Elizabeth Bennet reading some letters and a quotation from the same novel:" (Kelly, 2017)
*Label: Future simple (**passive**)*

c. "Devices that **have been discovered but have not yet been onboarded and secured by Microsoft Defender for Endpoint will** be listed." (Caparas, M.J. and Davis, Chris and Hanson, Diana and Simpson, Daniel, 2021)
Label: Future perfect (passive)

The issues encountered with passive constructions that are not related to the active-passive distinction could also be observed with constructions for ***past tenses***. An interesting error case constitute infinitives occurring with modals such as in the sentence given in Example 13a. While we are only interested in conjugated forms, infinitives tend to occur reasonably frequently especially with rare constructions.

However, the most prominent cause for incorrect labelling which was responsible for the overall poor performance in this category, with only 103 out of 232 occurrences labelled correctly, consists in the distinction between regular and irregular verbs such as illustrated in Example 13b. This concerns in particular tenses containing an auxiliary verb which is always irregular, such as *had* with past perfect or *been* with present perfect progressive. The identification of any irregular form within the construction triggers the *irregular* label even though the participle may be regular. Incorrectly assigned interrogative annotations are yet another issue. FLAIR marks entire sentences as questions whereas we are only interested in interrogative forms. Verbs which occur as sub-clauses to a clause containing an interrogative form, such as in Example 13c, are therefore recognized as being contained in a question and assigned the according annotation. Equally problematic are statement questions such as in Example 13d which do not follow the interrogative syntax yet are labelled as questions by the Stanford parser that FLAIR employs. Negation is incorrectly attributed when the verb construction is followed by a negation such as illustrated in Example 13e.

(13) a. "Also for your personal situation, do you consider that doing 3 years of post-doc instead of 3 of PhD, would**n't have given** you a good enough CV?" (Marshall, 2018)
Label: ***present perfect*** *(question, negated, irregular)*

b. "These factors enabled Hindenburg and other senior German leaders to spread the story that their armies **had not really been defeated**". (Wikipedia contributors, 2021g)
Label: past perfect (statement, negated, ***irregular****)*

c. "How was it possible that I **had not noticed** this the night before?" (Poe, 2013)
*Label: past perfect (****question****, negated, irregular)*

d. "He **didn't raise** the issue of service appointments and so on?" (Wikiquote, 2021)
*Label: simple past (****question****, negated, irregular)*

e. "Funk style of hip-hop **was popularized** not only by its music but also by the worldwide acceptance of the famous dance style called Electric Boogaloo, which was originally promoted in the 1970s by the dance

group of the same name." (Dance Facts, 2021)
*Label: simple past (statement, **negated**, irregular)*

Performance for ***simple present*** was slightly better with 48 out of 80 occurrences labelled correctly as these constructions do not distinguish between regular and irregular forms. However, the issues of interrogative and negated forms also appeared with this tense. Third person singular was largely assigned correctly with occasional mislabelled instances occurring mostly with negations, such as the one in Example 14a.

(14) a. "Capitalism without competition **isn't** capitalism." (Shear, 2021)
*Label: **present simple** (statement, negated, **not 3rd person**)*

Relative pronouns performed very well for the most common pronouns *who*, *which* and *that* with 28 out of 30 occurrences labelled correctly. Incorrect instances consisted in misclassifications of a wh-determiner (Example 15a) and of a conjunction (Example 15b). Only occurrences categorized as relative pronouns other than these three pronouns were throughout incorrect.

(15) a. "As there were seven teams from League C (three group winners and four non-group winners), the three group winners were placed in Path C, while a draw decided **which** of the four non-group winners was also placed in Path C." (Wikipedia contributors, 2021e)
*Label: **Relative pronoun** (which)*

b. "The same idea, **that** dance arises from musical rhythm, is still found in renaissance Europe in the works of the dancing master Guglielmo Ebreo da Pesaro [...]." (Wikipedia contributors, 2021c)
*Label: **Relative pronoun** (that)*

Suitable documents For the evaluation of support for visual display of different web sites, 8 news sites were considered. Their relevance is highlighted by the fact that some of the sites were shared among the two independent rankings provided by SimilarWeb and GlobalWebIndex, assessing frequency of visitors and trustworthiness respectively, on which the selection of evaluated sites was based. This is illustrated in Table 4.2.4 which summarizes the findings for the assessed sites. It shows that of the 9 evaluated document sources, 7 provide articles well suited for FLAIR$_{\text{ExGen}}$ and only two sites encounter issues in some instances if the entire document is used.

Web site	SimilarWeb Ranking	GlobalWeb-Index Ranking	Assessed articles	Usable
BBC	1	1	(BBC, 2021b), (Faulkner and Morton, 2021), (Lowen, 2021)	(●)
CNN	2	4	(Yeung, 2021), (Williams, 2021), (McGee, 2021)	●
The New York Times	3	4	(Crowley et al., 2021), (Gabriel, 2021), (Craig and Kasakove, 2021)	●
Daily Mail	4	21	(Tapsfield, 2021), (Bell and Graham, 2021), (Howes, 2021)	●
The Guardian	5	7	(Abdul-Ahad, 2021), (The Guardian, 2021b), (The Guardian, 2021a)	(●)
Reuters		2	(Winning, 2021), (Downie, 2021), (Pullella, 2021)	●
The Economist		6	(The Economist, 2021a), (The Economist, 2021c), (The Economist, 2021b)	●
Wikipedia			(Wikipedia contributors, 2021d), (Wikipedia contributors, 2021b), (Wikipedia contributors, 2021a)	●
Uploaded plain text				●

Table 4.2.4: Support for web sites

The rankings indicate the position attributed to the news sites for frequency of visitors and trustworthiness by SimilarWeb and GlobalWebIndex respectively. The usability has been determined globally for each site based on the assessed articles. Usable sites are marked with a dot. Dots in parentheses indicate site that are usable in some instances or when reduced to a selection of paragraphs.

Exercises generated from ***BBC*** articles do not display usable content at all. Except for the header, the H5P exercises cannot render any content, although no error messages are displayed in the browser's console so that the source of the issue cannot easily be identified. When using the configuration option to only use sections with text, however, usable content can be displayed, although styling elements feature a certain lack of aesthetics. Interestingly, BBC overview pages (e.g. BBC, 2021a) are displayed, yet with strongly distorted markup similar to the issues observed with exercises using only text sections of the articles, which makes them barely usable as exercise material. Analysis of the documents' source code identified the root causes of the issues in rigid CSS rules which only render correctly on the unaltered web page as well as JavaScript functions loading the page in lazy mode.
For ***The Guardian***, two of the three examined articles display very well, although the article's CSS styling also seems to interfere with the exercise elements defined in the H5P content types. The layout of the third article, however, is distorted so badly that the exercise is not usable. Yet when the first sentence, which does not belong to the article's main content, is cut out, the rest of the page is rendered correctly. Removing that sentence also removes the HTML elements containing disruptive CSS styles, thereby resolving the issue for the remainder of the document.
All ***CNN*** and ***The Economist*** articles generally display nicely with full visual context, although not all images are displayed and absolute positioning in some cases leads to overlapping elements. The exercises are usable for all tested articles.
FLAIR$_{\text{ExGen}}$ performed best with ***The New York Times***, ***Daily Mail***, ***Reuters*** and ***Wikipedia***, encountering only minor positioning issues and displaying all resources.
Uploaded text files can also be analyzed and displayed, although the intended context preservation is void in this case since the upload supports only plain text files without markup elements.

Focusing on the second aspect of suitability of documents, the number of contained target constructions, the results differ from one exercise topic to the other.
With FLAIR's standard settings, the search for Reuters articles returned no suitable documents containing ***conditional clauses*** either with or without construction weighting applied. When increasing the number of search results from 10 to 50 and setting high weights for the two conditional types, FLAIR$_{\text{ExGen}}$ was able to identify documents containing conditional sentences, although none of them contained

instances of both conditional types. When removing the restriction to articles from Reuters, this shortcoming was overcome. Construction weighting was, however, still required in order to identify suitable documents.

Analogous to these observations, suitable documents for ***relative pronouns*** both with and without site restrictions were only found when documents were weighted for relative pronouns. The number of targets in the identified documents was higher for site-unrestricted searches.

While the highest-ranked search results without construction weighting contained only simple past forms in our evaluation of ***past tenses***, individually increasing the weights for each tense allowed us to also identify suitable documents for past progressive, present perfect and past perfect. Documents containing present perfect progressive forms required increasing the number of search results in addition to construction weighting. Past perfect progressive forms could not be found for the search term under evaluation at all. Changing the search topic to *Trump* finally yielded results for the targeted construction, although the identified document contained only a single occurrence. A similar pattern was observed with site restrictions to Reuters articles, although an occurrences for each of the two problematic progressive forms could be found without altering the search term when the corresponding weights were increased.

Documents for ***adverb comparison*** practice are plentiful so that exercises could be generated in our evaluation on FLAIR's default settings for all or only Reuters documents. The number of targets within the documents, however, increased considerably when construction weighting was applied and sites were not restricted.

In the assessment of exercises on ***passive*** constructions, the highest-ranked documents for all sites and for only Reuters articles contained simple present and simple past forms. In order to also include passive sentences with present perfect forms for site-restricted searches, we had to increase weights for the targeted tenses and passive voice before relevant documents appeared at the top of the ranking. We encountered documents containing no by-agents, where all passive occurrences contained a by-clause, and also where some but not all of the constructions included a by-agent. This confirms our assumption that a large proportion of extracted passive sentences does not contain a by-agent, thus supporting our decision to not use exercises which require such a clause.

