Journal of Universal Computer Science, vol. 29, no. 8 (2023), 866-891
submitted: 19/5/2022, accepted: 25/4/2023, appeared: 28/8/2023 CC BY-ND 4.0

The evaluation of a semi-automatic authoring tool for knowledge extraction in the AC&NL Tutor

Ani Grubišić
(Faculty of Science, University of Split, Split, Croatia
https://orcid.org/0000-0003-4313-7851, ani.grubisic@pmfst.hr)

Slavomir Stankov
(Retired Full Professor, Split, Croatia
https://orcid.org/0000-0001-8997-7050, slavomir.stankov@gmail.com)

Branko Žitko
(Faculty of Science, University of Split, Split, Croatia
https://orcid.org/0000-0001-8946-0916, branko.zitko@pmfst.hr)

Ines Šarić-Grgić
(Faculty of Science, University of Split, Split, Croatia
https://orcid.org/0000-0002-9247-8890, ines.saric@pmfst.hr)

Angelina Gašpar
(Catholic Faculty of Theology, University of Split, Split, Croatia
https://orcid.org/0000-0002-4472-8648, angelina.gaspar@kbf-st.hr)

Emil Brajković
(Faculty of Science and Education, University of Mostar, Mostar, Bosnia and Herzegovina
https://orcid.org/0000-0002-1726-2029, emil.brajkovic@fpmoz.sum.ba)

Daniel Vasić
(Faculty of Science and Education, University of Mostar, Mostar, Bosnia and Herzegovina
https://orcid.org/0000-0002-7713-8396, daniel.vasic@fpmoz.sum.ba)

Abstract: This paper describes and evaluates the performance of a semi-automatic authoring tool (SAAT) for knowledge extraction in the AC&NL Tutor, highlighting its strengths and weaknesses. We assessed the accuracy of automatic annotation tasks (Part-of-Speech tagging, Name Entity Recognition, Dependency parsing, and Coreference Resolution) performed on a dataset of 160 sentences from unstructured Wikipedia text on a computer. We compared the automatic annotations to the gold standard, created after human post-editing and validation. Human-error analysis included 3769 words, 582 subsentences, 1129 questions, 917 propositions, 1020 concepts, and 667 relations. It resulted in the error type classification and the set of custom rules further used for automatic error identification and correction. The results showed that an average of 68.7% of the error corrections referred to CoreNLP performance and 31.3% to the SAAT extraction algorithms. Our main contributions include an integrated approach to the comprehensive pre-processing of the text, knowledge extraction and visualization; the consolidated evaluation of natural language processing tasks and knowledge extraction output (sentences, subsentences, questions, concept maps) and the newly developed reference dataset.

Keywords: Natural Language Processing, Knowledge Extraction, Automatic Question Generation, Human-Error Analysis, Gold Standard, AC&NL Tutor, Concept Maps
Categories: L.2.0, L.3.0, M.4
DOI: 10.3897/jucs.86745

1 Introduction

One of the main tasks in developing intelligent tutoring systems is domain-knowledge modelling. The AC&NL Tutor (Adaptive Courseware based on Natural Language) is a learning environment with adaptive content and communication based on natural language. The AC&NL Tutor consists of a Semi-Automatic Authoring Tool (SAAT), which involves a teacher who creates learning material, and an intelligent tutoring system (ITS) used only by a learner. The knowledge extraction from unstructured natural language text is a semi-automatic task because the teacher designs/redesigns natural language text to make knowledge extraction output more accurate. However, the SAAT automatically generates concept maps, sentences, and questions of different difficulty. For more on the AC&NL and the SAAT's functionalities, see [Grubišić, 2020] and [Grubišić, 2022].

We observe natural language processing (NLP) in two directions: natural language understanding (NLU), the ability of the authoring tool to 'understand' the unstructured text and use data structures to create new natural language text, the task known as natural language generation (NLG). Natural language processing in our authoring tool converts natural language text (sentences and phrases) into different types of data structures. Therefore, the expression "disassemble to reassemble", defined by [Kowata, 2010], describes the disassembling of the text until its structure is visible and then reassembling it in the form of concept maps, sentences and questions. The role of a teacher in preparing natural language text is to facilitate machine comprehension (disassemble phase) in the SAAT (e. g. restructuring of complex sentences) so that the generation of natural language sentences, questions, and concept maps (reassemble phase) is more accurate.

In our approach, we integrated different available resources and tools such as WordNet 3.1 (wordnet.princeton.edu, Princeton University, 2010), CoreNLP 3.8 (stanfordnlp.github.io/CoreNLP/index.html [Manning, 2014]), Senna SRL 3.0 (ronan.collobert.com/senna [Collobert, 2011], the verb lexicon from XTAG Project (www.cis.upenn.edu/~xtag/). We enhanced them with custom rules to increase their performance (the output quality and precision), hampered by inconsistencies and errors reported in the literature. For example, Mitri [2022] used CoreNLP in Story Analyzer, for sentence splitting, tokenization, part-of-speech tagging, named entity recognition, and coreference resolution. It processed an unstructured dataset consisting of 69 sentences. Named entity recognition and parsing tasks were considerably less accurate in NLP's current state of the art, with accuracy rates in the 80%+ range. Coreference resolution results were far less accurate, reaching 60% accuracy. Those limitations negatively affected the overall performance of the application. To test the Stanford PoS tagger, Manning [2011] conducted an error analysis on a sample of 100 errors from section 19 of the treebank, dividing them into seven classes of which inconsistent and wrong gold standard data due to the lack of clear tagging guidelines, comprised over 40% of the data. In those classes, the author saw opportunities for tagging performance

gains. Another example of the Stanford CoreNLP performance was reported by Nazaruka, Osis and Griberman [2020]. They stated that the parser used for semantic information extraction produced errors in tagging verbs and indicating dependencies between verbs and direct objects. Griffis et al. [2016] used specific (medical) and general-domain (BNC) English corpora to evaluate the accuracy of sentence boundary detection. The primary errors from any toolkit (Stanford, Lingpipe, Splitta, SPECIALIST, cTAKES) and on every corpus were semicolons and colons, treated as sentence separators. Regardless of tools and datasets, some NLP tasks are deeply interrelated (e.g., tokenization, lemmatization, PoS tagging), and errors in these low-level layers propagate to high-level layers (SRL, NER Coreference resolution, WSD). Caselli et al. [2015] reported that error propagation in different pipeline modules (e.g., SRL, NERC and NED) led to poor performance in event timeline extraction. A poor coreference resolution task negatively affected automatic summarization (Droog-Hayes, 2017).

This research aims to examine which knowledge extraction errors are due to SAAT extraction algorithms (applied custom rules) and which of the integrated linguistic resources and tools, specifically CoreNLP. The results of this research can be beneficial for boosting the performance of state-of-the-art NLP tools that rely on almost solved pre-processing tasks. Guided by the related work, we hypothesized that fine-tuning the integrated linguistic resources would enhance SAAT knowledge extraction. So, a research question (RQ) was: To what extent is the SAAT extraction algorithm output determined by the integrated linguistic resources and tools, specifically CoreNLP?

The presented paper is the follow-up work of the conference paper [Grubišić, 2020] which provides more detailed results. It is the consolidated evaluation of generated sentences, questions, concept maps and the reference human-annotated dataset. The contributions of this work are as follows:

1. An initial and brief description of the SAAT was in the conference paper [Grubišić, 2020]. This paper provides detailed descriptions and evaluations of SAAT's modules, supported by diagrams.
2. A reference human-annotated dataset created from the unstructured text on a computer (en.wikipedia.org/wiki/Computer) can be used to evaluate similar approaches to knowledge extraction. The dataset consists of 160 sentences, and its pre-processing yielded 3769 words, 582 subsentences, 1129 questions, 917 propositions, 1020 concepts, and 667 relations.
3. Human error analysis of machine-generated sentences/subsentences and questions resulted in the error type classification and the set of custom rules (incomplete), further used for automatic error identification and correction.

The following section provides the theoretical background of NLP tasks, their accuracy and authoring tools. The third section focuses on the SAAT's functionalities, followed by the Methodology section, describing the study protocol. The results and the discussion section summarize and interpret the research findings. The conclusion highlights key findings and opportunities for future work.

2 Research background

In this paper, we evaluate how the applied custom rules in CoreNLP affect the quality of knowledge extraction output, presented as sentences, subsentences, questions and

concept maps. We searched the Web of Science, Scopus, ACM, IEEE Xplore, and Google Scholar for the most recent systematic reviews on natural language processing, knowledge extraction, concept extraction, relation extraction, automatic question generation, and concept map generation. Related literature review showed that no similar authoring tool performs various NLP tasks in a single environment as the SAAT. So, we referred to works relevant to a specific NLP task.

