Automatic Crossword Clues Extraction for Language Learning

Santiago Berruti **Arturo Collazo** **Diego Sellanes**

Aiala Rosá **Luis Chiruzzo**

Instituto de Computación, Facultad de Ingeniería, Universidad de la República
Montevideo, Uruguay

Abstract

Crosswords are a powerful tool that could be used in educational contexts, but they are not that easy to build. In this work, we present experiments on automatically extracting clues from simple texts that could be used to create crosswords, with the aim of using them in the context of teaching English at the beginner level. We present a series of heuristic patterns based on NLP tools for extracting clues, and use them to create a set of 2209 clues from a collection of 400 simple texts. Human annotators labeled the clues, and this dataset is used to evaluate the performance of our heuristics, and also to create a classifier that predicts if an extracted clue is correct. Our best classifier achieves an accuracy of 84%.

1 Introduction

This paper presents a series of experiments on automatically extracting clues from a text, that could be used to generate a crossword puzzle. Crosswords are a very interesting tool that can be used in educational contexts, in particular for developing vocabulary (Orawiwatnakul, 2013). In this work, we will focus on extracting words and generating definitions for crosswords in the context of teaching English as a foreign language, in particular for students at the beginner level.

A crossword (see Fig. 1) is a type of puzzle where words are arranged horizontally or vertically, and often intersect each other. The puzzle is presented with blank spaces where the letters should be, and is accompanied by the set of definitions of the target words. In our case, these words and definitions will be related to a text, for example an article or story that an English teacher wants to work with in class. The crossword in the figure could be obtained by processing the following article[1]:

(1) *Mexico is part of the continent North America. Mexico is shaped like a hook with a wide top. (...) On its west side is the Pacific Ocean. (...) A peninsula is a piece of land that has water on most sides. The Yucatan Peninsula has the Gulf of Mexico on its west and north sides. It has the Caribbean Sea on its east side.*

In order to do this, we must detect a set of interesting words from the text, extract their corresponding definitions, and create the crossword puzzle. In this work, we are not focusing on building the actual puzzle grid, but on extracting appropriate clues from the text that could be used to populate the crossword.

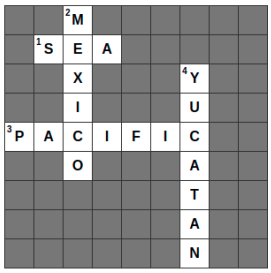


Figure 1: Possible crossword with clues extracted from example 1.

This kind of crosswords could be used as reading comprehension exercises, so it is expected that a student reads the text first, and then tries to solve the associated crossword. Notice that in this situation, the types of definitions we are trying to extract will generally be tied to the accompanying text, and would not exactly be dictionary definitions.

Throughout the text we will use the term

[1]Abridged version of the article "Where Is Mexico?" from ReadWorks.

Proceedings of the 19th Workshop on Innovative Use of NLP for Building Educational Applications, pages 381–390

"definiendum" for a word that could appear in a crossword, and *"definition"* for a short phrase that defines that word. Likewise, when we mention a *"clue"*, we are referring to a <definiendum, definition> pair in this context.

In this project, we created a series of heuristics for extracting clues from simple texts, stories and articles. We used the heuristics to create a small annotated dataset of <definiendum, definition> pairs, labeled according to how correct they are to be used in a crossword and how grammatical they are. With this dataset, we trained several machine learning systems that try to predict if new clues would be suitable for creating a crossword.

The main contributions of this work are the following: 1) We present a set of heuristics that can extract clues from simple English texts. The heuristics range from simple linguistic patterns extraction to more complex question-answer generation, and could also combine information from different sentences in a text. 2) We annotated a dataset of 2209 clues, generated using our heuristics, with information about grammaticality and correctness as a clue for a crossword (i.e. it would be suitable to include this definition for this definiendum in the context of a crossword)[2]. 3) We did experiments on automatic classification of clues, with the best classifier achieving 84% accuracy and 78% macro-F1 for detecting correct clues.