Topic	Type	Assessed document	# potential targets	# generated targets	Generation ratio
Conditional clauses	FiB	(Wikipedia contributors, 2021e)	52	46	.8846
	SC		52	44	.8462
	DD*		52	36	.6923
	DD		52	46	.8846
Relative pronouns	FiB	(Wikipedia contributors, 2021e)	61	61	1.0
	SC		61	61	1.0
	MtW		61	61	1.0
	DD*		61	61 (210)	1.0
	DD		61	61	1.0
Past tenses	FiB	(Daily Mail, 2021)	213	212	.9953
	SC		213	211	.9906
	MtW		213	212	.9953
	DD		213	212	.9953
Comparison	FiB	(Lange, David, 2021)	50	35	.7
	SC		50	35	.7
	MtW		50	35	.7
	DD		50	35	.7
Passive	FiB	(Smith, 2021)	236	231	.9788
	DD*		26	22 (44)	.8846
Simple present	FiB	(Blitz, 2021)	270	270	1.0
	SC		270	270	1.0
	MtW		270	270	1.0

Table 4.2.6: Target generation ratio

The evaluation comprises all exercise type-topic combinations that FLAIR$_{\text{ExGen}}$ supports. For DD exercises, a distinction between exercises spanning the entire document and tasks containing multiple sub-exercises spanning single sentences is made. The latter ones are marked with an asterisk (*). Some DD* exercises generate additional exercise targets from sentence parts which do not correspond to an original target construction. For these exercises, the number of generated targets indicates the number of target sentences containing an original target, and in addition the number of all target elements in parentheses.

Target generation ratio The results of our evaluation presented in Table 4.2.6 indicate that, while values between topics vary, they are generally reasonably high. For Relative pronouns and Simple present exercises of all types, all predicted target constructions could be turned into exercise targets. Values for Past tenses were only slightly lower at .9953 for FiB, MtW and DD exercises and .9906 for SC exercises. The lower index for SC exercises indicates that no distractors could be generated for some of the targets. Generation of Conditional clause exercises was most robust for FiB and document-spanning DD exercises with a generation ratio of .8846. For SC exercises, 84.62% of the predicted constructions could be used as exercise targets. DD exercises spanning single sentences proved most challenging with a generation ratio of .6923. Tasks on Passive sentences achieved very high values (.9788) on FiB exercises, which also include active sentences, and slightly lower generation rates (.8846) for DD exercises, where only passive sentences are targeted. All exercise types for Comparison tasks turned 70% of the constructions into exercise targets.

Variability of exercises As illustrated by the boxplots in Figure 4.2.2, the exercise targets are in general fairly varied for all exercise topics except Comparison tasks, the TTRs ranging from 0.13 to 1.0 with an overall average of .64 (SD=.27). The exercises thus generally constitute varied practice material which do not simply put exceeding emphasis on a small number of forms. The low ratios for Comparison exercises result from high frequencies of the adverbs *more* and *most*, a phenomenon already observed while evaluating grammaticality.

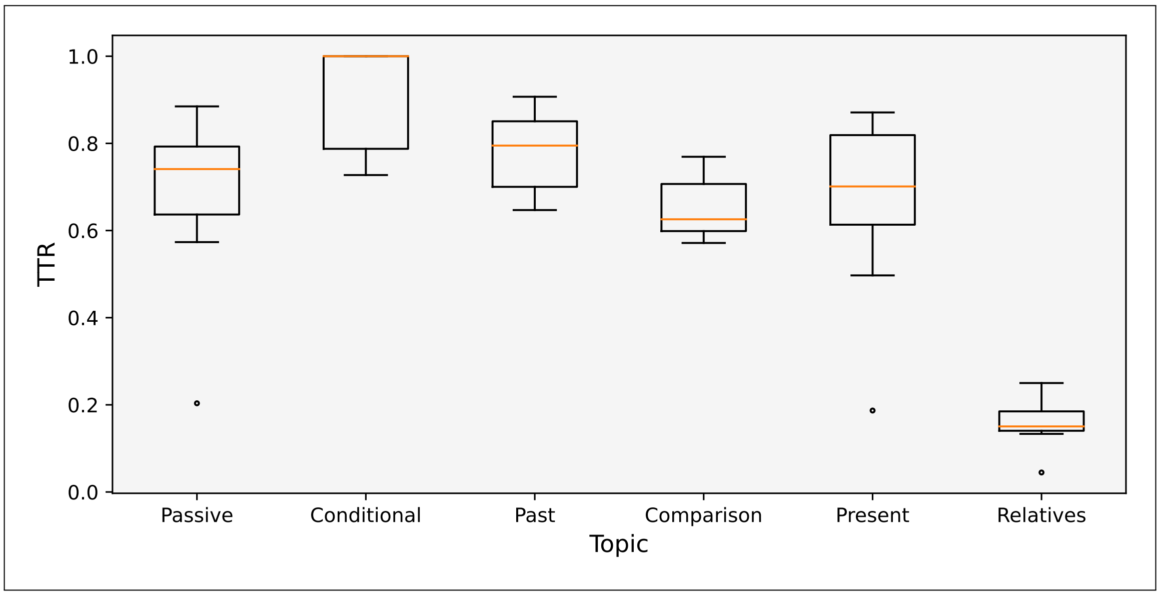


Figure 4.2.2: Boxplots for target generation ratios

Robustness and correctness of feedback Of our four exercise types, two support automatically generated feedback for all topics. For the evaluation of feedback in generated exercises, the batch size had to be reduced and failed requests repeated in some cases in order to receive a successful response instead of a timeout error from the microservice. Especially exercises on Passive and Tenses proved challenging for the FeedBook microservice. We do not further asses the timeout issues as we recognize that the microservice is still under development. With the adjusted evaluation setup in place, however, the generated exercises in general exhibited a high item-based success rate: on average, feedback could be generated for 90% of the exercise targets (SD=.22).

4.2.4 Usability

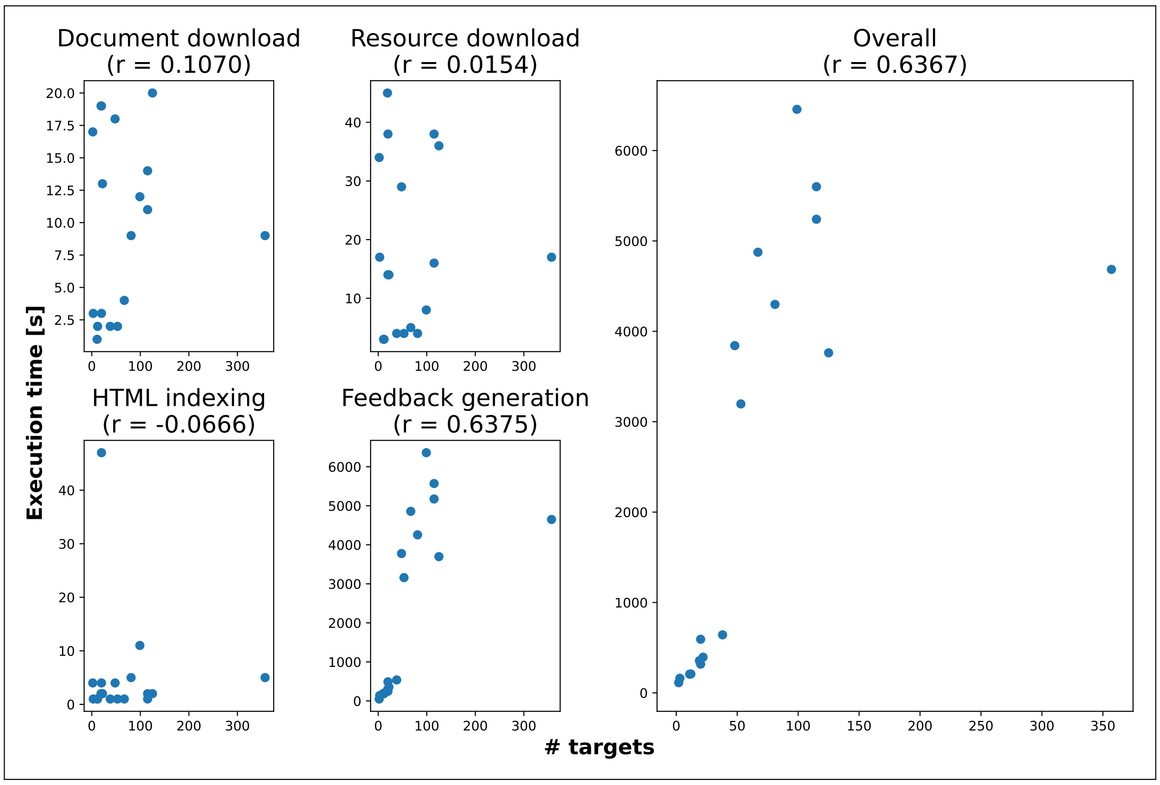


Figure 4.2.3: Correlation between number of targets and execution times

Overall execution times correlated strongly with the number of targets. The pattern is similar to that of feedback execution times which constitute the main contribution to overall times. The other processing steps have only weak correlations with the number of targets. Results for NLP analyses are not displayed as their processing times are in the microsecond range.

Execution time The measured execution times were split into times attributed to document download, HTML indexing, NLP processing, feedback generation and miscellaneous such as initializing the threads for parallel processing or creating H5P files from the generated content. As illustrated by the scatterplots in Figure 4.2.3, we observed a strong positive correlation between the number of exercise targets and overall execution time (r = .9053). The high correlation coefficient can be largely attributed to feedback generation times which comprise the major portion of execution times and increase with the number of targets. This is not surprising as feedback needs to be generated for each item individually. Although NLP processing times might be expected to depend on the amount of targets as well, corresponding execution times are in the microsecond range so that their impact on the overall processing time is negligible. Other than that, no particular performance bottlenecks were discernible. Overall processing times ranged from 113s to 6456s (mean = 2497.22, SD = 2336.95). When excluding feedback generation, this was reduced to 12-125s (mean = 53.83, SD = 28.88).

4.3 Discussion

We assessed quality criteria targeting the scope of FLAIR$_{\text{ExGen}}$'s applicability in real-world scenarios and that of suitable authentic documents, as well as the performance concerning the generated content and with regard to technical processing. Additional evaluations may be envisaged for future studies.