Horsmann, Erbs and Zesch [Horsmann, 2015] performed a comparative evaluation of 22 tagging models for English and German, using nine PoS tagger implementations. The results indicated that even the most accurate PoS tagging models for English, trained on different genres, achieved accuracy below 90%. Jacobsen, Sørensen and Derczynski [Jacobsen, 2021] evaluated ten PoS taggers across eight languages to see the size vs accuracy trade-off of classical and contemporary taggers. Token accuracy on the test set for Stanford tagger was 93,64%, unlike sentence accuracy, which was 45.17%, indicating the need for substantial improvement. Naseer et al. [Naseer, 2021] presented the advantages and disadvantages of different tools (SpaCy, StanfordNLP, TensorFlow and Apache OpenNLP for named entity recognition (NER). SpaCy NER tool obtained 100% F1score, followed by Tensorflow 97%, OpenNLP 96.5% and Standford 94%. Li et al. [Li, 2022] consolidated NER available NER resources, tagged NER corpora, NER systems, evaluation metrics, traditional approaches to NER, recently applied deep learning techniques and the applications of NER. Ghaddar and Langlais [Ghaddar, 2016] developed WikiCoref, a coreference-annotated corpus of Wikipedia articles and Bamman, Lewke and Mansoor's [Bamman, 2020] dataset of literary coreference. Lu and Poesio [Lu, 2021] provided a survey on coreference resolution for the biomedical domain. Shimorina, Heinecke and Herledan [Shimorina, 2022] developed a knowledge extraction pipeline for English entity detection and linking, coreference resolution, and relation extraction based on the Wikidata schema. Zhang [Zhang, 2020] thoroughly reviewed the past work on syntactic and semantic parsing based on constituent and dependency structures, highlighting that neural network methods with pre-trained contextualized word representations had achieved the top performances for almost all datasets. Also, using neural networks, global features across different tasks could be directly captured by deep LSTMs and self-attention, whereas building one share encoder across tasks could reduce the influence of error propagation.

Most NLP related authoring tools are used for authoring tutorial dialogues, especially checking natural language responses. Knowledge Construction Dialogue (KCD) in Atlas Project [Jordan, 2001] defines the formal grammar for dialogue execution, later supported by external NLP tools. AutoTutor Script Authoring Tool (ASAT) [Susarla, 2003] fills in the conversational content based on rule templates. Regular expressions formalize correct answers in ASAT, and during a run-time, a semantic engine checks correctness. The semantic tool relies on the distributional semantics of a large text. DeepTutor extends such latent semantics with logical entailment [Rus, 2008] and provides computational knowledge representation [Rus, 2013]. ConceptGrid [Blessing, 2015] uses a template-style approach to check sentence-level natural language responses. The author creates a lattice-style structure that contains the required concepts that need to be in a student response. The mentioned authoring tools use previously built semantic models, usually integrated into existing ITSs for natural language conversation design.

As for knowledge representation, we refer to a systematic review of automatic question generation [Kurdi, 2020] and that of [Panchal, 2021] who used various pre-processing NLP tasks (tokenization, NER, POS) for Fill in the blank, multiple choice and Wh type questions. The lack of rules for complex sentence parsing resulted in some ill-formed Wh questions. Authors also reported shortcomings in the NER of the Spacy library, which caused some incorrectly tagged entities, resulting in incorrect or poor questions and the lack of a variety of Wh questions, which require additional handcrafted rules. Automatic and manual evaluation results from the work of [Dhole, 2020] showed that their Syn-QG system, based on syntactic and shallow semantic rules, could generate highly grammatical and relevant questions. [Divate, 2017] summarized automatic question generation approaches and evaluation techniques. In our forthcoming work on question generation [Gašpar, 2023], we evaluated the output of our generic system for automatic factual question generation by using mixed evaluation strategies: 1) human evaluation, 2) qualitative error analysis, 3) automatic evaluation, 4) human and automatic evaluation of machine-generated questions from paraphrases compared to the reference questions, 5) preliminary comparison to other approaches.

Concept mapping denotes the task of creating concept maps ([Chang, 2001], [Novak, 2008]). To automate the concept mapping process, [Villalon, 2008] first introduced the term 'concept map mining' for "the automatic extraction of concept maps from documents such as essays", and indicated that concept maps should neither contain redundant information (synonyms) nor information loss. Kowata et al. [2010] assert that concept map mining must "face the complexity of the natural language" and "produce understandable output to humans" with minimum semantic loss while preserving the main idea of the source. In support of that claim, Zubrinic, Kalpic and Milicevic [2012] state that concept map mining uses NLP methods enriched with linguistic resources or tools and techniques. A review of concept map creation from NLP is provided by [dos Santos, 2018].

Most fully automatic approaches for generating concept maps from the text are based on frequencies of occurrence and co-occurrence of concepts. For example, using Bayesian decision theory, a Leximancer [Smith, 2006] extracts main concepts and their relations from text documents based on their frequency (it ignores the stop words). An automatic concept map constructor (ACMC) [Wafula, 2016] extracts concepts according to their frequency but their potential relations only if they occur in the same sentence and compound concepts only as two consecutive words. On the other hand, some approaches incorporated machine learning and other linguistic techniques to process the text to identify the necessary concepts and relations. For example, a text analysis-association rules mining (TA-ARM) algorithm is based on association rules mining and automatically generates concept maps from students' answer records [Shao, 2020]. In [dos Santos, 2018], a classification enabled detection of the associations between concepts from text documents. An unsupervised clustering algorithm was used by [Qasim, 2013] to extract the structural associations of the candidate terms in the unstructured documents. As for limitations of concept map tools, some concept map mining tools extract concepts and relations without topology, like Knowledge Puzzle [Zouaq, 2007], or retrieve semantic relationships only from predefined ontologies, not from the text ([de la Villa, 2012], [Elhoseiny, 2012], [Zouaq, 2009]).

Regarding the classification proposed by de Aguiar [2017] that refers to the state-of-the-art methods in concept map mining, our approach focuses on a data source that uses the unstructured text of a small size. There is no domain definition such as

predefined ontology, thesaurus, knowledge database or list of concepts, just the data source in the English language. The data source coverage is original, and its precedence is unsupervised. The knowledge extraction process includes linguistic methods. As for the graphic representation, the SAAT displays the concept map in its interface. Its quality is analyzed subjectively, and the lack of human intervention makes the whole process automatic. The category labelling is present (concept map represents a meaningful proposition), and the category connectivity is partially disassociated (not a fully unified map). The style of generated map is scientific, and its organization is a combination of hierarchical and spider web. The purpose of concept map building in our approach is text representation. The SAAT uses the following manipulation methods: tokenization, lemmatization, coreference resolution, named entity recognition, stop word list, synonym detection, anaphora resolution, lexical and syntactic analysis, and dependency parsing.

3 The SAAT's overview

The SAAT uses syntactic and semantic annotations for reading comprehension of natural language text, as shown in Figure 1. Syntactic labelling includes lemmatization, Part-of-Speech (POS Tagging), Named Entity Recognition (NER) and Dependency Parsing (DP). CoreNLP does all these tasks. Dependencies in NLP models enable relation extraction, question answering and other semantic tasks. Dependency grammar identifies dependency relations between headwords and their dependent words. These dependencies indicate who did what to whom in a sentence. Dependency grammar forms a dependency tree that consists of directed edges, linking heads and their dependents, one node representing the root and each node representing a word token. A phrase-structure grammar identifies the phrases within the sentence. It forms a constituency or hierarchical tree, having word tokens as the leaves and internal nodes as constituent phrases such as noun phrases (NP) or prepositional phrases (PP).

Semantic Role Labeling (SRL) refers to the identification of predicate-arguments relations. Senna SRL does the task. Semantic roles are associated with all arguments and modifiers of each predicate in a sentence. To solve lexical ambiguity Word Sense Disambiguation (WSD) method determines the meaning (sense) of a word in a particular context. This method relies on a WordNet resource where the senses of a word group into a set of synonyms (synset). If some words, the string of words (called mentions) in a sentence refer to the mentions in the same or previous sentence, then the task of finding and grouping these mentions is called Coreference Resolution, performed by CoreNLP. Syntactically and semantically annotated natural language text is used for knowledge extraction.

The SAAT uses two layers of knowledge extracted from each natural language sentence:

1. language knowledge – a language knowledge graph, generated from an annotated sentence
2. foreground knowledge – a foreground knowledge graph, generated from the language knowledge graph

The knowledge extractor's task is to take annotated natural language text and generate mentioned layers of knowledge. The first step is to apply rules over text and build the

language knowledge graph out of which the extractor creates the foreground knowledge graph.

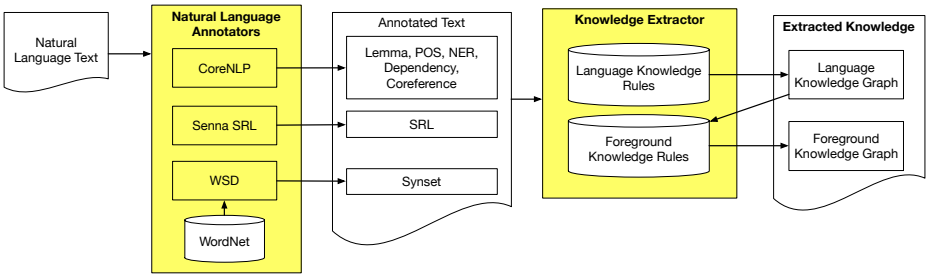


Figure 1: Natural Language Annotation and Knowledge Extraction

3.1 Custom rules for error detection and correction

The SAAT integrates different linguistic resources (WordNet, CoreNLP, Senna SRL, verb lexicon) that we have enhanced with custom rules to increase their precision and reduce inconsistencies and errors. The error in knowledge extraction likely results in the propagation of errors in all knowledge layers. So, the main aim of error detection is to identify grammatically or semantically wrong sentences. The SAAT alerts the teacher to the problematic parts that require correction. Some errors can have more consequences than others, so they are classified as low- and high-level errors. Low-level errors (or warnings) are modifications made by the SAAT to prevent potential high-level errors. The SAAT informs the teacher about their significance (Table 1). High-level errors indicate the need for sentence restructuring.