The rest of the paper is structured as follows: Section 2 presents some relevant related work, section 3 describes the corpus we used and the heuristics we created for extracting clues, section 4 shows the quality evaluation of the extracted clues and presents the classifier we built, and finally section 5 presents some conclusions and future work.

2 Related Work

The works on automatic generation of crossword clues from texts are scarce. We comment below those that are closer to our objectives.

In (Percovich et al., 2019), the authors present two approaches to the generation of crossword puzzles, with the aim of using them for teaching English as a second language at the beginner level. On the one hand, a set of definitions organized by classical categories (e.g., colors, food, animals) is created, from which crossword puzzles are automatically generated according to the selected category. The definitions are extracted from different sources: existing children's dictionaries were used, and new definitions are also generated by applying patterns on Simple Wikipedia texts and filtering those that do not correspond to the expected categories by applying heuristics based on word embeddings. On the other hand, crossword puzzles are generated from texts entered by teachers, from which pairs <definiendum, definition> are extracted automatically, as in our work. For this, some heuristics based on information from a dependency parser are applied, using the verb "to be" as a central element, and each clue is extracted from a single sentence.

In (Rigutini et al., 2012), a traditional approach based on linguistic analysis tools is presented. A pipeline is applied to generate crossword clues from texts obtained by web crawling. The system processes the texts by applying different analyzers in sequence: sentence splitter, POS-tagger, chunker, and specific rules for the identification of subject, object and predicate (verbal or nominal) of each sentence. Then a finite state automaton is applied to detect which sentences are definitions, and finally, to generate the crossword clue, the subject of the sentence is removed. The system was used to create Italian crosswords.

In (Esteche et al., 2017), a system for crossword generation from Spanish news texts is presented. They use tools for linguistic analysis –a POS-tagger, a constituency parser, and a clause segmenter– and from the information they provide they define recursive regular expressions to extract clues from the texts. The paper presents a wide variety of patterns, and includes a tool implemented in Prolog to generate different crossword grids. This last task of actually generating the crossword grid has been explored in the past (Meehan and Gray, 1997; Botea, 2007), and is not particularly relevant from an NLP perspective, although some of the ideas in (Esteche et al., 2017) such as using different priority levels for words when building the grid might be relevant in an educational context to make sure the words that the teacher wants to highlight are included in the puzzle.

In (Katinskaia et al., 2018), a platform for language learning is presented. The platform includes crossword-based exercises created from stories. The crosswords are composed of words taken from the story, the student has to guess the words in

[2]The corpus is not yet available because we are waiting for the authorization of the text owners to publish it, we hope this will happen soon.

their correct grammatical form.

Some of our extraction heuristics that use linguistic patterns bear some resemblance to the classic method proposed by Hearst (1992) in the context of hyponyms extraction, although in that work there is an iterative step in which previously extracted information is used to generate new patterns from a large corpus. We have not tried the iterative process in this work, although a similar approach has already been explored in the context of clues extraction for crosswords in the past (Esteche et al., 2017), where it was unable to find new productive patterns.

A similar task to the one addressed in this paper is the generation of Question & Answering exercises for English teaching, aiming at the same objective, which is to evaluate the comprehension of a text. The extraction of question/answer pairs from texts can be used as input to generate clues for crossword puzzles, by means of some transformations, as we show below. In (Yao et al., 2022; Berger et al., 2022) neural approaches for generating Q&A exercises for teaching are presented, in (Morón et al., 2021) a similar work is presented using patterns based on different linguistic analyses (POS-tagging, semantic role labeling, coreference resolution, named entities recognition).