4.3.1 Scope of the tool's applicability

The evaluation revealed that FLAIR$_{\text{ExGen}}$ provides extensive support for portable, highly configurable exercises, thus constituting a versatile exercise generation system that supports a range of possible learning scenarios. It is targeted at instructional settings where a teacher introduces the topic, determines appropriate configuration settings and revises the generated exercises in order to edit potential errors. Since dynamic feedback is also provided by the exercises, use of the generated tasks is not limited to in-class application but also supports homework or supplementary practice assignments. The high portability of the output format facilitates integrating the exercises into an established e-learning platform with defined workflows or otherwise organised structures, thus allowing for more independent and individually orchestrated learning. As the tool is freely accessible, students may also generate exercises themselves for individual study. However, since such a scenario bypasses the step of revision by a proficient speaker of the targeted language, learners need to be aware of potential errors some exercises may contain.

As FLAIR$_{\text{ExGen}}$ covers most of the pedagogical targets included in the 7th and 8th grade curriculum of Baden-Württemberg high schools, it is well suited to be employed in English classes of public classroom teaching throughout the school year. Rare constructions such as conditional clauses, which represent a challenge to common exercise generation systems operating on authentic texts (Phoocharoensil, 2014), are well supported by our tool thanks to the combination with FLAIR's document ranking which identifies suitable texts. Yet although systems like FLAIR increase the likelihood of finding documents containing the targeted constructions, they cannot provide any guarantees. A certain flexibility is still required with respect to the topic of the web search and to the sites from which documents are considered. In order to extend coverage of the curriculum even further, question tags could easily be supported in the future as FLAIR already annotates them. For the remaining topics of the curriculum which are not or not fully included in FLAIR$_{\text{ExGen}}$, additional NLP analyses will need to be implemented in order to identify the relevant constructions. Since the generated exercises are intended as supplementary practice material, however, it is not imperative that the curriculum be covered entirely. The various exercise types allow for varied exercises practicing both comprehension and production of written language. As no auditory components are included, neither listening comprehension nor speech production may be practiced. However, the flexible H5P exercises allow to include such elements so that it is possible to extend the tool to support according exercises.

FLAIR$_{\text{ExGen}}$ performs as good as or better than related work on these quality criteria, especially considering that none of the existing exercise generation tools perform well on all of them. Only the range of supported languages is limited to a single language so far, but since learners of English constitute by far the largest group of language learners, we address a major target group nonetheless.

4.3.2 Scope of applicable documents

Although not all web pages are equally suitable due to rendering issues in some cases, most of the assessed sites preserve visual context to a high degree. The encountered issues related to JavaScript functions which were not executed on the target platform were all coupled with lazy loading. Although libraries exist which execute JavaScript code before downloading the resulting HTML structure, they only target functions that are executed on loading the page. As lazy loading is triggered by a user action, integrating such libraries would therefore not resolve the problem. Another approach to the issue consists in including the JavaScript code of the original web sites in the generated exercises. However, this would introduce considerable liability as the scripts often contain dependencies to libraries which may not be accessible from the target platform or are incompatible with the target platform's own JavaScript implementations. The issues arising from absolute positioning in CSS styling, on the other hand, could be resolved by embedding the exercise document into an IFrame. However, implementing this functionality requires profound adjustments to the H5P content types, and therefore needs to be deferred to future work. In the meantime, these issues can be resolved by making use of the options provided in the configuration interface to cut out parts of the documents.

4.3.3 Performance for generating content

In order to determine whether the generated exercises constitute correct practice material, we evaluated exemplary exercises that we generated for assessment purposes. The assessments of sampled documents are limited in sample size as they require manual generation of exercises and also manual analysis thereof whenever no gold standard annotations or objective measures are available. The results therefore provide a good indication for the system's performance on documents from sites that we identified as suitable for academic teaching. They may, however, not be representative for other online sources. The insights gained from the evaluations provide valuable pointers to possible issues and allow to determine potentials for improvement.

The errors in identifying target constructions can be divided into two sub-categories: (1) errors inherent to FLAIR's construction extraction and (2) errors stemming from determining more fine-grained constructions from the extracted constructions. In the first category, especially the identification of progressive and passive forms still poses a challenge. Ensuring that all such constructions contain a form of the lemma *be* could already improve their recognition precision. The implementation for verb constructions relies on a dependency parse. A straightforward approach to address errors resulting from alleged components belonging to distinct clauses therefore consists in only considering the auxiliary dependants of the main verb as components of the construction. Figure 4.3.1 illustrates that the incorrectly classified sentence from Example 12c could then be labelled correctly. However, this requires correct analyses by the parser, a problem already recognized by Chinkina et al. (2016) in an evaluation of FLAIR's performance where the Stanford CoreNLP parser did not reliably identify present participle forms.

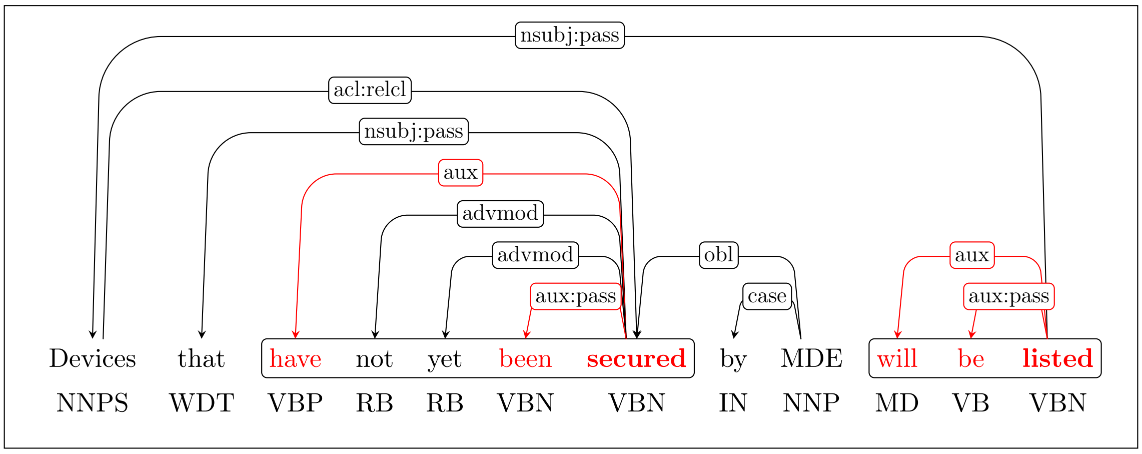


Figure 4.3.1: Dependency graph for compound tense example

The sentence constitutes a simplified adaptation of Example 12c. The dependency graph is based on the output generated in the web tool CoreNLP. The main verbs of compound tense forms are highlighted in bold, their auxiliary dependants are marked in red. The resulting constructions are encircled by rounded rectangles.

Since conditional sentences where one clause indicates a real conditional and the other clause an unreal conditional might not be suitable practice material for learners at beginner's levels, they might need to be excluded from exercise generation altogether. This would also increase the target generation ratio for Conditional DD exercises where the rather flexible use of tenses and modalities in authentic language, as compared to that taught in language classes, complicates determining the type of conditional sentence and the verbs of the clauses. It would also be possible to not extract the problematic conditional constructions at all, thus not using them for ranking purposes either.

Although not technically incorrect, labelling occurrences of *more* and *most* as analytic comparison forms might not be the desired behaviour for the prototypical instructor when these adverbs are used within a synthetic comparative or superlative form. By only assigning a tag for *analytic form* if the occurrence is not part of a synthetic construction, the issue could be resolved with minimal programming effort. In addition, the ranking would then favour documents with more diverse occurrences of these constructions. This would at the same time address the issue of low variability for Comparison tasks.

In order to correctly determine irregular tenses, considerably more effort will be required. FLAIR currently only takes into account simple past and past participle forms for the *irregular* construction and does not make a distinction between these two forms. Present participles are not considered at all. For exercises targeting compound tense forms, we need to determine irregularity based on the main verb as this part differs from one instance of the construction to another. Our current implementation, however, considers a tense construction to be irregular if it contains any irregular verb. Irregular auxiliaries may therefore corrupt the classification. In order to consider only the main verb when checking for irregularity, a distinction between present participles, past participles, and conjugated verb forms is necessary. FLAIR's current implementation for irregular constructions therefore needs to be replaced by the three constructions *irregular past participles*, *irregular present participles* and *irregular simple past* in order to support a more reliable identification of exercise targets.

Another construction worthwhile introducing constitutes main clauses. If only those verbs contained in the main clause of a question were classified as interrogative, this would eliminate the misclassification of verbs in sub-clauses of questions. Considering the rather low frequency of negated and interrogative forms, it is questionable whether any user would like to specifically target such constructions. It might therefore make more sense to instead offer the configuration option to exclude target constructions in interrogative or negated forms from exercise generation in order to reduce task complexity. Including only negated and/or interrogative forms but not affirmative statements would not be possible. Incorrectly labelling occurrences as interrogative or negated would then only reduce recall of construction identification which, as argued above, is of negligible importance to exercise generation.

Since all sampled occurrences of relative pronouns other than *who*, *which* and *that* were incorrect, this configuration option might have to be removed. The pronouns *whose* and *whom*, which are explicitly listed in the curriculum, could be introduced instead, either combined into a single configuration parameter or as individual options. More rarely used relative pronouns such as *where*, *when*, *why*, *how*, and *what* would then not be targeted by the exercises. An alternative approach consists in keeping the current configuration options for other relative pronouns and adding a string-based filter for all above mentioned pronouns, which have been targeted as important representatives of relative pronouns by previous research (Khullar et al., 2018; Ball, 1994). Thus, only occurrences which are both identified as relative pronouns by the parser and in addition correspond to one of the defined token values would be considered.

The target generation ratio is quite promising, yet a couple of optimizations could improve the prediction of the number of exercise targets with little effort. A number of targets is rejected as they overlap with another annotation. This is especially the case with tenses where auxiliaries are annotated as part of the compound tense verb and in addition as individual verbs. Comparison forms where the synthetic form always contains an analytic form, usually either *more* or *most* are also affected by this issue. The problem therefore resides in an incorrect estimate of the number of exercise targets during exercise configuration. Identifying overlaps before calculating the number of possible targets would allow for a more accurate estimate of the actual number of generated targets. The code adjustments to this purpose could either be integrated into FLAIR's construction extraction on the server, or else into the client's presenter in order to keep the logic for document ranking unchanged.