The most common reasons for the errors in Table 1 are incorrect processing of longer and more complex sentences, especially those that involve subordinated clauses, incorrect use of punctuation, imprecise rules for automatic sentence generation, and imprecision of the integrated resources.

Error message	**Err. level**	**Knowl Layer**	**Rule**	**Corrections made by SAAT**
A subject or object of copula construction contains conjunctions.	Low	Langu age	While transforming copula into a standard verb dependency by replacing triples (Z,nsubj,X), (Z,cop,Y) for ordered language nodes X<Y< Z, with (Y,nsubj,X), (Y,dobj,Z)	Copula is used with conjunction
***Gerund**: The gerund or infinitive {word} is changed to a noun.*	Low	Langu age	If there is a gerund (not preceded by auxiliary verb) and noun infinitive, then they are changed to nouns.	Gerund or infinitive "doing" is changed to noun.
***Language knowledge**: an unknown dependency between {word1} and {word2}.*	High	Langu age	If there is a triple (X,dep,Y)	Unknown dependency between "continued" and "until"

***Part-of-speech**: The past participle {word} is changed to the past tense. {word} is changed to a verb.*	Low	Language	If there is a word X whose POS is VBN and a triple (X,conj,Y), then POS of Y is changed to VBD. If there is a word which is not verb and has outgoing relation which is the subject or object, then POS of that word is changed to verb.	Past participle "aided" is changed to past tense
A word is changed to a verb/ adjective / noun.	Low	Language	If the language node has POS that is not related to its dependencies (i.e., if language node X is a noun and (X,nsubj,Y),(X,dobj,Y) or (X,iobj,Y) then POS of X should be a verb). In that case POS is changed to be related with the dependencies.	"details" is changed to verb
A word is turned into an acronym.	Low	Language	If the language node has a majority of upper letters and denotes an acronym	"ENIAC" is turned into acronym.
***Inner conjunction**: word1 is replaced with word1word2.*	Low	Language	While merging headwords with conjunction function words. If language nodes are ordered as $X_1<\ldots<X_m<Y$ and if $(X_1,conj,X_i)$, $1<i\leq n$, and (Y,rel,X_i) where rel is functional relation (i.e. amod, compound, etc.), than all X_i nodes are replaced with X_iY nodes.	"mechanical" is replaced with "mechanical model" ´
***Open Clausule Complement**: {d} became object of {gov}.*	Low	Language	While replacing open clausule dependency whose dependent is not the verb with object relation. If there is a triple (X,xcomp,Y) where Y is not verb, it is replaced with (X,dobj,Y). If there is (Y,nsubj or nsubjpass,Z), then it is replaced with (X,iobj,Z).	"it" became object of "making"
Word has changed from a proper noun to a common noun.	Low	Language	If there is proper noun X whose lemma is the same as capitalized lemma and if triple (Y,compound,X) exists, then X becomes a common noun. The same happens if X is the first word and it is not an entity.	"Silicon" has changed from proper noun to noun.
Merge headword.	Low	Language	If there is a triple of (X,rel,Y) where rel is functional relation (such as amod, compound, etc.), then these triples are removed and X and Y in other triples are replaced with XY.	"elements" is merged using amod with later word "essential"
***Semantic role**: Verb {verb} has an unknown role for argument {argument}. an extended adverbial role between {word1} and {word2}. Crated role between {word1} and {word2}.*	High	Language	The idea is to put semantic roles into dependency relations. If there is a semantic role triple (X,arg,Y) where X is predicate and Y is argument, then dependency triple (X,rel,Y) is replaced with (X,rel:arg,Y). If such dependency triple does not exists, then (X,SRL:arg,Y) triple is added.	Verb "could write" has unknown role for argument AM-MOD.
***Disambiguation**: Concept {word} is disambiguated by adding/removing 'ing'.*	High	Foreground	The result is a synset or the set of synsets which will make a part of the concept. Error can occur when one word cannot be disambiguated. Algorithm tries to find sub-words that can be disambiguated.	Concept for "implementing " is disambiguated by adding/ removing 'ing'

***Foreground knowledge**: {gov} is badly related with {dep}.*	High	Foregr ound	The idea is to simplify language knowledge relations and use only nodes that have concepts.	"required" is badly related with "Changing"

Table 1: Custom rules for error detection and correction

The Natural Language Generation (NLG) consists in reassembling data structures disassembled through natural language understanding to further generate sentences with their graphic representations and questions.

3.2 Sentence generation

The original natural language sentence (a simple, compound, complex or compound-complex) always occurs at the *1st level*. The *2nd level* includes as many subsentences as there are predicates. The *3rd level* has as many subsentences or subsentence variants as predicates and conjunctions.

Figure 2 shows three levels of generated subsentences for an example sentence (A computer is a device that can be instructed to carry out an arbitrary set of arithmetic or logical operations automatically) and the predicate (v) argument (ARG) relations.

Level 1 (original sentence)	
*A computer **is** a device that can be instructed to carry out an arbitrary set of arithmetic or logical operations automatically.*	
ARG1	A computer
VERB(V)	is
ARG1	A device that can be instructed to carry out an arbitrary set of arithmetic or logical operations automatically.
Level 2 (predicates)	
*A device **can be instructed** to carry out an arbitrary set of arithmetic or logical operations automatically.*	
ARG1	A device
VERB(V)	can be instructed
ARG2	To carry out an arbitrary set of arithmetic or logical operations automatically.
*A computer **is** a device.*	
ARG1	A computer
VERB(V)	is
ARG1	A device
Level 3 (predicates and conjunction)	
*A device **can be instructed** to carry out an arbitrary set of **logical operations** automatically.*	
ARG1	A device
VERB(V)	can be instructed
ARG2	To carry out an arbitrary set of logical operations automatically.
*A device **can be instructed** to carry out an arbitrary set of **arithmetic operations** automatically.*	
ARG1	A device
VERB(V)	can be instructed
ARG2	To carry out an arbitrary set of arithmetic automatically.
*A computer **is** a device.*	
ARG1	A computer
VERB(V)	is
ARG2	A device

Figure 2: Three levels of generated subsentences for an example sentence

The final step in sentence assembly needs adjustment of a verb form, i.e., verb conjugation used later in question and concept map generation. The SAAT relies on a verb lexicon from the XTAG project.

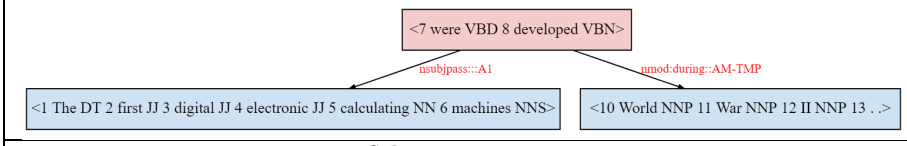
3.3 Question generation

The SAAT generates the following questions for the example sentence and its subsentences (Figure 3). Some errors in generated questions result from original and unproofread text taken from Wikipedia or the lack of more precise question generation rules. Responses to the same question differ at each level. Level 1 question "What is a computer?" requires a response "A device that can be instructed to carry out an arbitrary set of arithmetic or logical operations automatically", but the same question on level 3 requires a short answer "A device". The system will accept all the possible responses.

Level 1	
A computer is a device that can be instructed to carry out an arbitrary set of arithmetic or logical operations automatically.	
Subject	What is a device that can be instructed to carry out an arbitrary set of arithmetic or logical operations automatically?
Answer	A computer
Object	What is a computer?
Answer	A device that can be instructed to carry out an arbitrary set of arithmetic or logical operations automatically.
Level 2	
A device can be instructed to carry out an arbitrary set of arithmetic or logical operations automatically.	
Subject	What can be instructed to carry out an arbitrary set of arithmetic or logical operations automatically?
Answer	A device
Object	What can a device be instructed to?
Answer	To carry out an arbitrary set of arithmetic or logical operations automatically.
A computer is a device.	
Subject	What is a device?
Answer	A computer
Object	What is a computer?
Answer	A device
Level 3	
A device can be instructed to carry out an arbitrary set of logical operations automatically.	
Subject	What can be instructed to carry out an arbitrary set of logical operations automatically?
Answer	A device
Object	What can a device be instructed to?
Answer	To carry out an arbitrary set of logical operations automatically.
A device can be instructed to carry out an arbitrary set of arithmetic operations automatically.	
Subject	What can be instructed to carry out an arbitrary set of arithmetic automatically?
Answer	A device
Object	What can a device be instructed to?
Answer	To carry out an arbitrary set of arithmetic automatically.
A computer is a device.	
Subject	What is a device?
Answer	A computer
Object	What is a computer?
Answer	A device

Figure 3: Three levels of generated questions for an example sentence

The sentence *"The first digital electronic calculating machines were developed during World War II."* has no numeric argument after the predicate node. So, this sentence has language subsentence and subsentence elements, and semantic role `AM-TMP` (temporal) is used to make the adverb question, as presented in Figure 4.