Another related NLP task is definition extraction, although with important differences from the problem addressed in this paper. Our goal is to extract clues for creating crosswords from texts. These clues may not make any sense outside the context of that text since they are not true definitions of the terms, in the strict sense of a dictionary definition. An important reference on definition extraction is SemEval-2020 shared task 6, "Definition extraction from free text with the DEFT corpus" (Spala et al., 2020), in which a specific corpus for definition extraction was used to train models. Fifty-one teams participated in this competition and most of them based their approaches on the use of pretrained language models.

3 Dataset and Clue Extraction

We created a number of heuristic rules or patterns that can be used to extract <definiendum, definition> pairs from simple texts in English. These rules were created by experimenting with a corpus of short texts, manually exploring and analyzing the frequencies of different expressions and patterns.

We used a dataset comprised of 400 short texts obtained from the ReadWorks website[3], an educational technology nonprofit organization. ReadWorks contains thousands of short texts and stories ranked in levels K, 1, 2, 3, 4, and 5 and categorized in the Lexile scale. The texts are written by experts and curated by educators, and could be non-fiction, fiction or poetry, within these three thematic areas: science, social studies and art. In our experiments we used 400 texts, most of them belong to level 1, and a few to level 2. These texts include short articles about history, geography and science, and some short stories.

Our clue detection and extraction rules begin with a pre-processing phase in which we perform coreference resolution using the AllenNLP tool (Gardner et al., 2018) and simple sentence splitting. Then we have a series of modules that apply different extraction patterns based on: syntax, Semantic Role Labeling (SRL), extended patterns that combine sentences, Named Entity Recognition (NER), and Question-Answering (QA). All these patterns extract rough clues, and we use a post-processing module to improve the shape of the definienda and definitions.

3.1 Syntax-based patterns

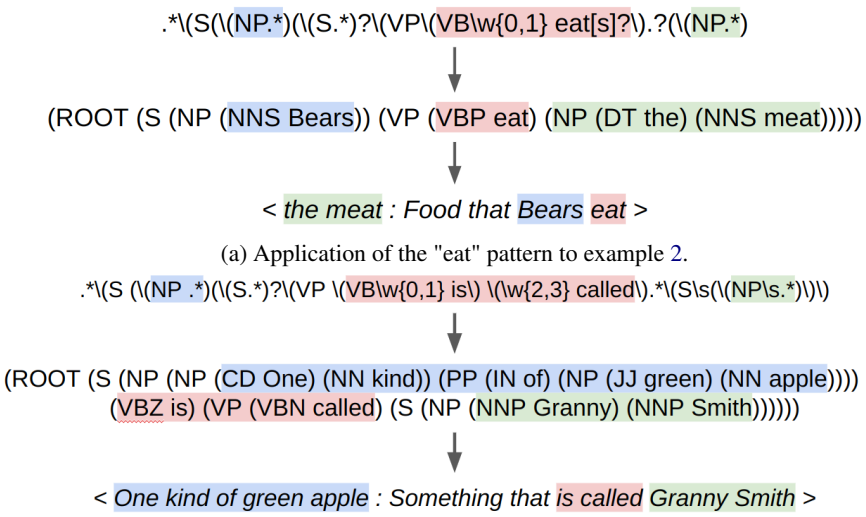
The first heuristic processes the constituency tree looking for some key verbs, and performs basic transformations to turn the phrase into a clue. We first analyzed our dataset searching for the most common verbs, trying to find ways in which the sentences these verbs took part in could be transformed into clues. Consider the following examples:

(2) *Bears eat the meat*

(3) *One kind of green apple is called Granny Smith.*

These examples use two frequent constructions in the corpus: the verb ‘to eat’ and the construction ‘is called’. We crafted regular expressions for these frequent verbs that could be used to extract <*definiendum, definition*> pairs. These expressions operate over the text representation of the constituency tree, obtained using AllenNLP (Gardner et al., 2018), and use capture groups to define the parts of the text we want to extract. The patterns created for these verbs are shown in Fig. 2.

[3]https://www.readworks.org/



(a) Application of the "eat" pattern to example 2.

(b) Application of the "is called" pattern to example 3.