4.3.4 Technical performance

Execution times are currently mostly determined by feedback generation. We hypothesize that the challenge of the exercise topics with the longest feedback generation times, Passive and Tenses, lies in the length of the target items, which often span multiple tokens. These performance issues are being addressed as the microservice is being developed into a more stable release. When they are overcome, the feedback generation mechanism is quite robust, indicating that it is well applicable to exercises generated from authentic texts.

4.3.5 Scope of the evaluation

We have covered a range of important quality criteria in the evaluation. Future studies should focus on assessing additional features relevant to exercise generation systems. Table 4.3.2 gives an overview of valuable evaluation criteria with indications about which ones were covered in this thesis. Criteria which were not addressed in our evaluation are described in the following.

Criterion		Covered
Versatility	Portability	●
	Configurability	●
Coverage of learning scenarios	Pedagogical targets	●
	Languages	●
	Language use & modalities	●
	Skill acquisition perspective	●
	Learner model	●
Robustness and Correctness	Grammaticality	●
	Documents suitability	●
	Target generation ratio	●
	Variability of exercises	●
	Appropriateness for the pedagogical goal	
	Appropriateness for the language level	
	Unambigousness	
	Robustness and correctness of feedback	(●)
Usability	User friendliness	
	Execution time	●
	Time savings over manual generation	

Table 4.3.2: Evaluation criteria

The evaluation criteria that we covered in our assessment are marked with a dot. Dots in parentheses indicate criteria of which only some aspects were evaluated.

Evaluating the generated exercises in terms of their **appropriateness for the pedagogical goal** (Perez-Beltrachini et al., 2012; Lee et al., 2016) should be based on judgement by subject matter experts who are not otherwise involved in the project in order to be objective.

In order to judge whether the exercises are **appropriate to the language level** of the target group (Perez-Beltrachini et al., 2012; Hoshino and Nakagawa, 2008; Lee et al., 2016), possible approaches include expert judgement and computational analyses of exercise complexity. Both exercise targets and the text context need to bee considered.

Perez-Beltrachini et al. (2012) introduce **unambigousness**, which is best evaluated in large-scale user studies. This criterion requires exercises to give sufficient information in the task description, brackets of FiB exercises or the text context for the learner to determine the correct answer.

Our evaluation of feedback only focused on robustness without considering **correctness of the generated feedback**. Assessing the FeedBook microservice's performance on authentic texts also requires user studies.

In order to evaluate the system's **user friendliness**, the configuration interface, manageability of the exercises on the target platform, and the entire workflow need to be considered (Naiara Perez Miguel, 2017; Antoniadis et al., 2004; Chen et al., 2006; Volodina et al., 2014). FLAIR$_{\text{ExGen}}$ has been designed to be used by instructors with no technical knowledge other than using a web browser. Users with no prior experience with the FLAIR system are granted easy access by also enabling exercise generation without applying construction weighting. The tool assists them in generating valid exercise configurations by automatically disabling settings that cannot result in successful exercise generation. In order to maximize usability of the H5P exercises on the target platform, the custom content types were kept as close to the original H5P types as possible so that they require minimal re-familiarization. Markup elements are not displayed to users, allowing instructors to operate solely on plain texts if they use the H5P authoring interface for post-processing. The hybrid approach to specify exercise targets inline with references to separately defined incorrect answers aims at high maintainability of these elements through direct integration of the targets into the text context while at the same time not cluttering the plain text with large amounts of meta-information on targets. A representative assessment of the current implementation requires a large-scale usability study.

For a pedagogical evaluation of **time savings** when compared to manual compilation of exercises, an experimental setup is required in order to determine the effort necessary for manual exercise generation as well as that for configuration and post-processing of automatically generated exercises.

Chapter 5

Conclusion

We have presented a tool for automatic generation of English form-based grammar exercises from authentic web texts. It uses a language aware search engine to address the challenge of identifying documents rich in target constructions. By putting a focus on preserving the visual context of the original exercise text source, high configurability, integrating feedback and providing the exercises in a portable format, we combine the strengths of previous exercise generation approaches into a single application. An evaluation of the current implementation yielded promising results concerning the range of use cases for the tool and its robustness in generating usable exercises that comply with the instructor's configuration. Current issues regarding processing times have been discussed along with approaches to resolve them. Potentials to extend the tool have been identified in the support of additional pedagogical targets as well as other languages, starting with German which is already supported by the base system FLAIR and the feedback microservice.

Future work should in a first step focus on addressing remaining issues of the current implementation. The logic to identify target constructions needs to be revised and extended in order to improve performance for already included constructions and extend support to additional ones relevant to the curriculum. Improvements concerning automatically generated feedback are closely linked to the continually developed microservice. As work on the feedback microservice progresses, additional exercise types can also offer support for feedback and the implementation might need to be adjusted to meet changing requirements of the service's interface. Especially integrating online feedback into the JavaScript implementations of the H5P content types is expected to enhance the user experience considerably. Updates to the H5P base content types will need to be integrated into the derived types on a regular basis. It might further be worth considering to replace the currently used content types with custom types developed from scratch. This would allow to keep the content types more uniform and to include only those features that are actually applicable and relevant to our automatically generated, context-embedded exercises. Moreover, such an endeavor would allow us to embed the exercise content into an IFrame without the need to consider implications on the current implementation as manipulating DOM elements inside an IFrame requires different access mechanisms. This is especially relevant considering that the use of IFrames constitutes the only clean solution to the rendering issues that occur in the case of absolutely positioned HTML elements. Eliminating these problems will enable a larger range of web documents to be displayed properly. All changes will need to be evaluated in more large-scale assessments, also including a usability study.

Since the generated exercises support building a learner model, bringing our tool to the next level requires using that data to then automatically generate exercise configurations adapted to the user's profile. By thus completing the loop to encompass all steps from exercise configuration over generation to tracking of learner data, the entire workflow can be automated. This would relieve instructors from the monotonous and time consuming task of standard exercise generation and enable them to instead take on the role of a guide and supervisor which is being postulated in current literature (e.g. Peredrienko et al., 2020).

Bibliography

Mohammad Javad Ahmadian. 2012. Task repetition in ELT. *ELT Journal*, 66(3):380–382.

Itziar Aldabe, Maddalen Lopez de Lacalle, Montse Maritxalar, Edurne Martinez, and Larraitz Uria. 2006. Arikiturri: An automatic question generator based on corpora and nlp techniques. volume 4053, pages 584–594.

J. João Almeida, Eliana Grande, and Georgi Smirnov. 2017. Exercise generation on language specification. In *Recent Advances in Information Systems and Technologies*, pages 277–286, Cham. Springer International Publishing.

Luiz Alexandre Mattos Do Amaral. 2007. *Designing Intelligent Language Tutoring Systems for Integration into Foreign Language Instruction.* Ph.D. thesis, USA. AAI3262074.

Georges Antoniadis, Sandra Echinard, Olivier Kraif, Thomas Lebarbé, Mathieu Loiseau, and Claude Ponton. 2004. Nlp-based scripting for call activities. In *Proceedings of the Workshop on eLearning for Computational Linguistics and Computational Linguistics for eLearning*, pages 18–25.

Georges Antoniadis and Claude Ponton. 2004. Mirto : un système au service de l'enseignement des langues.

Mahmoud Azab, Ahmed Salama, Kemal Oflazer, Hideki Shima, Jun Araki, and Teruko Mitamura. 2013. An English reading tool as a NLP showcase. In *The Companion Volume of the Proceedings of IJCNLP 2013: System Demonstrations*, pages 5–8, Nagoya, Japan. Asian Federation of Natural Language Processing.

Catherine N. Ball. 1994. Relative pronouns in it-clefts: The last seven centuries. *Language Variation and Change*, 6(2):179–200.

Lisa Marina Beinborn. 2016. *Predicting and Manipulating the Difficulty of Text-Completion Exercises for Language Learning*. Ph.D. thesis, Technische Universität Darmstadt, Darmstadt.

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

Ali M. Bianco, Maria De Marsico, and Marco Temperini. 2004. Standards for e-learning. http://www2.tisip.no/quis/public_files/wp5-standards-for-elearning.pdf. [Online; accessed 30-July-2021].

Eckhard Bick. 2000. Live use of corpus data and corpus annotation tools in call: Some new developments in visl.

Stephen Bodnar and Roy Lyster. 2021. Choose Your Own Content: a technology-based approach for automatically creating tailored grammar practice exercises from the web. 9th International Conference on Second Language Pedagogies.

Oliver Bohl, Jörg Scheuhase, Ruth Sengler, and Udo Winand. 2002. The sharable content object reference model (scorm)-a critical review. In *International Conference on Computers in Education, 2002. Proceedings.*, pages 950–951. IEEE.

Elisabeth Breidt and Helmut Feldweg. 1997. Accessing foreign languages with compass. *Machine Translation*, 12(1/2):153–174.

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

Jonathan Brown, Gwen Frishkoff, and Maxine Eskenazi. 2005. Automatic question generation for vocabulary assessment. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 819–826, Vancouver, British Columbia, Canada. Association for Computational Linguistics.

Capterra. 2021. eLearning Authoring Tools. https://www.capterra.com/elearning-authoring-tools-software/. [Online; accessed 30-July-2021].

Chia-Yin Chen, Hsien-Chin Liou, and Jason S. Chang. 2006. FAST – an automatic generation system for grammar tests. In *Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions*, pages 1–4, Sydney, Australia. Association for Computational Linguistics.

Chih-Ming Chen and Ching-Ju Chung. 2008. Personalized mobile english vocabulary learning system based on item response theory and learning memory cycle. *Computers & Education*, 51(2):624–645.

Chih-Ming Chen and Shih-Hsun Hsu. 2008. Personalized intelligent mobile learning system for supporting effective english learning. *Journal of Educational Technology & Society*, 11(3):153–180.

Tao Chen, Naijia Zheng, Yue Zhao, Muthu Kumar Chandrasekaran, and Min-Yen Kan. 2015. Interactive second language learning from news websites. In *Proceedings of the 2nd Workshop on Natural Language Processing Techniques for Educational Applications*, pages 34–42, Beijing, China. Association for Computational Linguistics.

Maria Chinkina, Madeeswaran Kannan, and Detmar Meurers. 2016. Online information retrieval for language learning. In *Proceedings of ACL-2016 System Demonstrations*, pages 7–12, Berlin, Germany. Association for Computational Linguistics.