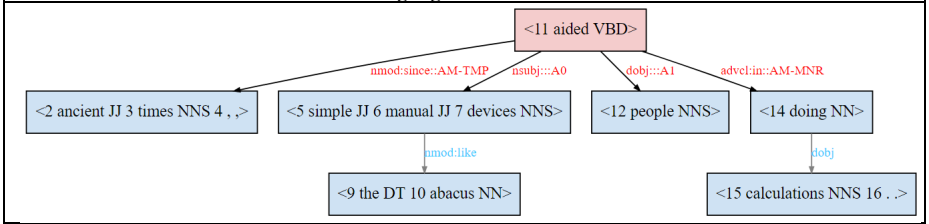
Language Sentence

Subsentence segment

Semantic role	Root	List of nodes in a subgraph
ARG1	1/The 2/first 3/digital 4/electronic 5/calculating 6/machine	
VERB(V)	7/were 8/developed	
AM-TMP	10/World 11/War 12/II 13/.	9/during

Figure 4: Adverb question generation

The example sentence, *"Since ancient times, simple manual devices like the abacus aided people in doing calculations."* has language subsentence and its elements, which include two adverbial modifiers (AM-TMP (temporal), AM-MNR (manner)) as non-core arguments (Figure 5).



Language Subsentence

Subsentence Segment

Semantic role	Root	List of nodes in a subgraph
AM-TMP	2/ancient 3/times 4/,	1/Since
ARG0	5/simple 6/manual 7/devices	8/like 9/the 10/abacus
VERB(V)	11/aided	
ARG1	12/people	
AM-MNR	14/doing	13/in 15/calculations 16/.

Figure 5: The example sentence with core and non-core arguments

The SAAT generates the subject and object questions using core arguments `ARG0` and `ARG1` or replacing them with the question words *what* and *who*. As for adverb questions, non-core arguments are replaced with the question words referring to time and manner, and they are appended to the core arguments of the sentence. Questions generated for this sentence, according to the question type and animacy, are:

- adverbial modifier AM-TMP: When did simple manual devices like the abacus aid people?
- subject ARG0: What aided people in doing calculations?
- object ARG1: Who did simple manual devices like the abacus aid?
- adverbial modifier AM-MNR: How did simple manual devices like the abacus aid people?

The placement of the main verb depends on the predicate node and its active or passive voice construction. To generate the subject question for the example sentence in the passive, *"The first digital electronic calculating machines were developed during World War II."*, auxiliary and main verbs follow the question word *what*. As for the adverb question, the question word *when* precedes the auxiliary verb, whereas the main verb moves to the final position. The order of other words depends on the placement of auxiliary or main verbs, as illustrated in generated questions:

- subject ARG1: What was developed during World War II?
- adverb modifier AM-TMP: When were the first digital electronic calculating machines developed?

If we improve simple rules for the reordering of words in subsentences, we will improve the accuracy of questions.

3.4 Concept map generation

Knowledge representation is the blend of knowledge graph and concept map that consists of vertices (language nodes) and arcs (predicate language nodes). A pair of language and predicate nodes is called knowledge triple or knowledge proposition. The knowledge proposition consists of a parent, relation (the predicate) and child. The parent and child represent concepts. Figure 6 shows the predicate foreground node `be.v.01`, having the knowledge proposition `(computer, is, device)`, i. e. `computer` as the parent, `is` as relation and `device` as a child. The node `arbitrary_set.n.00` presents the `CMOD:of` relation with nodes `logical_operation.n.01` and `arithmetic.n.01`.

If there is a sequence of `COBJ` relations, all nodes are related to the domain knowledge concept. Figure 6 illustrates the relationship between the node `carry_out.n.00` and the node `arbitrary_set.n.00`, based on the `COBJ` relation. These nodes create the domain knowledge concept `carry out arbitrary set`.

The second sentence in Figure 6 shows that if the modifier foreground relation (`ROBJ:as`) with some foreground node (`control_system.n.01`) creates the predicate foreground node (`use.v.02`, and some other foreground node `(computer)` forms the subject foreground relation (`RSUBJPASS`), the dependent of the subject foreground relation becomes the domain knowledge parent (`computer`); the dependent of the modifier foreground relation becomes the domain knowledge child (`control system`), and the predicate node becomes the domain knowledge relation (`is used as`). The relation name normalization includes all nodes having the function words of foreground knowledge relation in their final position. The normalized knowledge proposition is (`computer, is used as, control system`).

To boost the performance of the integrated resources (the WordNet 3.1, the CoreNLP 3.8, the Senna SRL 3.0, the verb lexicon from the TAG Project), custom rules were created.

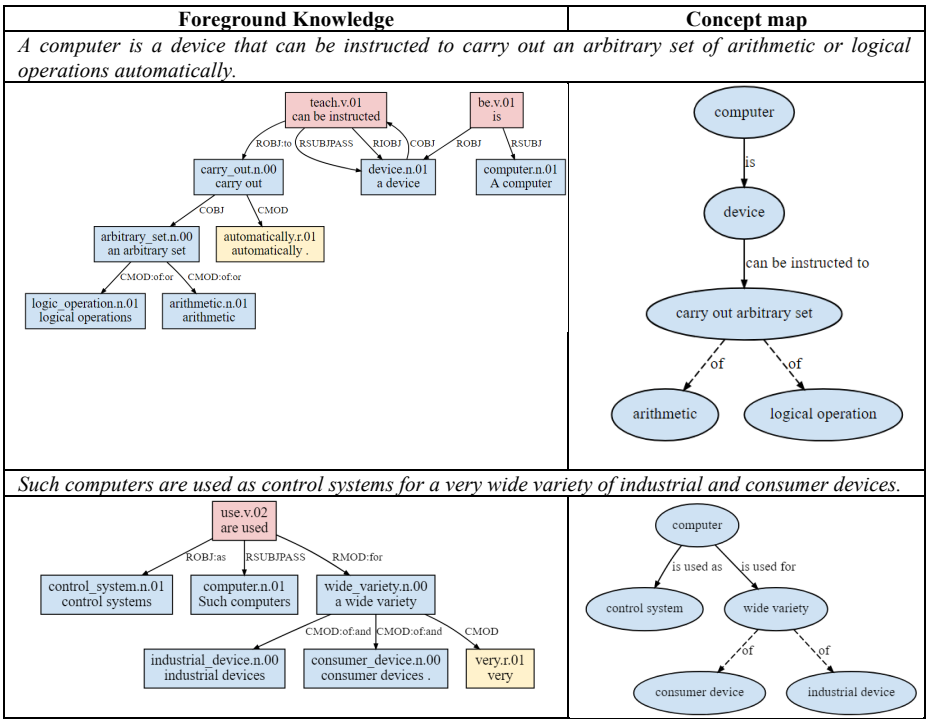
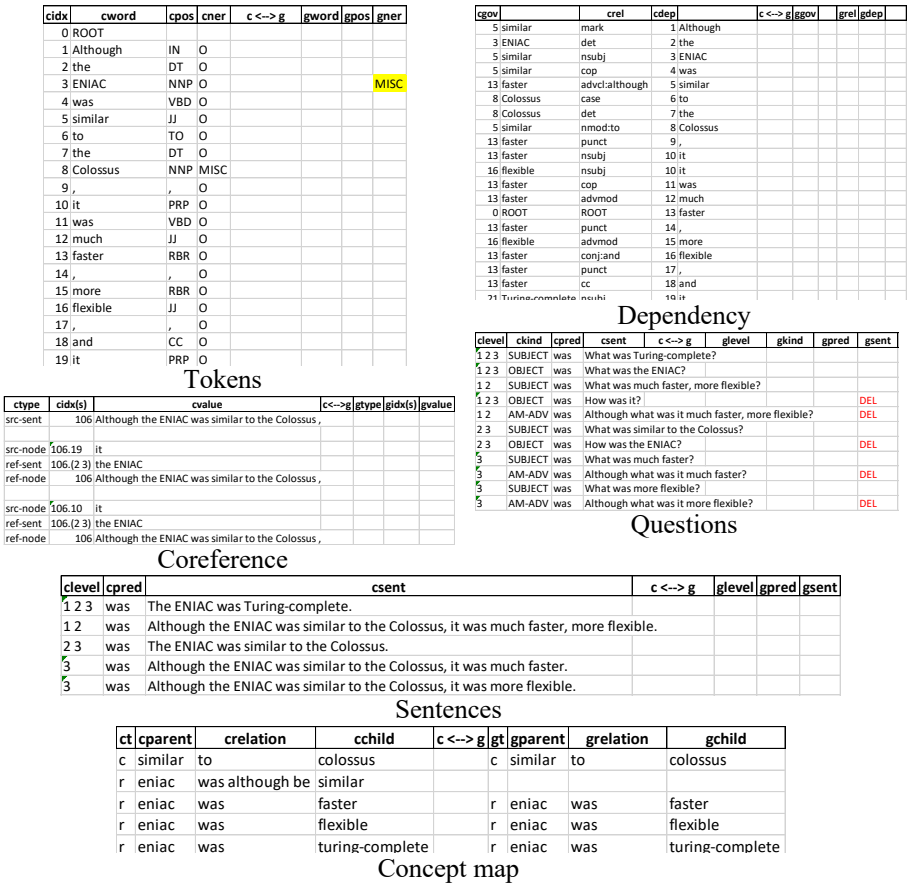


Figure 6: Foreground Knowledge and corresponding concept map for an example sentence

4 Methodology

Methodology used in this research include automatic annotation and knowledge extraction tasks performed by the SAAT on a subset (the first 160 sentences) of the unstructured text on a computer, taken from a Wikipedia article (https://en.wikipedia.org/wiki/Computer, May 2018). We took the text from Wikipedia to show how well the system performed on such unedited text. Then, the annotator analyzed the annotation and extraction outputs. The annotator had to check the accuracy of data in the rows of each spreadsheet (per each sentence) and correct erroneous data (columns having the prefix letter "g") – a false positive (FP). If machine annotations missed some data, the annotator inserted them in a column having the prefix letter "g" – a false negative (FN). Correctly annotated rows were left empty as – a true positive (TP). The annotator filled true positives in the concept map row. Finally, for all annotations, except the concept map, TP has a value in the c-column and not in the g-column; FN has a value in the c-column and g-column, and FP has a "DEL" value in the g-column. For concept map annotations, TP has the same values in the c-column and g-column; FN contains values only in the c-columns, while FP contains values only in the g-columns.