Figure 2: Examples of application of syntactic patterns: a carefully tailored regular expression is applied to the textual representation of the constituency tree, and the capture groups are used to build the clue.

Note that it is actually not possible to use regular expressions to capture any type of expression from a constituency tree, but the type of simple sentences existing in the corpus, with low nesting levels, could mostly be treated with this tool.

As shown in Fig. 2, the types of clues extracted in this way could be rough around the edges, but the post-processing phase intends to fix some of these imperfections.

3.2 SRL-based patterns

Semantic role labels are a way of categorizing parts of a sentence as arguments of a predicate and the role they play in the described action (Palmer et al., 2005). The use of semantic role labels provides a more expressive way to define patterns that could capture some subtleties that regular expressions over constituency trees cannot. Semantic roles are, in a way, invariant to the syntactic position in the sentence, e.g. an argument with the role 'agent' could be acting as a subject or an object but still have the same semantic role.

We used the AllenNLP SRL analyzer and defined patterns that could be applied to these structures. In the SRL patterns, we look for combinations of phrases with role agent or theme (ARG0/ARG1) and phrases with role theme or attribute (ARG1/ARG2) associated to the same predicate. Several patterns were composed in this way, that work over the verbs like "to be", "to have", and "to like".

In the "to like" pattern, the analyzer already disambiguates the uses of "like" as a verb (predicate) or as a preposition, so the following examples are correctly resolved:

(4) *Bobby **likes** to play basketball.*

(5) *Bobby plays sports **like** basketball.*

The pattern for the verb "to live" is slightly different, because instead of an ARG1 it generally defined an ARGM which can either be a temporal or a location modifier. See the following examples:

(6) *Aztecs lived in Mexico.*

(7) *Dinosaurs lived in prehistoric ages.*

In example 6 the argument is labeled as `ARG-LOC`, so the clue is extracted as <*Mexico : Place where Aztecs lived*>, while in example 7 it is labeled as `ARG-TMP` and the clue is <*Prehistoric ages : Time when dinosaurs lived*>.

Besides looking for particular verbs, we built a more generic SRL pattern that captures any verb given that some valid combination of arguments is found. Optionally this pattern can also take some other types of modifiers, like in the following example:

(8) *A grown-up kangaroo can be bigger than a person.*

From example 8, the generic pattern can extract the clue <*kangaroo : something that can be bigger than a person*>, that includes the modifier "can" which is labeled as `ARGM-MOD` by the SRL module.

3.3 Extended patterns

The patterns seen so far work within the boundaries of one sentence, but we can create richer definitions if we combine the contents of more sentences. Consider the following example:

(9) *Bears are apex predators. They eat small mammals, like foxes.*

Two separate patterns (for the verbs "be" and "eat") could be applied independently, and if coreferences have been properly resolved, they both would have the same definiendum "Bears". In this kind of situations, where there is a nominative pattern ("be" or "is called") and a pattern that describes an action (such as "eat" or "live"), we can combine the definitions of the two patterns to create a new clue: <*Bears : apex predators that eat small mammals, like foxes*>.

3.4 NER-based patterns

In the NER patterns, we use the spaCy NER module (Honnibal et al., 2020) to find the named entities of the text and their categories. One pattern that already could capture the use of named entities was the "live" pattern, but in this case it is generalized when a named entity with a particular category is found. Take a look at the following examples:

(10) *Many chili peppers are grown in Mexico.*

(11) *Lebron James plays basketball at NBA.*

We have two cases with named entities of different categories. We can extract both names as definiendums, and use the category to create a definition tailored to that named entity. Mexico is classified as `GPE` (geo-political entity), so its clue would be <*Mexico : Place where many chili peppers are grown*>. Lebron James is classified as `PER` (person), so its clue ends up being <*Lebron James : Organization where Lebron James plays basketball*>.