Dorothy Chun. 2016. The role of technology in sla research. *Language, Learning and Technology*, 20:98–115.

Prof. Dr. Jozef Colpaert and Emre Sevinc. 2010. ClozeFox: Gap Exercise Generator with Scalable Intelligence for Mozilla Firefox. https://github.com/emres/clozefox. [Online; accessed 23-June-2021].

Ricardo Conejo, Eduardo Guzmán, Jose-Luis Perez de-la Cruz, and Beatriz Barros. 2014. An empirical study on the quantitative notion of task difficulty. *Expert Systems with Applications*, 41(2):594–606.

David Coniam. 1997. A preliminary inquiry into using corpus word frequency data in the automatic generation of english language cloze tests. *CALICO Journal*, 14.

Berthold Crysmann, Nuria Bertomeu, Peter Adolphs, Daniel Flickinger, and Tina Klüwer. 2008. Hybrid processing for grammar and style checking. In *Proceedings of the 22nd International Conference on Computational Linguistics. International Conference on Computational Linguistics (COLING-2008), 22nd, August 18-22, Manchester, United Kingdom*, pages 153–160. Coling 2008 Organizing Committee.

Suyada Dansuwan, Kikuko Nishina, Kanji Akahori, and Yasutaka Shimizu. 2013. Development and evaluation of a thai learning system on the web using natural language processing. *CALICO Journal*, 19.

Rodolfo Delmonte. 2003. Linguistic knowledge and reasoning for error diagnosis and feedback generation. *CALICO Journal*, 20.

Markus Dickinson and Joshua Herring. 2008. Developing online icall exercises for russian. In *Proceedings of the Third Workshop on Innovative Use of NLP for Building Educational Applications*, EANL '08, page 1–9, USA. Association for Computational Linguistics.

Duco Dokter, John Nerbonne, Lily Schurcks-Grozeva, and Petra Smit. 1997. Glosser-rug; a user study.

eLearning Industry. 2021. Tin Can API Compliant eLearning Authoring Tools. https://elearningindustry.com/directory/software-categories/elearning-authoring-tools/compliance/tin-can?gclid=CjwKCAjw47eFBhA9EiwAy8kzNDaQReSu5U4MkHpv42_b6RU7DnJkqSYty37ani5P-JSHVgcSiSj3BRoC7ygQAvD_BwE. [Online; accessed 30-July-2021].

FGFH EWR and Stephanie Seneff. 2007. Automatic generation of cloze items for prepositions. pages 2173–2176.

Anne Vandeventer Faltin. 2003. Syntactic error diagnosis in the context of computer assisted language learning.

Kledilson Ferreira and Álvaro R. Pereira Jr. 2018. Verb tense classification and automatic exercise generation. WebMedia '18, page 105–108, New York, NY, USA. Association for Computing Machinery.

Ken Feuerman, Catherine Marshall, David Newman, and Marikka Rypa. 1987. The calle project. *CALICO Journal*, 4(3):25–34.

Bridgid Finn and Janet Metcalfe. 2010. Scaffolding feedback to maximize long-term error correction. *Memory & cognition*, 38:951–61.

Carmen Garcia. 2008. Using authentic reading texts to discover underlying socio-cultural information. *Foreign Language Annals*, 24:515 – 526.

Global Web Index. 2019. News consumption.

Said Hadjerrouit. 2010. Developing web-based learning resources in school education: A user-centered approach. *Interdisciplinary Journal of e-Skills and Lifelong Learning*, 6:115–135.

Trude Heift. 2003. Multiple learner errors and meaningful feedback: A challenge for icall systems. *CALICO Journal*, 20:533–548.

Trude Heift. 2015. Web delivery of adaptive and interactive language tutoring: Revisited. *International Journal of Artificial Intelligence in Education*, 26:489–503.

Michael Heilman, Kevyn Collins-Thompson, Jamie Callan, Maxine Eskenazi, Alan Juffs, and Lois Wilson. 2010. Personalization of reading passages improves vocabulary acquisition. *I. J. Artificial Intelligence in Education*, 20:73–98.

Johannes Heinecke, Jurgen Kunze, Wolfgang Menzel, and Ingo Schroder. 1998. Eliminative parsing with graded constraints. In *COLING 1998 Volume 1: The 17th International Conference on Computational Linguistics*.

Ayako Hoshino and Hiroshi Nakagawa. 2005a. A real-time multiple-choice question generation for language testing: A preliminary study. In *Proceedings of the Second Workshop on Building Educational Applications Using NLP*, pages 17–20, Ann Arbor, Michigan. Association for Computational Linguistics.

Ayako Hoshino and Hiroshi Nakagawa. 2005b. Webexperimenter for multiple-choice question generation. In *Proceedings of HLT/EMNLP on Interactive Demonstrations*, HLT-Demo '05, page 18–19, USA. Association for Computational Linguistics.

Ayako Hoshino and Hiroshi Nakagawa. 2008. A cloze test authoring system and its automation. In *Advances in Web Based Learning – ICWL 2007*, pages 252–263, Berlin, Heidelberg. Springer Berlin Heidelberg.

Ayako Hoshino and Hiroshi Nakagawa. 2010. Predicting the difficulty of multiple-choice close questions for computer-adaptive testing. *Natural Language Processing and its Applications*, page 279.

Yuko Hoshino. 2013. Relationship between types of distractor and difficulty of multiple-choice vocabulary tests in sentential context. *Language Testing in Asia*, 3:16.

ICEF. 2019. The world's changing language landscape. https://monitor.icef.com/2019/10/the-worlds-changing-language-landscape/. [Online; accessed 24-August-2021].

Shu Jiang and John Lee. 2017. Distractor generation for Chinese fill-in-the-blank items. In *Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 143–148, Copenhagen, Denmark. Association for Computational Linguistics.

Joubel. 2016. H5P - About the project. https://h5p.org/about-the-project. [Online; accessed 30-July-2021].

Joubel. 2018. H5P and xAPI. https://h5p.org/documentation/x-api. [Online; accessed 30-July-2021].

Joubel. 2021. H5p - integrations. https://h5p.org/integrations#integrations. [Online; accessed 11-June-2021].

Joubel. n.d.a. Allowed File Extensions. https://h5p.org/allowed-file-extensions. [Online; accessed 30-July-2021].

Joubel. n.d.b. H5P as a Service. https://h5p.com/. [Online; accessed 30-July-2021].

Julien Monty. 2015. Moodle original logo symbol. https://icon-icons.com/de/symbol/moodle-original-logo/146420. [Online; accessed 3-June-2021].

Nikiforos Karamanis, Le An Ha, and Ruslan Mitkov. 2006. Generating multiple-choice test items from medical text: A pilot study. In *Proceedings of the Fourth International Natural Language Generation Conference*, pages 111–113, Sydney, Australia. Association for Computational Linguistics.

Payal Khullar, Konigari Rachna, Mukul Hase, and Manish Shrivastava. 2018. Automatic question generation using relative pronouns and adverbs. In *Proceedings of ACL 2018, Student Research Workshop*, pages 153–158.

Charles Knight and Sam Pryke. 2012. Wikipedia and the University, a case study. *Teaching in Higher Education*, 17(6):649–659.

Susanne Knoop and Sabrina Wilske. 2013. Wordgap - automatic generation of gap-filling vocabulary exercises for mobile learning.

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.

John Lee, Donald Sturgeon, and Mengqi Luo. 2016. A call system for learning preposition usage. pages 984–993.

Michael Levison, Greg Lessard, and D. Walker. 2000. A multi-level approach to the detection of second language learners errors. *Literary and Linguistic Computing*, 15(3):313–322.

Sébastien L'haire and Anne Vandeventer Faltin. 2003. Error diagnosis in the freetext project. *CALICO Journal*, 20(3):481–495.

Chen Liang, Xiao Yang, Drew Wham, Bart Pursel, Rebecca Passonneaur, and C. Lee Giles. 2017. Distractor generation with generative adversarial nets for automatically creating fill-in-the-blank questions. In *Proceedings of the Knowledge Capture Conference*, K-CAP 2017, New York, NY, USA. Association for Computing Machinery.

Chao-Lin Liu, Chun-Hung Wang, Zhao-Ming Gao, and Shang-Ming Huang. 2005. Applications of lexical information for algorithmically composing multiple-choice cloze items. In *Proceedings of the Second Workshop on Building Educational Applications Using NLP*, pages 1–8, Ann Arbor, Michigan. Association for Computational Linguistics.

Peng Liu and Zhizhong Li. 2012. Task complexity: A review and conceptualization framework. *International Journal of Industrial Ergonomics*, 42(6):553–568.

Sandra Lourenco. 2015. Syntactic reap . pt exercises on passive transformation.

Alison Mackey. 2006. Feedback, Noticing and Instructed Second Language Learning. *Applied Linguistics*, 27(3):405–430.

Aisha Majid. 2021. Top 50 largest news websites in the world: Surge in traffic to Epoch Times and other right-wing sites. *PressGazette.* [Online; accessed 11-July-2021].

R M Sumudu Medawela, D.R.D.L Ratnayake, Wijeyapala Abeyasinghe, Ruwan Jayasinghe, and Kosala Marambe. 2017. Effectiveness of “fill in the blanks” over multiple choice questions in assessing final year dental undergraduates. *Educación Médica*, 19.

MAITE MELERO and ARIADNA FONT. 2001. Construction of a Spanish Generation Module in the framework of a general-purpose, Multilingual Natural Language Processing System. *VII International Symposium on Social Communication.*

Detmar Meurers. 2012. *Natural Language Processing and Language Learning*. American Cancer Society.

Detmar Meurers, Kordula De Kuthy, Florian Nuxoll, Björn Rudzewitz, and Ramon Ziai. 2019. Scaling up intervention studies to investigate real-life foreign language learning in school. *Annual Review of Applied Linguistics*, 39:161–188.

Detmar Meurers, Ramon Ziai, Luiz Amaral, Adriane Boyd, Aleksandar Dimitrov, Vanessa Metcalf, and Niels Ott. 2010. Enhancing authentic web pages for language learners. pages 10–18.