After human post-editing, a reference dataset consists of a .xlsx file, created for each sentence, containing the sheets shown in Figure 7. Each sheet in the .xlsx file consists of columns whose headings have the prefix letter 'c' denoting computer (machine) annotations (cword, cpos) and prefix letter 'g' indicating human (gold standard) annotations or modifications.



cidx	cword	cpos	cner	c <--> g	gword	gpos	gner
0	ROOT						
1	Although	IN	O				
2	the	DT	O				
3	ENIAC	NNP	O				MISC
4	was	VBD	O				
5	similar	JJ	O				
6	to	TO	O				
7	the	DT	O				
8	Colossus	NNP	MISC				
9	,	,	O				
10	it	PRP	O				
11	was	VBD	O				
12	much	JJ	O				
13	faster	RBR	O				
14	,	,	O				
15	more	RBR	O				
16	flexible	JJ	O				
17	,	,	O				
18	and	CC	O				
19	it	PRP	O				

Tokens

cgov		crel	cdep		c <--> g	ggov		grel	gdep	
5	similar	mark	1	Although						
3	ENIAC	det	2	the						
5	similar	nsubj	3	ENIAC						
5	similar	cop	4	was						
13	faster	advcl:although	5	similar						
8	Colossus	case	6	to						
8	Colossus	det	7	the						
5	similar	nmod:to	8	Colossus						
13	faster	punct	9	,						
13	faster	nsubj	10	it						
16	flexible	nsubj	10	it						
13	faster	cop	11	was						
13	faster	advmod	12	much						
0	ROOT	ROOT	13	faster						
13	faster	punct	14	,						
16	flexible	advmod	15	more						
13	faster	conj:and	16	flexible						
13	faster	punct	17	,						
13	faster	cc	18	and						
21	Turing-complete	nsubj	19	it						

Dependency

ctype	cidx(s)	cvalue	c<-->g	gtype	gidx(s)	gvalue
src-sent	106	Although the ENIAC was similar to the Colossus ,				
src-node	106.19	it				
ref-sent	106.(2 3)	the ENIAC				
ref-node	106	Although the ENIAC was similar to the Colossus ,				
src-node	106.10	it				
ref-sent	106.(2 3)	the ENIAC				
ref-node	106	Although the ENIAC was similar to the Colossus ,				

Coreference

clevel	ckind	cpred	csent	c <--> g	glevel	gkind	gpred	gsent
1 2 3	SUBJECT	was	What was Turing-complete?					
1 2 3	OBJECT	was	What was the ENIAC?					
1 2	SUBJECT	was	What was much faster, more flexible?					
1 2 3	OBJECT	was	How was it?					DEL
1 2	AM-ADV	was	Although what was it much faster, more flexible?					DEL
2 3	SUBJECT	was	What was similar to the Colossus?					
2 3	OBJECT	was	How was the ENIAC?					DEL
3	SUBJECT	was	What was much faster?					
3	AM-ADV	was	Although what was it much faster?					DEL
3	SUBJECT	was	What was more flexible?					
3	AM-ADV	was	Although what was it more flexible?					DEL

Questions

clevel	cpred	csent	c <--> g	glevel	gpred	gsent
1 2 3	was	The ENIAC was Turing-complete.				
1 2	was	Although the ENIAC was similar to the Colossus, it was much faster, more flexible.				
2 3	was	The ENIAC was similar to the Colossus.				
3	was	Although the ENIAC was similar to the Colossus, it was much faster.				
3	was	Although the ENIAC was similar to the Colossus, it was more flexible.				

Sentences

ct	cparent	crelation	cchild	c <--> g	gt	gparent	grelation	gchild
c	similar	to	colossus		c	similar	to	colossus
r	eniac	was although be	similar					
r	eniac	was	faster		r	eniac	was	faster
r	eniac	was	flexible		r	eniac	was	flexible
r	eniac	was	turing-complete		r	eniac	was	turing-complete

Concept map

Figure 7: The .xlsx file example

5 Results and discussion

To investigate our research question (RQ), we observed the accuracy of machine annotations compared to the reference dataset, the quality of knowledge extraction outputs (the accuracy of generated sentences/subsentences, questions and concept maps), and the effectiveness of the extraction algorithm.

5.1 Linguistic annotations

We compared machine annotations to the human-validated output at the sentence level. For each sentence and observed features, we calculated precision P, recall R, and their harmonic mean the F1 [Manning, 1999], [Jia, 2018]. We also calculated the mean and standard deviation for the whole dataset of 160 sentences.

Results from Table 2 indicate that the SAAT performed linguistic annotations very well since the calculated values for word, POS, NER, and dependency were over 96% (the annotation task performed by CoreNLP, enhanced by custom rules). High precision and high recall suggested that linguistic annotations were almost entirely accurate. The obtained results were not good for the automatic extraction of concept maps. We gained the lowest scores for natural language sentence and question generation.

	Mean	**sd**		**mean**	**sd**		**mean**	**sd**
word_p	1.000	0.002	**word_r**	0.999	0.005	**word_f1**	1.000	0.003
pos_p	0.973	0.048	**pos_r**	0.972	0.048	**pos_f1**	0.972	0.048
ner_p	0.965	0.066	**ner_r**	0.964	0.066	**ner_f1**	0.965	0.066
dep_p	0.936	0.114	**dep_r**	0.939	0.113	**dep_f1**	0.938	0.113
coref_p	0.516	0.493	**coref_r**	0.675	0.468	**coref_f1**	0.585	0.479
sent_p	0.281	0.209	**sent_r**	0.491	0.413	**sent_f1**	0.512	0.168
quest_p	0.332	0.149	**quest_r**	0.534	0.316	**quest_f1**	0.444	0.165
cmap_p	0.61	0.334	**cmap_r**	0.616	0.321	**cmap_f1**	0.669	0.268
conc_p	0.814	0.214	**conc_r**	0.841	0.184	**conc_f1**	0.832	0.172
rel_p	0.772	0.278	**rel_r**	0.808	0.258	**rel_f1**	0.800	0.231

Table 2: Precision, recall and F1 measures

5.2 Correlations

We calculated the Pearson's correlation coefficient ($p<0.01$) for all variables and their features as TP, FN, FP, P, R and F1 to examine any correlation between the observed features.

We found positive and statistically significant correlations between correctly annotated linguistic variables (TP) and correctly annotated concepts and relations (TP). Correct concept extraction variable correlates strongly with all linguistic variables (their correct annotations – TP): word r=0.767, pos r=0.766, ner r=0.790, dep r=0.779. Correct relation extraction variable correlates moderately with linguistic variables: word r=0.522, pos r=0.529, ner r=0.532, dep r=0.499. Correlation between sentence generation and linguistic variables is moderate (word r=0.390, pos r=0.389, ner r=0.404, dep r=0.441) and slightly higher for question generation (word r=0.403, pos r=0.407, ner r=0.417, dep r=0.443). There are moderate correlations between correct concept extraction and correct sentence and question generation (r=0.519).

There were strong positive and statistically significant correlations ($r>0.97$) between all obtained values of correctly annotated linguistic variables (TP). Good tokenization results influenced the quality of POS, NER and dependency annotations. The incorrect annotations (FP) and non-annotations (FN) did not show significant correlations with other variables in this group. Regarding text variables, there were strongly positive and statistically significant correlations between correct sentences and questions (TP r=0.812), incorrect sentences and questions (FP r=0.879) and not-generated sentences and questions (FN r=0.804), as expected because the tool generated questions from sentences. As for the map variables, there were strongly positive and

statistically significant correlations between correct propositions and sentences (TP r=0.611) and questions (TP r=0.606). The accuracy of extracted concepts and relations positively affected propositions. We calculated a strong positive and statistically significant correlation between non-generated propositions (FN) and incorrect propositions (FP) ($r=0.809$, $p<0.01$). The presented results correspond to the calculated precision and recall, requiring the improvement of the knowledge extraction process (the sentence, question, and concept map generation). In future research, we will examine whether human experts can produce the same gold standard without being guided by the SAAT output. We will also analyze how often the expert added new sentences and questions that were not simply corrections of errors.

5.3 The accuracy of machine-annotations compared to the reference dataset

Besides correlations, we observed the accuracy of machine annotations compared to the reference dataset.

5.3.1 Linguistic variables

The first group of variables were linguistic variables. Out of 3769 words from 160 sentences, 3766 (99.9%) words were correct, 3673 (97.5%) of them had proper POS, and 3633 (96.4%) had correct NER. Dependency relations were accurate for 93.9% of words (out of 3959), and only 5.5% (217) required correction due to incorrect governor (81.1%), relation (57.6%) or dependent (1%). In 92 out of 160 sentences, 50.0% (46) coreference annotations were correct, 26.1% (24) added manually, and 23.9% (22) required correction. All 22 sentences had the wrong co-referring word, and 11 sentences wrongly found their coreferences in other sentences.

5.3.2 Text variables

The second group included text variables (sentences and questions) which we analyzed separately.