3.5 QA-based patterns

Another way of generating clues is by casting the problem as a question answering task. First we use the NER module to extract named entities from the text that could be candidates to be used as definienda, together with some features like the category and number. Using the story as context, we create a question related to the named entity, and use the HuggingFace QA module[4] to generate an answer.

For example, using as context a story about the Great Sphynx, the process could detect the terms "Egypt", "Ancient Egyptians" and "Africa" as candidates. The following are the questions the process creates for those candidates, and the answers given by HugginFace QA:

- *What is Egypt?* || *A country in Africa.*
- *What are Ancient Egyptians?* || *They made the statue by cutting into a huge rock.*
- *What is Africa?* || *Egypt is a country.*

The first answer is a definition that fits perfectly, creating the clue <*Egypt : A country in Africa*>, but the other two are not correct in this context. In order to improve the quality of the clues, we filter out answers with a low confidence score as predicted by HuggingFace QA.

3.6 Post-processing

As mentioned above, some of the clues extracted might not be directly usable in crosswords, but we have a post-processing phase that can transform some of them to make them better. Consider the following examples of clues extracted with the patterns:

1. <*A snake : a reptile that moves its tail*>
2. <*Beauty and charm : Features Cleopatra had*>
3. <*Solar system : the name for the sun, planets, and other smaller bodies*>
4. <*Statue of Liberty : a symbol of freedom*>

The first step of post-processing is using NLTK POS tagger (Bird, 2006). There are different transformations that could be done after identifying the POS tags. If the definiendum is a determiner and a noun as in the first case, we can just drop the determiner. When a coordination is found, as in the second case, we can forget the conjunction and create two different clues with the remaining words. In other cases we select one or more words from the definiendum, giving preference to nouns, and move the rest of the words to the definition (third case). In the fourth case, as both words are good definienda, we create two different clues, keeping the remaining words in the definition.

After this process, the transformed clues are the following:

[4] https://huggingface.co/tasks/question-answering

1. <*snake : a reptile that moves its tail*>
2. <*beauty : Feature Cleopatra had*>
3. <*charm : Feature Cleopatra had*>
4. <*system : (Solar ___) the name for the sun, planets, and other smaller bodies*>
5. <*Statue : (___ of Liberty) a symbol of freedom*>
6. <*Liberty : (Statue of ___) a symbol of freedom*>

Sometimes the patterns tend to return underspecified definitions, like "something that is called Granny Smith" or "something that can be bigger than a person". These definitions are not fit for a crossword as they describe the terms too vaguely. As described above, the NER based pattern uses the named entity category to specify this, indicating whether the referred term is a person, location, organization, etc., while the SRL pattern can sometimes infer a more specific term for location or temporal modifiers.

However, this information is not available in all cases, so we implemented a heuristic based on the WordNet ontology (Miller, 1995; Fellbaum, 1998) that tries to improve this. WordNet is a lexical database that contains thousands of terms taxonomically structured by the hypernymy/hyponymy relation. The idea is to replace the vague term for a category that is still hypernymy of the definiendum, but is simple enough for students at the beginner level. In our case, we considered a list of simple categories that are generally part of the beginner level curricula: *animal, food, fruit, clothing, city, country, region, location, instrument, plant, tool, activity, action, relative, feeling, sensation*.

The heuristic tries to visit all the hypernyms of a definiendum and stops when one that belongs to our simple category list is found, otherwise, if we reach the most abstract "entity" term and no suitable candidate was found, we keep the first term of the definition as "something".

4 Experiments

After running our heuristics methods on all 400 texts of our dataset, our process generated 2321 <definiendum, definition> pairs. However, the quality of these clues might be very variable, depending on the pattern and on the text they were extracted from. It is very important to analyze which clues were correct and could be used for crosswords, and also we are very interested in making the whole process more accurate. One way to do this would be having a classifier that could discriminate if a new clue was correct or not according to some criteria. In this section, we describe the annotation of our corpus and the classifier we built.