Eleni Miltsakaki and Audrey Troutt. 2011. Read-x: Automatic evaluation of reading difficulty of web text.

Jugend und Sport Ministerium für Kultus. 2016. Englisch als erste Fremdsprache.

Mozilla. 2005. MIME types (IANA media types). https://developer.mozilla.org/en-US/docs/Web/HTTP/Basics_of_HTTP/MIME_types. [Online; accessed 24-June-2021].

Eugene W Myers. 2005. Ano(nd) difference algorithm and its variations. *Algorithmica*, 1:251–266.

Noriko Nagata. 1995. An effective application of natural language processing in second language instruction. *the CALICO Journal*, 13:47–67.

Noriko Nagata. 2009. Robo-sensei’s nlp-based error detection and feedback generation. *CALICO Journal*, 26(3):562–579.

Noriko Nagata. 2011. Rules vs. examples: An experimental study using the banzai parser.

Montserrat Cuadros Oller Naiara Perez Miguel. 2017. Multilingual call framework for automatic language exercise generation from free text. pages 49–52.

Rick Noack and Lazaro Gamio. 2015. The world’s languages, in 7 maps and charts. *The Washington Post*, 4(23):65–70.

Yusuf Sulistyo Nugroho, Hideaki Hata, and Kenichi Matsumoto. 2019. How different are different diff algorithms in git? *Empirical Software Engineering*, 25(1):790–823.

David Nunan. 1989. *Designing tasks for the communicative classroom.* Cambridge university press.

Lourdes Ortega. 2014. *Understanding Second Language Acquisition*. Understanding Language. Taylor & Francis.

Niels Ott and Detmar Meurers. 2011. Information retrieval for education: Making search engines language aware. *Themes in Science and Technology Education*, 3(1-2):9–30.

Irina Pandarova, Torben Schmidt, Johannes Hartig, Ahcene Boubekki, Roger Jones, and Ulf Brefeld. 2019. Predicting the difficulty of exercise items for dynamic difficulty adaptation in adaptive language tutoring. *International Journal of Artificial Intelligence in Education*.

Petros Papazoglou Papazoglakis. 2013. The past, present and future of scorm. *Academy of Economic Studies. Economy Informatics*, 13(1):16.

Dana M. Paramskas. 2013. Courseware-software interfaces: Some designs and some problems. *Calico Journal*, 1(3):4–6.

M. Parrott. 2010. *Grammar for English Language Teachers: With Exercises and a Key*. Cambridge University Press.

Matthew Peacock. 1997. The effect of authentic materials on the motivation of EFL learners. *ELT Journal*, 51(2):144–156.

Radek Pelánek, Tomáš Effenberger, and Jaroslav Čechák. 2021. Complexity and difficulty of items in learning systems. *International Journal of Artificial Intelligence in Education*, pages 1–37.

Tatiana Peredrienko, Oxana Belkina, and Elena Yaroslavova. 2020. New language learning environment: Employers'-learners' expectations and the role of teacher 4.0. *International Journal of Instruction*, 13(3):105–118.

Laura Perez-Beltrachini, Claire Gardent, and German Kruszewski. 2012. Generating grammar exercises. In *Proceedings of the Seventh Workshop on Building Educational Applications Using NLP*, pages 147–156, Montréal, Canada. Association for Computational Linguistics.

Supakorn Phoocharoensil. 2014. If-conditionals in authentic corpus-based english. *Review of European Studies*, 6.

Sebastian Rettig. 2019. scorm-h5p-wrapper. https://github.com/sr258/scorm-h5p-wrapper. [Online; accessed 30-July-2021].

Veit Reuer. 2003. Error recognition and feedback with lexical functional grammar. *CALICO Journal*, 20(3):497–512.

Robert Joshua Reynolds, Eduard Schaf, and Walt Detmar Meurers. 2014. A view of russian: Visual input enhancement and adaptive feedback.

Richard Willy. 2013. elasticsearch-opennlp-auto-tagging. https://raw.githubusercontent.com/richardwilly98/elasticsearch-opennlp-auto-tagging/master/src/main/resources/models/en-lemmatizer.dict. [Online; accessed 17-July-2021].

Peter Robinson. 2001. *Cognition and Second Language Instruction*. Cambridge Applied Linguistics. Cambridge University Press.

Peter Robinson, Sarah Ting, and Jian Urwin. 1995. Investigating second language task complexity. *RELC Journal*, 26:62–79.

Cameron Romney. 2016. Considerations for Using Images in Teacher Made Materials. *The2016PanSIGJournal.*

Björn Rudzewitz, Ramon Ziai, Kordula De Kuthy, and Detmar Meurers. 2017. Developing a web-based workbook for English supporting the interaction of students and teachers. In *Proceedings of the joint workshop on NLP for Computer Assisted Language Learning and NLP for Language Acquisition*, pages 36–46, Gothenburg, Sweden. LiU Electronic Press.

Björn Rudzewitz, Ramon Ziai, Kordula De Kuthy, Verena Möller, Florian Nuxoll, and Detmar Meurers. 2018. Generating feedback for English foreign language exercises. In *Proceedings of the Thirteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 127–136, New Orleans, Louisiana. Association for Computational Linguistics.

Keisuke Sakaguchi, Yuki Arase, and Mamoru Komachi. 2013. Discriminative approach to fill-in-the-blank quiz generation for language learners. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 238–242, Sofia, Bulgaria. Association for Computational Linguistics.

Jacobijn Sandberg, Marinus Maris, and Pepijn Hoogendoorn. 2014. The added value of a gaming context and intelligent adaptation for a mobile learning application for vocabulary learning. *Computers & Education*, 76:119–130.

David A. Schneider and Kathleen F. McCoy. 1998. Recognizing syntactic errors in the writing of second language learners. *arXiv preprint cmp-lg/9805012.*

Ulf Schuetze. 2018. Practicing grammar online: Multiple-choice or fill-in-the-blanks. *Electronic Journal of Foreign Language Teaching*, 15(1):55–65.

Lee Schwartz, Takako Aikawa, and Michel Pahud. 2004. Dynamic language learning tools. *Proceedings of InSTIL/ICALL Symposium 2004.*

Kathleen Sheehan, Irene Kostin, and Yoko Futagi. 2007. Sourcefinder: A construct-driven approach for locating appropriately targeted reading comprehension source texts.

Simon Smith, Avinesh P.V.S, and Adam Kilgarriff. 2010. Gap-fill tests for language learners: Corpus-driven item generation.

Eiichiro Sumita, Fumiaki Sugaya, and Seiichi Yamamoto. 2005. Measuring non-native speakers' proficiency of English by using a test with automatically-generated fill-in-the-blank questions. In *Proceedings of the Second Workshop on Building Educational Applications Using NLP*, pages 61–68, Ann Arbor, Michigan. Association for Computational Linguistics.

Yuni Susanti, Takenobu Tokunaga, Hitoshi Nishikawa, and Hiroyuki Obari. 2018. Automatic distractor generation for multiple-choice english vocabulary questions. *Research and Practice in Technology Enhanced Learning*, 13.

The Alpheios Project. 2018. The Alpheios Reading Environment. http://alpheios.net/pages/tools/. [Online; accessed 23-June-2021].

The MITRE Corporation. 2020. CAPEC-209: XSS Using MIME Type Mismatch (Version 3.4). https://capec.mitre.org/data/definitions/209.html. [Online; accessed 24-June-2021].

Janine Toole and Trude Heift. 2001. Generating learning content for an intelligent language tutoring system. In *Proceedings of NLP-CALL Workshop at the 10th Int. Conf. on Artificial Intelligence in Education (AI-ED). San Antonio, Texas*, pages 1–8.

Stephen Victor and Art Werkenthin. 2016. Cracking the mobile learning code: xapi and cmi5.

Elena Volodina, Ildikó Pilán, Lars Borin, and Therese Lindström Tiedemann. 2014. A flexible language learning platform based on language resources and web services. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 3973–3978, Reykjavik, Iceland. European Language Resources Association (ELRA).

Joel Walz. 1989. Context and contextualized language practice in foreign language teaching. *The Modern Language Journal*, 73(2):160–168.

Ralph M. Weischedel and John E. Black. 1980. Responding intelligently to unparsable inputs. *Comput. Linguist.*, 6(2):97–109.

Brad Wilcox, Timothy G. Morrison, and Dallin D. Oaks. 1999. Computer corpora and authentic texts: Toward more effective language teaching. *Reading Research and Instruction*, 38:415–423.

Dave Willis and Jane Willis. 2001. *Task-based language learning*, The Cambridge Guides, page 173–179. Cambridge University Press.

Shaoqun Wu, Margaret Franken, and Ian H. Witten. 2009. Refining the use of the web (and web search) as a language teaching and learning resource. *Computer Assisted Language Learning*, 22(3):249–268.

Ting-Ting Wu and Tien-Wen Sung. 2017. Dynamic e-book guidance system for english reading with learning portfolio analysis. *Electronic Library*, 35:358–373.

Rosely Perez Xavier. 2007. Language tasks and exercises: How do teachers perceive them. In *International Conference on Task-Based Language Teaching*.

Jie Chi Yang and Kanji Akahori. 1998. Error analysis in japanese writing and its implementation in a computer assisted language learning system on the world wide web. *CALICO Journal*, 15(1/3):47–66.

Chak Yan Yeung, John Lee, and Benjamin Tsou. 2019. Difficulty-aware distractor generation for gap-fill items. In *Proceedings of the 17th Workshop of the Australasian Language Technology Association*, pages 167–172. Australasian Language Technology Association. The 17th Workshop of the Australasian Language Technology Association ; Conference date: 04-12-2019 Through 06-12-2019.

Ramon Ziai, Bjoern Rudzewitz, Kordula De Kuthy, Florian Nuxoll, and Detmar Meurers. 2018. Feedback strategies for form and meaning in a real-life language tutoring system. In *Proceedings of the 7th workshop on NLP for Computer Assisted Language Learning*, pages 91–98, Stockholm, Sweden. LiU Electronic Press.