Sentences

Out of 582 generated subsentences, only 241 (41.8%) were grammatically and semantically correct, so we removed 86 (14.8%) subsentences. The SAAT failed to create 16 (2.8%) subsentences. 239 (41.1%) subsentences required correction (only four predicate corrections). The annotator manually corrected 313 errors to obtain valid subsentences (some had more than one error). There were 138 (41.6%) grammatical and punctuation errors: punctuation (50), case (37), word order (14), a copula (13), prepositions (8), articles (4), genitive (4), conjunction (3), plural/singular (3) and appositions (2). The SAAT made 67 (21.4%) coreference errors. The annotator inserted 41 (13.1%) words or phrases. There were 29 (9.3%) insertions or replacements of verbs and tense changes; 24 (7.7%) deletions of words or phrases; 14 (4.5%) replacements of words or phrases.

We calculated the word-level edit distance ratio between machine-generated subsentences and those corrected by the expert ("The minimum edit distance between two strings is the minimum number of insertions, deletion, and substitution, needed to transform one into the other", web.stanford.edu). The edit distance ratio of value 1 indicated no differences, i. e. the strings were the same. We obtained the mean value of

0.855 (with 0.141 standard deviations) for all erroneous subsentences. This measure also indicated that most corrections were simple ones. Briefly, out of 582 generated subsentences, 319 (54.8%) were grammatically and semantically correct and 86 (14.8%) were deleted. There are 16 (2.8%) subsentences that the SAAT did not generate. The annotator inserted, deleted, or replaced some words or phrases in only 161 (27.7%) subsentences.

The SAAT successfully generated subsentences of different levels of complexity for 63 out of 160 original natural language sentences. The errors in the remaining 97 sentences were mainly related to CoreNLP performance for 71 (73.2%) original natural language sentences (Table 3).

Table 3, representing the 15 combinations of linguistic variables, contains the number of erroneous sentences related to each variable. For example, combination 3 contains one incorrect sentence out of 97 (0.6%), having POS, NER and Coreference errors. We calculated the total number of erroneous sentences, e.g. having wrong POS, as the sum of all sentences from the combinations 1, 2, 3, 4, 6, 7, 8 and 12. Table 3 shows that 42 (combinations 12-15) of the 97 erroneous sentences had an error related to only one of the linguistic variables, 21 sentences (combinations 6-11) had an error related to two of the linguistic variables, 7 (combinations 2-5) with three variables and only one (combination 1) with all linguistic variables. The errors detected in the remaining 26 sentences (97 minus 71S) did not relate to the linguistic variables.

	POS		**NER**		**Dependency**		**Coreference**		**No. of erroneous sentences for each combination of errors**
1	error		error		error		error		1
2	error		error		error				2
3	error		error				error		1
4	error				error		error		1
5			error		error		error		3
6	error		error						2
7	error				error				5
8	error						error		1
9			error		error				7
10			error				error		1
11					error		error		5
12	error								1
13			error						7
14					error				15
15							error		19
Total	14 S	14.4 %	24 S	24.7 %	39 S	40.2 %	32 S	33.0 %	Total 71 S

Table 3: Number of original sentences (in total 160) and sources of errors in sentence (S) generation

Questions

Out of 1129 generated questions, only 505 (44.7%) were grammatically and semantically correct. The annotator removed 175 (15.5%) meaningless questions. There were 49 (4.3%) questions that the SAAT did not generate. 400 (35.4%) questions required corrections (only 8 the predicate correction). The annotator made 513 corrections (some questions required more than one correction) to transform machine-generated questions into valid ones. Most of the corrections 185 (36.1%) were the following ones: punctuation (49), case (42), a copula (38), prepositions (31), articles (9), word order (9), plural/singular (2), appositions (2), pronouns (2) and

demonstratives (1). Further, corrections included 75 (14.6%) wrong interrogative pronouns; 66 (12.9%) insertions or replacement of verbs and tense change; 62 (11.8%) resolving coreference errors; 56 (10.9%) deletions of words or phrases; 33 (6.4%) replacements of words or phrases; 26 (5.1%) insertions of words or phrases; 6 (1.1%) genitive error and 4 (0.8%) conjunction errors. The calculated edit distance ratio had a mean value for all questions that needed correction of 0.816 (with 0.151 standard deviations).

Out of 1129 machine-generated questions, 613 (54.3%) were grammatically and semantically correct. The annotator removed 175 (14.8%) questions. The SAAT did not generate 49 (4.3%) questions. 292 (25.9%) questions had wrong interrogative pronouns, missed, misused or surplus words or phrases and coreference errors.

The SAAT successfully generated questions of different levels of complexity for 42 out of 160 original natural language sentences. The errors in the remaining 118 sentences were mainly related to CoreNLP performance for 74 (62.7%) original natural language sentences (Table 4).

	POS		**NER**		**Dependency**		**Coreference**		**No. of incorrect questions for each combination of errors**
1	error		error		error		error		2
2	error		error		error				3
3	error		error				error		2
4	error				error		error		1
5			error		error		error		3
6	error		error						2
7	error				error				7
8	error						error		2
9			error		error				8
10			error				error		1
11					error		error		5
12	error								0
13			error						8
14					error				14
15							error		16
Total	19 Q	16.1 %	29 Q	24.6 %	43 Q	36.4 %	30 Q	25.4 %	Total 74 Q

Table 4: Number of original sentences (in total 160) and sources of errors in question (Q) generation

5.3.3 Concept map variables

The third group of variables were concept map variables. We present a separate analysis for propositions, concepts, and relations.

Propositions

Out of 917 propositions, only 489 (53.3%) were correct. The annotator deleted 72 (7.9%) propositions. The SAAT did not generate 60 (6.5%) propositions. 296 (32.3%) propositions required the correction of at least one concept or relation. The calculated edit distance ratio had a mean value for all corrected propositions of 0.611 (with 0.440 standard deviations) for corrected parent concept, 0.705 (with 0.364 standard deviations) for corrected relation and 0.700 (with 0.401 standard deviations) for corrected child concept.

The SAAT successfully extracted propositions for 35 (out of 160 original) natural language sentences. The errors in the remaining 125 sentences were mainly related to CoreNLP performance for 97 (77.6%) original natural language sentences (Table 5).

	POS		**NER**		**Dependency**		**Coreference**		**No. of incorrect propositions for each combination of errors**
1	error		error		error		error		1
2	error		error		error				4
3	error		error				error		1
4	error				error		error		1
5			error		error		error		3
6	error		error						0
7	error				error				13
8	error						error		3
9			error		error				12
10			error				error		0
11					error		error		8
12	error								2
13			error						4
14					error				22
15							error		18
Total	30 P	24.0%	30 P	29.1%	69 P	55.2%	36 P	28.8%	Total 97 P

Table 5: Number of original sentences (in total 160) and sources of errors in proposition (P) generation

Concepts

Out of 1020 identified concepts, 805 (78.9%) were correct. The annotator deleted 40 (3.9%) concepts. The SAAT did not extract 65 (6.4%) concepts. The annotator corrected 110 (10.8%) concepts and made 112 corrections (only two concepts required two corrections) to transform machine-generated concepts into valid ones. Those corrections included 48 (42.9%) insertions of words or phrases; 23 (20.5%) resolving coreferences; 18 (16.1%) replacements of words or phrases; 9 (8.0%) deletions of words or phrases. Other corrections 9 (8.0%) included plural/singular (4), prepositions (2), articles (1), word order (1), apposition (1), 3 (2.7%) verb changes, and 2 (1.8%) conjunctions. The calculated edit distance ratio for all corrected concepts was 0.462 (with 0.319 standard deviations). The calculated values indicated that most error corrections included insertions.

The SAAT successfully extracted concepts for 57 out of 160 original natural language sentences. The errors in the remaining 103 sentences were related to CoreNLP performance for 75 (72.8%) original natural language sentences (Table 6).

	POS	**NER**	**Dependency**	**Coreference**	**No. of incorrect concepts for each combination of errors**
1	error	error	error	error	1
2	error	error	error		4
3	error	error		error	1
4	error		error	error	0
5		error	error	error	2
6	error	error			0
7	error		error		9

8	error						error		2
9			error		error				8
10			error				error		1
11					error		error		6
12	error								1
13			error						5
14					error				14
15							error		21
Total	18 C	17.5%	22 C	21.4%	44 C	42.7%	34 C	33.0%	Total 75 C

Table 6: Number of original sentences (in total 160) and sources of errors in concept (C) generation

Relations

Out of 667 identified relations, 478 (71.7%) were correct. There were 35 (5.2%) deleted relations. The SAAT did not extract 55 (8.2%) relations. The annotator corrected 99 (14.8%) relations and made 101 corrections (only two relations required two corrections) to transform machine-generated relations into valid ones. Those corrections included 73 (72.3%) insertions or replacements of verbs and tense changes. Other corrections 20 (19.8%) included copulas (6), prepositions (13), and word order (1).

Furthermore, corrections included 3 (3.0%) deletions, 3 (3.0%) replacements, and only 2 (2.0%) insertions of words or phrases. The calculated edit distance ratio for all corrected relations was 0.444 (with 0.305 standard deviations). The calculated measure confirmed that that most error corrections were very invasive in verb manipulation.

The SAAT successfully extracted relations for 69 out of 160 original natural language sentences (Table 7). The errors in the remaining 91 sentences were related to CoreNLP performance for 52 (57.1%) original natural language sentences.