4.1 Annotation

First of all, we annotated manually all the generated clues. Eleven annotators participated in this process[5], and they were asked to answer two questions for each clue: First, if the clue could be considered correct in the context of a crossword, considering that the person solving the puzzle would have read the corresponding text. Secondly, if the clue is grammatically correct.

After an initial annotation round that was used to discuss criteria and labeling conventions, we noticed that there was a third dimension we wanted to address. There were cases in which the original text had mistakes (probably transcription errors) that made the extracted clues unusable, these cases were to be marked as invalid and would be left out of the final corpus.

Each annotator was given a spreadsheet with the following information:

- **Definiendum:** Word to guess in the crossword.
- **Definition:** Text that defines the definiendum.
- **Context:** Main sentence were the clue was extracted from.
- **Text Name:** Name of the original text, so it could be checked for further context.
- **Method:** Heuristic pattern used to generate the clue.

They would also have the full texts that were needed for understanding their clues. The annotators had to indicate if the clue was valid or invalid (due to errors in the text), if it was correct (for a crossword), and if it was grammatical.

4.2 Analysis

All the 2321 clues originally extracted by the heuristics were considered for the annotation. In

[5]The annotators were researchers that took part in this work and a related project, and students that were compensated with undergraduate credits.

total, 112 of them were deemed as invalid because of errors in the texts, and were not considered for the rest of the analysis. The following is an example of a clue that is invalid, because the text contained a transcription error:

- **Context:** This painting Photos.comis titled Breaking Home Ties. It was painted by Thomas Hovenden, an Irish-born artist.
- **Definiendum:** Irish
- **Definition:** Breaking Home Ties

Some examples of clues that were considered correct and grammatical:

- **Context:** The White House has a swimming pool and a movie theater.
- **Definiendum:** Pool
- **Definition:** (Swimming ___) Something The White House has

- **Context:** So Franklin D. Roosevelt came up with plans to add more jobs.
- **Definiendum:** Roosevelt
- **Definition:** (Franklin D. ___) President that came with plans to add more jobs

The following is an example of a clue that could be considered correct, but is ungrammatical:

- **Context:** Green iguanas eat leaves, flowers, and fruit
- **Definiendum:** iguanas
- **Definition:** (Green ___) large that eat leaves, flowers, and fruit

The definition should be changed to something like "large animal that eats..." to be considered grammatical.

Table 1 shows the number of clues extracted by each pattern, and the corresponding values of correctness and grammaticality according to the annotators. The first thing we can notice is that some patterns are much more productive than others: all the SRL patterns were very productive, but especially the extended pattern that combines a sentence with the verb "to be" and another sentence generates a lot of clues, mainly because it could combine the already productive "to be" pattern with any other related clue. If we analyze the correctness and grammaticality of the clues, it is interesting to see that the SRL patterns once again are the most trustworthy: except the generic SRL pattern all the rest are very accurate in terms of grammaticality, and also mostly correct. On the other hand, the QA pattern was an under-performer, obtaining very few clues from the texts, and even then most of them were wrong, even if we set a confidence threshold for the generation. Exploring better and more powerful QA generation models (e.g. Yao et al. (2022); Berger et al. (2022)) would be necessary to improve this pattern.

Pattern	Total	Correct	Gramm.
`is called`	35	48%	86%
`eat`	53	83%	87%
`live`	15	13%	60%
`SRL have`	276	65%	90%
`SRL like`	54	65%	94%
`SRL live`	74	84%	88%
`SRL to be`	538	81%	96%
`SRL gen.`	217	75%	82%
`to be ext.`	887	70%	83%
`NER`	49	71%	86%
`QA`	11	28%	81%
Total	2209	72%	87%

Table 1: Number of clues extracted by each pattern from the whole dataset, together with their average correctness and grammaticality according to the manual annotation. The 112 invalid clues are not included in this table.