Leonardo Zilio, Rodrigo Wilkens, and Cédrick Fairon. 2017. Using NLP for enhancing second language acquisition. In *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017*, pages 839–846, Varna, Bulgaria. INCOMA Ltd.

Evaluation Source Documents

Ghaith Abdul-Ahad. 2021. ‘They will never let go’: Isis fighters regroup in the heart of Iraq. *The Guardian*. [Online; accessed 11-July-2021].

BBC. 2020. Euro 2020 postponed until next summer. https://www.bbc.com/sport/football/51909518. [Online; accessed 01-August-2021].

BBC. 2021a. Donald Trump. https://www.bbc.com/news/topics/cp7r8vgl2lgt/donald-trump. [Online; accessed 11-July-2021].

BBC. 2021b. Haiti president’s assassination: What we know so far. *BBC*.

BBC. 2021c. Uefa Euro 2020: Everything you need to know about the summer’s tournament. https://www.bbc.com/sport/football/54917417. [Online; accessed 29-July-2021].

Joanna Bell and Caroline Graham. 2021. ’I’ve been a mug’: Aristocrat Henry Roper-Curzon says he was ’duped’ into marrying his ’Kurdish Princess’ bride - and was only rescued when his mother turned detective to investigate. *Daily Mail*. [Online; accessed 11-July-2021].

Chris Bevan. 2021. Euro 2020: BBC pundits make their European Championship predictions. https://www.bbc.com/sport/football/57413544. [Online; accessed 01-August-2021].

Sam Blitz. 2021. Can England and Scotland both get out of their group and which one out of France, Germany and Portugal will suffer in the ’Group of Death’? All you need to know about the Euro 2020 group stages as the tournament edges closer... *Daily Mail*. [Online; accessed 12-July-2021].

Caparas, M.J. and Davis, Chris and Hanson, Diana and Simpson, Daniel. 2021. Device discovery overview. https://docs.microsoft.com/en-us/microsoft-365/security/defender-endpoint/device-discovery?view=o365-worldwide. [Online; accessed 13-July-2021].

Matt Craig and Sophie Kasakove. 2021. Death Valley Hits 130 Degrees as Heat Wave Sweeps the West. *The New York Times*. [Online; accessed 11-July-2021].

Michael Crowley, Michael D. Shear, and Eric Schmitt. 2021. Biden Administration Shows Little Appetite for Haiti’s Troop Request. *The New York Times*. [Online; accessed 11-July-2021].

Daily Mail. 2021. Sport. *Daily Mail*. [Online; accessed 12-July-2021].

Dance Facts. 2021. Types of dance - categories. http://www.dancefacts.net/dance-types/types-of-dances/. [Online; accessed 13-July-2021].

Andrew Downie. 2021. Argentina beat Brazil 1-0 to win Copa America, 1st major title in 28 yrs. *Reuters*. [Online; accessed 11-July-2021].

Simon Evans. 2021. Soccer-Tactical breakdown of Italy v England Euro 2020 final. https://www.reuters.com/article/us-soccer-euro-ita-eng-tactics-idCAKCN2EF1L1. [Online; accessed 01-August-2021].

Nicholas Fandos. 2021. 'We have a lot of work to do.' Schumer warns Democrats to expect a busy July. . *The New York Times*. [Online; accessed 13-July-2021].

Doug Faulkner and Becky Morton. 2021. Euro 2020: Queen and Boris Johnson wish England well as final looms. *BBC*. [Online; accessed 11-July-2021].

Trip Gabriel. 2021. The Big Question of the 2022 Midterms: How Will the Suburbs Swing? *The New York Times*. [Online; accessed 11-July-2021].

Matias Grez. 2021. Italy crowned European champion after beating England on penalties. https://edition.cnn.com/2021/07/11/football/england-italy-euro-2020-final-wembley-spt-intl/index.html. [Online; accessed 01-August-2021].

Michael Holden and Mitch Phillips. 2021. England's Black players face racial abuse after Euro 2020 defeat. https://www.reuters.com/world/uk/uk-pm-johnson-condemns-racist-abuse-england-soccer-team-2021-07-12/. [Online; accessed 01-August-2021].

Scarlet Howes. 2021. Matt Hancock's lover Gina Coladangelo breaks cover (without her wedding ring) as her first husband's friend describes her marital woes as 'karma'. *Daily Mail*. [Online; accessed 11-July-2021].

Helena Kelly. 2017. The Many Ways in Which We Are Wrong About Jane Austen. https://lithub.com/the-many-ways-in-which-we-are-wrong-about-jane-austen/. [Online; accessed 11-July-2021].

Danica Kirka and Kathy Gannon. 2021. 'He was our eye': Reuters photographer killed in Afghanistan. https://chicago.suntimes.com/2021/7/16/22580303/reuters-photographer-killed-as-afghan-forces-fight-taliban. [Online; accessed 19-August-2021].

Lange, David. 2021. Uefa euro 2020 - statistics & facts. https://www.statista.com/topics/7971/euro-2020/. [Online; accessed 12-July-2021].

Mark Lowen. 2021. Football-mad Italians gear up for big night. *BBC*. [Online; accessed 11-July-2021].

Alex Marshall. 2018. Is it good to finish a phd fast? https://academia.stackexchange.com/questions/116761/is-it-good-to-finish-a-phd-fast/116819. [Online; accessed 14-July-2021].

Luke McGee. 2021. Five years after the Brexit vote, the United Kingdom is more divided than ever. *CNN*. [Online; accessed 11-July-2021].

Ben Morse. 2021. Racist abuse directed at England players after Euro 2020 final defeat is described as 'unforgivable' by manager Gareth Southgate. https://edition.cnn.com/2021/07/12/football/england-racist-abuse-bukayo-saka-jadon-sancho-marcus-rashford-euro-2020-final-spt-intl/index.html. [Online; accessed 31-July-2021].

Edgar Allan Poe. 2013. *Edgar Allan Poe: Storyteller*. American Literary Classics. Office of English Language Programs. [Online; accessed 16-July-2021].

Philip Pullella. 2021. Pope appears in public for first time since surgery. *Reuters*. [Online; accessed 11-July-2021].

Reuters. 2021. Best of the Euro 2020. https://www.reuters.com/news/picture/best-of-the-euro-2020-idINRTXDBQR2. [Online; accessed 01-August-2021].

Michael D. Shear. 2021. Biden urges Putin to 'take action to disrupt' Russia-based hackers behind ransomware attacks. *The New York Times*. [Online; accessed 12-July-2021].

Barbara Slater. 2021. UEFA Men's Euro 2020. https://www.bbc.com/mediacentre/mediapacks/euro-2020/. [Online; accessed 29-July-2021].

Rory Smith. 2021. Euro 2020: Italy Beats Turkey, 3-0, in Opener in Rome. *The New York Times*. [Online; accessed 12-July-2021].

James Tapsfield. 2021. Vaccines minister Nadhim Zahawi says people will still be 'expected' to wear masks in confined spaces - as poll shows 50 per cent of Britons want 'Freedom Day' delayed. *Daily Mail*. [Online; accessed 11-July-2021].

The Economist. 2021a. AI is transforming the coding of computer programs. *The Economist Group Limited*. [Online; accessed 11-July-2021].

The Economist. 2021b. Deep in rural China, bitcoin miners are packing up. *The Economist Group Limited*. [Online; accessed 11-July-2021].

The Economist. 2021c. Why life without parole is nearly always too long. *The Economist Group Limited*. [Online; accessed 11-July-2021].

The Guardian. 2021a. Covid live: public in England expected to wear masks when measures lift; Indonesia reports 1,007 daily deaths. *The Guardian*. [Online; accessed 11-July-2021].

The Guardian. 2021b. Euro 2020: buildup to Italy v England final – live! *The Guardian*. [Online; accessed 11-July-2021].

Wikipedia contributors. 2021a. 1860 boden professor of sanskrit election — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=1860_Boden_Professor_of_Sanskrit_election&oldid=1032186102. [Online; accessed 11-July-2021].

Wikipedia contributors. 2021b. 7 world trade center — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=7_World_Trade_Center&oldid=1032951238. [Online; accessed 11-July-2021].

Wikipedia contributors. 2021c. Dance — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Dance&oldid=1026145565. [Online; accessed 3-June-2021].

Wikipedia contributors. 2021d. Plants vs. zombies (video game) — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Plants_vs._Zombies_(video_game)&oldid=1033073566. [Online; accessed 11-July-2021].

Wikipedia contributors. 2021e. Uefa euro 2020 — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=UEFA_Euro_2020&oldid=1033344902. [Online; accessed 12-July-2021].

Wikipedia contributors. 2021f. Uefa euro 2020 qualifying — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=UEFA_Euro_2020_qualifying&oldid=1033315814. [Online; accessed 12-July-2021].

Wikipedia contributors. 2021g. World war i — Wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=World_War_I&oldid=1033325218. [Online; accessed 13-July-2021].

Wikiquote. 2021. Yes, minister — wikiquote,. https://en.wikiquote.org/w/index.php?title=Yes,_Minister&oldid=2943305. [Online; accessed 16-July-2021].

David Williams. 2021. Extreme heat cooked mussels, clams and other shellfish alive on beaches in Western Canada. *CNN*. [Online; accessed 11-July-2021].

Alexander Winning. 2021. Violence spreads to South Africa's economic hub in wake of Zuma jailing. *Reuters*. [Online; accessed 11-July-2021].

Jessie Yeung. 2021. Under Philippine law, 12-year-olds can consent to sex. Activists are trying to change that. *CNN*. [Online; accessed 11-July-2021].