	POS		**NER**		**Dependency**		**Coreference**		**No. of incorrect relations for each combination of errors**
1	error		error		error		error		0
2	error		error		error				0
3	error		error				error		0
4	error				error		error		0
5			error		error		error		0
6	error		error						0
7	error				error				10
8	error						error		0
9			error		error				2
10			error				error		0
11					error		error		1
12	error								3
13			error						1
14					error				31
15							error		4
Total	13 R	14.3%	3 R	3.3%	44 R	48.4%	5 R	5.5%	Total 52 R

Table 7: Number of original sentences (in total 160) and sources of errors in relation (R) generation

5.4 Summary of the results

In total, 41.8% of generated subsentences were correct, and 41.1% needed corrections. Subsentences generated from 39.4% (63) of original natural language sentences were accurate. While 73.2% of error corrections were affected by CoreNLP performance, only 26.8% were due to the proposed extraction algorithms.

Regarding question generation, we still have problems to resolve. Namely, 44.7% of the generated questions were correct, and 35.4% required corrections. Questions generated from 26.3% (42) of original natural language sentences were accurate. While 62.7% of error corrections were affected by CoreNLP performance, 37.3% were due to our extraction algorithms. Since the edit distance showed discrepancies between machine-generated and human-corrected subsentences and questions, we plan to improve this aspect of the SAAT in future work.

In total, 53.3% of the generated propositions were correct, and 32.2% required corrections. The propositions generated from 21.9% (35) of original natural language sentences were accurate. While 77.6% of error corrections were due to CoreNLP, only 22.4% were due to knowledge extraction algorithms.

We obtained better results for concept extraction. Namely, 78.9% of extracted concepts were correct, and only 10.8% needed corrections. The concept extraction from 35.6% (57) of original natural language sentences was accurate. The performance of CoreNLP resulted in 72.8% error corrections, and only 27.2% were due to the knowledge extraction algorithm.

We obtained good results for relation extraction. Overall, 71.7% of extracted relations were accurate and 14.8% required corrections. The relation extraction from 43.1% (69) of original natural language sentences was accurate. While 57.1% of relation corrections were due to CoreNLP performance, 42.9% were due to the extraction algorithm.

With these results, we can answer our research question (RQ): To what extent is the SAAT extraction algorithm output determined with the integrated linguistic resources and tools, specifically CoreNLP? The errors identified in the SAAT output were mainly related to CoreNLP performance: 73.2% for sentences, 62.7% for questions, 77.6% for propositions, 72.8% for concepts and 57.1% for relations. In other words, error analysis and the results showed that an average of 68.7% of the error corrections referred to the efficiency of the integrated linguistic resources, specifically CoreNLP, whereas 31.3% referred to the extraction algorithm. Therefore, to boost SAAT performance, we need to enhance CoreNLP performance. To do so, all modules in the pipeline require better integration, a rich semantic model, and improved pre-processing tasks to restrict error propagation.

The SAATs extraction gained better results than the ones from [Jacobsen, 2021] where sentence accuracy was only 45.17%. Unfortunately, the SAAT extraction was correct in only 50% of coreference cases, lower than in [Mitri, 2022], with coreference resolution results reaching 60% accuracy. The SAAT extraction results correspond to those reported in the literature, indicating that NER, parsing, POS tagging, SRL, and coreference tasks in the CoreNLP were less accurate due to error propagation, inconsistencies and errors in the gold standard, sentence boundary detection, or semantic analysis. The consolidated evaluation also highlighted that high accuracy scores achieved for the individual NLP tasks could hardly be maintained when interrelated with other ones.

6 Conclusion

In this paper, we examined how the applied custom rules in CoreNLP affected the quality of knowledge extraction, represented textually as sentences, subsentences and questions and visually as concept maps. To evaluate the accuracy of machine annotations compared to the reference dataset, we observed POS, NER, dependency, and coreference (where applicable) for each word. For each generated subsentence and question, for each generated proposition, concept and relation, we observed whether they were grammatically and semantically correct and which errors should be corrected and how (this was how we created custom rules). These errors included: punctuation, case, word order, copula, prepositions, articles, genitive, conjunction, copulas, plural/singular, appositions, demonstratives, coreferences, wrong interrogative pronouns, insertions of words or phrases, insertions or replacements of verbs and tense changes, replacements and deletions of words or phrases.

Hand-crafted grammatical rules cannot cover the broad range of language uses because ambiguous and idiosyncratic human language evolves fast. NLP tools use statistical methods and machine learning techniques to learn from data fed by humans. A detailed analysis of error-prone machine annotations indicated the SAAT disadvantages, affected by the performance of CoreNLP and our knowledge extraction algorithm. Besides the quantitative data, the experience of the human teacher in using the SAAT is positive. An instructional unit design requires critical thinking, summarizing skills and linguistic competence. To assess how usable and efficient the SAAT is in supporting teachers' instructional unit designing, we will compare human-authored instructional units with ones produced by the SAAT. We plan to develop an automatic knowledge extraction model using state-of-the-art methods and a larger dataset for its evaluation in future research.

Acknowledgements

The presented results are the outcome of the research project “Adaptive Courseware based on Natural Language Processing (AC & NL Tutor)” undertaken with the support of the United States Office of Naval Research Grant (N00014-15-1-2789).

References

[Bamman, 2020] Bamman, D., Lewke, O., Mansoor, A.: An Annotated Dataset of Coreference in English Literature, In Proceedings of the Twelfth Language Resources and Evaluation Conference. Marseille, France: European Language Resources Association, 2020, 44–54.

[Blessing, 2015] Blessing, S. B., Devasani, S., Gilbert, S. B., Sinapov, J.: Using ConceptGrid as an easy authoring technique to check natural language responses, International Journal of Learning Technology, 10, 1, 2015, 50–70.

[Caselli, 2015] Caselli, T., Vossen, P. T. J. M., Van Erp, M., Fokkens-Zwirello, A. S., Ilievski, F., Izquierdo, R., Le, M. N., Morante, R., Postma, M. C.: When it’s all piling up: investigating error propagation in an NLP pipeline, In Proceedings of the Workshop on NLP Applications: Completing the Puzzle, {WNACP} 2015, co-located with the 20th International Conference on Applications of Natural Language to Information Systems {(NLDB} 2015), Passau, Germany, June 17-19, 2015.

[Chang, 2001] Chang, K. E., Sung, Y. T., Chen, S. F.: Learning through computer-based concept mapping with scaffolding aid, Journal of Computer Assisted Learning, 17, 1, 2001, 21–33. https://doi.org/10.1111/j.1365-2729.2001.00156.x

[Collobert, 2011] Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., Kuksa, P.: Natural Language Processing (Almost) from Scratch, Journal of Machine Learning Research, 12, 2011, 2493-2537

[de Aguiar, 2017] de Aguiar, C. Z.: Concept Maps Mining for Text Summarization, PhD Thesis, Universidade Federal do Espírito Santo, ES, Brasil (2017).

[de la Villa, 2012] de la Villa, M., Aparicio, F., Maña, M. J., de Buenaga, M.: A Learning Support Tool with Clinical Cases Based on Concept Maps and Medical Entity Recognition, In Proceedings of the 2012 ACM International Conference on Intelligent User Interfaces. New York, NY, USA: ACM, 2012, 61–70. https://doi.org/10.1145/2166966.2166978

[Dhole, 2020] Dhole, K., Manning, C. D.: Syn-QG: Syntactic and Shallow Semantic Rules for Question Generation, In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. Online: Association for Computational Linguistics, 2020, 752–765. https://doi.org/10.18653/v1/2020.acl-main.69

[Divate, 2017] Divate, M., Salgaonkar, A.: Automatic Question Generation Approaches and Evaluation Techniques, Current Science, 113, 9, 2017, 1683-1691. https://doi.org/10.18520/cs/v113/i09/1683-1691

[dos Santos, 2018] dos Santos, V.: Concept maps construction using natural language processing to support studies selection, In Proceedings of the 33rd Annual ACM Symposium on Applied Computing. New York, NY, USA: Association for Computing Machinery, 2018, 926–927. https://doi.org/10.1145/3167132.3234663

[Droog-Hayes, 2017] Droog-Hayes, M.: The effect of poor coreference resolution on document understanding, European Summer School in Logic, Language and Information (ESSLLI) Student Session, 2017, 209–220.

[Elhoseiny, 2012] Elhoseiny, M., Elgammal, A.: English2MindMap: An Automated System for MindMap Generation from English Text, IEEE International Symposium on Multimedia, Irvine, CA, USA, 10-12 December, 2012, 323–331., doi: 10.1109/ISM.2012.103

[Gašpar, 2023] Gašpar, A., Grubišić, A., Šarić-Grgić, I.: Evaluation of a Rule-Based Approach to Automatic Factual Question Generation Using Syntactic and Semantic Analysis, Language Resources and Evaluation, 2023, https://doi.org/10.1007/s10579-023-09672-1

[Ghaddar, 2016] Ghaddar, A., Langlais, P.: WikiCoref: An English Coreference-annotated Corpus of Wikipedia Articles, In Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16). Portorož, Slovenia: European Language Resources Association (ELRA), 2016, 136–142.

[Griffis, 2016] Griffis, D., Shivade, C., Fosler-Lussier, E., Lai A.M.: A Quantitative and Qualitative Evaluation of Sentence Boundary Detection for the Clinical Domain, AMIA Joint Summits on Translational Science Proceedings, 2016, 88–97.