Appendix A

Evaluation results

Table A.0.1

Grammatical construction	# correct	# incorrect	Precision
Conditional: real	9	1	0.9
Conditional: unreal	2	8	0.2
Passive: present simple	9	1	0.9
Passive: past simple	10	0	1.0
Passive: future simple	0	2	0.0
Passive: present perf.	10	0	1.0
Passive: past perf.	10	0	1.0
Passive: future perf.	1	7	0.125
Passive: present prog.	0	0	
Passive: past prog.	0	0	
Passive: future prog.	0	2	0.0
Passive: present perf. prog.	0	4	0.0
Passive: past perf. prog.	1	2	0.3333
Passive: future perf. prog.	0	0	
Active: present simple	6	4	0.6
Active: past simple	10	0	1.0
Active: future simple	10	0	1.0
Active: present perf.	10	0	1.0

Continued on next page

Table A.0.1: Results of the target identification assessment

Table A.0.1 – continued from previous page

Grammatical construction	# correct	# incorrect	Precision
Active: past perf.	10	0	1.0
Active: future perf.	6	4	0.6
Active: present prog.	10	0	1.0
Active: past prog.	10	0	1.0
Active: future prog.	10	0	1.0
Active: present perf. prog.	7	3	0.7
Active: past perf. prog.	5	5	0.5
Active: future perf. prog.	1	1	0.5
Past simple: stmt., affirm., reg.	9	1	0.9
Past simple: stmt., affirm., irreg.	10	0	1.0
Past simple: stmt., neg., reg.	1	9	0.1
Past simple: stmt., neg., irreg.	6	4	0.6
Past simple: quest., affirm., reg.	4	6	0.4
Past simple: quest., affirm., irreg.	4	6	0.4
Past simple: quest., neg., reg.	1	0	1.0
Past simple: quest., neg., irreg.	4	6	0.4
Present perf.: stmt., affirm., reg.	9	1	0.9
Present perf.: stmt., affirm., irreg.	10	0	1.0
Present perf.: stmt., neg., reg.	9	1	0.9
Present perf.: stmt., neg., irreg.	6	4	0.6
Present perf.: quest., affirm., reg.	4	6	0.4
Present perf.: quest., affirm., irreg.	3	7	0.3
Present perf.: quest., neg., reg.	3	1	0.75
Present perf.: quest., neg., irreg.	4	3	0.5714
Past perf.: stmt., affirm., reg.	0	0	
Past perf.: stmt., affirm., irreg.	4	6	0.4
Past perf.: stmt., neg., reg.	0	0	
Past perf.: stmt., neg., irreg.	5	5	0.5
Past perf.: quest., affirm., reg.	0	0	
Past perf.: quest., affirm., irreg.	0	9	0.0

Continued on next page

Table A.0.1: Results of the target identification assessment

Table A.0.1 – continued from previous page

Grammatical construction	# correct	# incorrect	Precision
Past perf.: quest., neg., reg.	0	0	
Past perf.: quest., neg., irreg.	0	1	0.0
Past prog.: stmt., affirm., reg.	0	3	0.0
Past prog.: stmt., affirm., irreg.	3	7	0.3
Past prog.: stmt., neg., reg.	0	0	
Past prog.: stmt., neg., irreg.	1	9	0.1
Past prog.: quest., affirm., reg.	0	0	
Past prog.: quest., affirm., irreg.	0	10	0.0
Past prog.: quest., neg., reg.	0	0	
Past prog.: quest., neg., irreg.	0	1	0.0
Present perf. prog.: stmt., affirm., reg.	0	7	0.0
Present perf. prog.: stmt., affirm., irreg.	3	7	0.3
Present perf. prog.: stmt., neg., reg.	0	1	0.0
Present perf. prog.: stmt., neg., irreg.	0	1	0.0
Present perf. prog.: quest., affirm., reg.	0	1	0.0
Present perf. prog.: quest., affirm., irreg.	0	4	0.0
Present perf. prog.: quest., neg., reg.	0	0	
Present perf. prog.: quest., neg., irreg.	0	0	
Past perf. prog.: stmt., neg., reg.	0	0	
Past perf. prog.: stmt., neg., irreg.	0	2	0.0
Past perf. prog.: quest., affirm., reg.	0	0	
Past perf. prog.: quest., affirm., irreg.	0	0	
Past perf. prog.: quest., neg., reg.	0	0	
Past perf. prog.: quest., neg., irreg.	0	0	
Present simple: stmt., affirm., 3rd pers.	7	3	0.7
Present simple: stmt., affirm., not 3rd pers.	8	2	0.8
Present simple: stmt., neg., 3rd pers.	9	1	0.9
Present simple: stmt., neg., not 3rd pers.	8	2	0.8
Present simple: quest., affirm., 3rd pers.	3	7	0.3
Present simple: quest., affirm., not 3rd pers.	4	6	0.4

Continued on next page

Table A.0.1: Results of the target identification assessment

Table A.0.1 – continued from previous page

Grammatical construction	# correct	# incorrect	Precision
Present simple: quest., neg., 3rd pers.	5	5	0.5
Present simple: quest., neg., not 3rd pers.	4	6	0.4
Relative pronouns: who	10	0	1.0
Relative pronouns: which	9	1	0.9
Relative pronouns: that	9	1	0.9
Relative pronouns: other relative pronoun	0	10	0.0
Adjective: comparative, synthetic	10	0	1.0
Adjective: superlative, synthetic	8	2	0.8
Adjective: comparative, analytic	9	1	0.9
Adjective: superlative, analytic	10	0	1.0
Adverb: comparative, synthetic	9	1	0.9
Adverb: superlative, synthetic	5	0	1.0
Adverb: comparative, analytic	10	0	1.0
Adverb: superlative, analytic	9	1	0.9

Table A.0.1: Results of the target identification assessment

Up to 10 occurrences were randomly sampled for all possible target constructions. From these samples, the numbers of correctly and incorrectly labelled instances were determined and the precision value was calculated.

Table A.0.2

Exercise topic	Processing step	Web site	Execution time	File size
Comparison	Document download	BBC (Slater (2021))	2	.82
	HTML indexing		1	
	NLP processing		<1	
	Resource download		4	
	Feedback generation		538	
	Miscellaneous		38	
	Overall		642	
Passive	Document download	BBC (BBC (2020))	12	3.51
	HTML indexing		11	
	NLP processing		<1	
	Resource download		8	
	Feedback generation		6360	
	Miscellaneous		64	
	Overall		6456	
Past tenses	Document download	BBC (Slater (2021))	2	.83
	HTML indexing		1	
	NLP processing		<1	
	Resource download		4	
	Feedback generation		3159	
	Miscellaneous		21	
	Overall		3197	
Simple present	Document download	BBC (Bevan (2021))	11	3.87
	HTML indexing		1	
	NLP processing		<1	
	Resource download		16	
	Feedback generation		5570	
	Miscellaneous		1	
	Overall		5600	
Relative pronouns	Document download	BBC (Bevan (2021))	13	3.86
	HTML indexing		2	
	NLP processing		<1	

Continued on next page

Table A.0.2: Results of the performance assessment

Table A.0.2 – continued from previous page

Exercise topic	Processing step	Web site	Execution time [s]	File size [MB]
	Resource download		14	
	Feedback generation		356	
	Miscellaneous		9	
	Overall		395	
Conditional clauses	Document download	BBC (BBC (2021c))	3	.85
	HTML indexing		47	
	NLP processing		<1	
	Resource download		14	
	Feedback generation		490	
	Miscellaneous		38	
	Overall		593	
Comparison	Document download	CNN (Grez (2021))	19	8.72
	HTML indexing		4	
	NLP processing		<1	
	Resource download		38	
	Feedback generation		247	
	Miscellaneous		9	
	Overall		318	
Passive	Document download	CNN (Morse (2021))	20	8.59
	HTML indexing		2	
	NLP processing		<1	
	Resource download		36	
	Feedback generation		3697	
	Miscellaneous		6	
	Overall		3762	
Past tenses	Document download	CNN (Grez (2021))	14	8.74
	HTML indexing		2	
	NLP processing		<1	
	Resource download		38	
	Feedback generation		5174	
	Miscellaneous		11	

Continued on next page

Table A.0.2: Results of the performance assessment

Table A.0.2 – continued from previous page

Exercise topic	Processing step	Web site	Execution time [s]	File size [MB]
	Overall		5240	
Simple present	Document download	CNN (Grez (2021))	18	8.72
	HTML indexing		4	
	NLP processing		<1	
	Resource download		29	
	Feedback generation		3776	
	Miscellaneous		14	
	Overall		3842	
Relative pronouns	Document download	CNN (Grez (2021))	19	8.72
	HTML indexing		2	
	NLP processing		<1	
	Resource download		45	
	Feedback generation		285	
	Miscellaneous		4	
	Overall		356	
Conditional clauses	Document download	CNN (Grez (2021))	17	.91
	HTML indexing		4	
	NLP processing		<1	
	Resource download		34	
	Feedback generation		49	
	Miscellaneous		8	
	Overall		113	
Comparison	Document download	Reuters (Holden and Phillips (2021))	1	3.53
	HTML indexing		1	
	NLP processing		<1	
	Resource download		3	
	Feedback generation		195	
	Miscellaneous		6	
	Overall		207	
Passive	Document download	Reuters (Reuters (2021))	9	2.44
	HTML indexing		5	

Continued on next page

Table A.0.2: Results of the performance assessment

Table A.0.2 – continued from previous page

Exercise topic	Processing step	Web site	Execution time [s]	File size [MB]
	NLP processing		<1	
	Resource download		17	
	Feedback generation		4649	
	Miscellaneous		4	
	Overall		4685	
Past tenses	Document download	Reuters and (Holden Phillips (2021))	4	3.55
	HTML indexing		1	
	NLP processing		<1	
	Resource download		5	
	Feedback generation		4857	
	Miscellaneous		7	
	Overall		4875	
Simple present	Document download	Reuters (Reuters (2021))	9	2.44
	HTML indexing		5	
	NLP processing		<1	
	Resource download		4	
	Feedback generation		4255	
	Miscellaneous		24	
	Overall		4298	
Relative pronouns	Document download	Reuters and (Holden Phillips (2021))	2	3.53
	HTML indexing		1	
	NLP processing		<1	
	Resource download		3	
	Feedback generation		186	
	Miscellaneous		16	
	Overall		209	
Conditional clauses	Document download	Reuters (Evans (2021))	3	1.11
	HTML indexing		1	
	NLP processing		<1	
	Resource download		17	
	Feedback generation		138	

Continued on next page

Table A.0.2: Results of the performance assessment

Table A.0.2 – continued from previous page

Exercise topic	Processing step	Web site	Execution time [s]	File size [MB]
	Miscellaneous		2	
	Overall		162	

Table A.0.2: Results of the performance assessment