[Grubišić, 2020] Grubišić, A., Stankov, S., Žitko, B., Šarić-Grgić, I., Gašpar, A., Tomaš, S., Brajković, E., Vasić, D.: Declarative Knowledge Extraction in the AC&NL Tutor, In R. A. Sottilare & J. Schwarz (Eds.), Adaptive Instructional Systems. Cham: Springer International Publishing, 2020, 293–310. https://doi.org/10.1007/978-3-030-50788-6_22

[Grubišić, 2022] Grubišić, A., Žitko, B., Gašpar, A., Vasić, D., Dodaj, A.: Evaluation of split-and-rephrase output of the knowledge extraction tool in the intelligent tutoring system, Expert Systems with Applications, 187, 2022, 115900. https://doi.org/10.1016/ j.eswa.2021.115900

[Horsmann, 2015] Horsmann, T., Erbs, N., Zesch, T.: Fast or Accurate? – A Comparative Evaluation of PoS Tagging Models, The International Conference of the German Society for Computational Linguistics and Language Technology (GSCL 2015), Sep 30 – Oct 2, 2015, University of Duisburg-Essen, Germany, https://doi.org/10.17185/ duepublico/72101

[Jacobsen, 2021] Jacobsen, M., Sørensen, M. H., Derczynski, L.: Optimal Size-Performance Tradeoffs: Weighing PoS Tagger Models, arXiv, 2021, https://doi.org/10.48550/arXiv. 2104.07951

[Jia, 2018] Jia, Y., Qi, Y., Shang, H., Jiang, R., Li, A.: A Practical Approach to Constructing a Knowledge Graph for Cybersecurity, Engineering, 4, 1, 2018, 53–60. https://doi.org/10.1016/ j.eng.2018.01.004

[Jordan, 2001] Jordan, P. W., Rose, C., Vanlehn, K.: Tools for Authoring Tutorial Dialogue Knowledge, In: Moore, J.D., Redfield, C.L., Johnson, W.L. (eds.) Artificial Intelligence in Education: AI-ED in the Wired and Wireless Future, Proceedings of AI-ED 2001, IOS Press, Amsterdam, 222–233.

[Kowata, 2010] Kowata, J. H., Cury, D., Claudia, M., Boeres, S.: Concept Maps core elements candidates recognition from text, In Proceedings of the Fourth International Conference on Concept Mapping. Viña del Mar, Chile, 2010, 120–127.

[Kurdi, 2020] Kurdi, G., Leo, J., Parsia, B., Sattler, U., Al-Emari, S.: A Systematic Review of Automatic Question Generation for Educational Purposes, International Journal of Artificial Intelligence in Education, 30, 1, 2020, 121–204. https://doi.org/10.1007/s40593-019-00186-y

[Li, 2022] Li, J., Sun, A., Han, J., Li, C.: A Survey on Deep Learning for Named Entity Recognition, IEEE Transactions on Knowledge and Data Engineering, 34, 1, 2022, 50–70. https://doi.org/10.1109/TKDE.2020.2981314

[Lu, 2021] Lu, P., Poesio, M.: Coreference Resolution for the Biomedical Domain: A Survey, In Proceedings of the Fourth Workshop on Computational Models of Reference, Anaphora and Coreference. Punta Cana, Dominican Republic: Association for Computational Linguistics, 2021, 12–23, https://doi.org/10.18653/v1/2021.crac-1.2

[Manning, 1999] Manning, C. D., Schütze, H.: Foundations of Statistical Natural Language Processing, Cambridge, MA, USA: MIT Press (1999).

[Manning, 2011] Manning, C.D.: Part-of-Speech Tagging from 97% to 100%: Is It Time for Some Linguistics?, In: Gelbukh, A.F. (eds) Computational Linguistics and Intelligent Text Processing (CICLing), Lecture Notes in Computer Science, vol 6608. Springer, Berlin, Heidelberg, 2011, https://doi.org/10.1007/978-3-642-19400-9_14

[Manning, 2014] Manning, C., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S., McClosky, D.: The Stanford CoreNLP Natural Language Processing Toolkit, In Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations. Baltimore, Maryland: Association for Computational Linguistics, 2014, 55–60. https://doi.org/10.3115/v1/ P14-5010

[Mitri, 2022] Mitri, M.: Story Analysis Using Natural Language Processing and Interactive Dashboards, Journal of Computer Information Systems, 62:2, 2022, 216-226, DOI: 10.1080/ 08874417.2020.1774442

[Nazaruka, 2020] Nazaruka, E., Osis, J., Griberman, V.: Using Stanford CoreNLP Capabilities for Semantic Information Extraction from Textual Descriptions, In: Damiani, E., Spanoudakis, G., Maciaszek, L. (eds) Evaluation of Novel Approaches to Software Engineering (ENASE), Communications in Computer and Information Science, vol 1172. Springer, Cham., 2019, https://doi.org/10.1007/978-3-030-40223-5_1.

[Naseer, 2021] Naseer, S., Ghafoor, M. M., Alvi, S. bin K., Kiran, A., Shafique Ur Rahmand, G. M., Murtaza, G.: Named Entity Recognition (NER) in NLP Techniques, Tools Accuracy and Performance, Pakistan Journal of Multidisciplinary Research, 2, 2, 2021, 293–308.

[Novak, 2008] Novak, J. D., Cañas, A. J.: The Theory Underlying Concept Maps and How to Construct and Use Them, Technical Report. Institute for Human and Machine Cognition, Florida, 2008.

[Panchal, 2021] Panchal, P., Thakkar, J., Pillai, V., Patil, S.: Automatic Question Generation and Evaluation, Journal of University of Shanghai for Science and Technology, 23, 2021, 751–761. https://doi.org/10.51201/JUSST/21/05203

[Qasim, 2013] Qasim, I., Jeong, J.-W., Heu, J.-U., Lee, D.-H.: Concept map construction from text documents using affinity propagation, Journal of Information Science, 39, 2013, 719–736. https://doi.org/10.1177/0165551513494645

[Rus, 2013] Rus, V., D'Mello, S., Hu, X., Graesser, A.: Recent Advances in Conversational Intelligent Tutoring Systems, AI Magazine, 34, 3, 2013, 42–54, https://doi.org/10.1609/aimag.v34i3.2485

[Rus, 2008] Rus, V., McCarthy, P. M., McNamara, D. S., Graesser, A. C.: A study of textual entailment, International Journal on Artificial Intelligence Tools, 17, 04, 2008, 659–685. https://doi.org/10.1142/S0218213008004096

[Shao, 2020] Shao, Z., Li, Y., Wang, X., Zhao, X., Guo, Y.: Research on a new automatic generation algorithm of concept map based on text analysis and association rules mining, Journal of Ambient Intelligence and Humanized Computing, 11, 2, 2020, 539–551. https://doi.org/10.1007/ s12652-018-0934-9

[Shimorina, 2022] Shimorina, A., Heinecke, J., Herledan, F.: Knowledge Extraction From Texts Based on Wikidata, In Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Industry Track. Hybrid: Seattle, Washington + Online: Association for Computational Linguistics, 2022, 297–304. https://doi.org/10.18653/v1/2022.naacl-industry.33

[Smith, 2006] Smith, A. E., Humphreys, M. S.: Evaluation of unsupervised semantic mapping of natural language with Leximancer concept mapping, Behavior Research Methods, 38, 2, 2006, 262–279. https://doi.org/10.3758/BF03192778

[Susarla, 2003] Susarla, S., Adcock, A., Van Eck, R., Moreno, K. and Graesser, A. C. (2003). Development and evaluation of a lesson authoring tool for AutoTutor, In V. Aleven, U. Hoppe, J. Kay, R. Mizoguchi, H. Pain, F. Verdejo, & K. Yacef (Eds.), AIED2003 Supplemental Proceedings, Sydney, Australia: University of Sydney School of Information Technologies, 2003, 378-387

[Villalon, 2008] Villalon, J. J., Calvo, R. A.: Concept Map Mining: A Definition and a Framework for Its Evaluation, In Proceedings of the 2008 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology - Volume 03. Sydney, Australia: IEEE Computer Society, 2008, 357–360. https://doi.org/10.1109/WIIAT.2008.387

[Wafula, 2016] Wafula, B. N.: Automatic construction of concept maps, master's thesis, University of Eastern Finland, Faculty of Science and Forestry, Joensuu School of Computing, 2016, retrieved on 25 April 2023 from https://erepo.uef.fi/handle/123456789/16854

[Zhang, 2020] Zhang, M.: A survey of syntactic-semantic parsing based on constituent and dependency structures, Science China Technological Sciences, 63, 10, 2020, 1898–1920. https://doi.org/10.1007/s11431-020-1666-4

[Zouaq, 2007] Zouaq, A., Nkambou, R., Frasson, C.: Building Domain Ontologies from Text for Educational Purposes, In E. Duval, R. Klamma & M. Wolpers (Eds.), Creating New Learning Experiences on a Global Scale. Springer Berlin Heidelberg, 2007, 393–407.

[Zouaq, 2009] Zouaq, A., Nkambou, R.: Evaluating the Generation of Domain Ontologies in the Knowledge Puzzle Project, IEEE Transactions on Knowledge and Data Engineering, 21, 11, 2009, 1559–1572, https://doi.org/10.1109/TKDE.2009.25

[Zubrinic, 2012] Zubrinic, K., Kalpic, D., Milicevic, M.: The Automatic Creation of Concept Maps from Documents Written Using Morphologically Rich Languages, Expert Systems with Applications, 39, 16, 2012, 12709–12718. https://doi.org/10.1016/ j.eswa.2012.04.065