4.3 Classifier

Using this annotated set, we performed a series of experiments on creating a classifier that could automatically determine if a given clue is correct or not. For these experiments, we split the set in 80% for training and 20% test, and we used 5 fold cross validation for parameter tuning.

The different classification models we experimented with are the following:

Centroid distance baseline Based on the simple classifier presented in (Percovich et al., 2019), we obtain the FastText embeddings (Bojanowski et al., 2017) centroid of the context sentence, and of the definiendum and definition pair, and we calculate the Euclidian distance between them. Then we experimentally determined a distance threshold

that maximized the F1 metric.

Machine learning methods Using a representation that takes the FastText embeddings of the context, definiendum and definition, we experimented with several classical machine learning models (Kowsari et al., 2019): KNN, Naïve Bayes, Decision Trees, Gradient Boosting and MLP.

Deep learning methods We carried out experiments with BERT based models, inspired by (Yao et al., 2022), which used a similar model for ranking automatically generated question-answer pairs. For these experiments we used the BERT (Devlin et al., 2018) and DistilBERT (Sanh et al., 2019) pretrained models, finetuned to our data with the HuggingFace Transformers (Wolf et al., 2019) `AutoModelForSequenceClassification` class. In both cases, the input is the context, the definiendum and the definition, separated by `SEP` tokens.

Table 2 shows the results of these experiments, and we can see that both BERT-based models outperform the rest, DistilBERT being the best model for our task.

Model	Accuracy	Macro F1
Centroid	0.66	0.58
KNN	0.72	0.57
Dec. Tree	0.64	0.55
Grad Boosting	0.71	0.59
MLP	0.73	0.59
NB	0.66	0.49
BERT	0.77	0.70
DistilBERT	**0.84**	**0.78**

Table 2: Accuracy and Macro F1 of the classifiers that predict the correctness of a clue.

5 Conclusions

We presented some experiments on automatic extraction of clues for crosswords from simple texts, considering a clue as a <definiendum, definition> pair that could be used in a crossword puzzle. We created several heuristic patterns for detecting and extracting clues using different NLP tools, like constituency parsing, SRL, NER and QA. With these heuristics, we extracted 2209 clues from a dataset of 400 documents from ReadWorks, and annotated them with information about correctness and grammaticality. The best heuristic patterns for extracting clues, according to our annotation, are the ones based on SRL.

Using our annotated dataset, we trained several classifiers on the problem of detecting whether an extracted clue is correct for a crossword. The best model for this turned out to be a DistilBERT model finetuned on our training data, obtaining 84% accuracy and 78% macro F1.

In the future, we want to explore the possibility of using large language models such as GPT or LLAMA for this task, which have shown promising results according to some preliminary experiments. We also want to explore the possibility of improving the QA based pattern by using better QA extraction modules. Currently we are in the process of testing our extraction system integrated to a crossword generation tool in a real case (Chiruzzo et al., 2022), with school children that are beginning to learn English, which would give us a better sense about how well our heuristics work and how they can be improved.

6 Ethics Statement

We understand that by using pretrained statistical NLP tools, our work could be infusing undesired biases in the results. This is especially dangerous in the situation we want to use the system, which is the context of a classroom with school children. Because of this, we consider that the results obtained by this tool must not be used directly by the students, but the supervision of a teacher is always necessary. In the system we are building, a teacher can automatically extract clues from a text and create a crossword, but they always have the possibility to inspect the generated clues in order to modify or remove any term or definition that might not be suitable, before the crossword is presented to students.

7 Limitations

In the experiments described in this paper, we have worked only with ReadWorks stories. These are texts designed to be simple and easy to read, and intend to be varied in terms of contents, but nonetheless they are only one data source and this means our process might end up be too tailored to the style and vocabulary of these texts and not generalize well to other sources.

Furthermore, given that these texts are very short, during our experiments we found that our heuristics generally can extract very few clues

from each text. On average we can extract around four <definiendum, definition> pairs from a text (this can be noticed in the numbers presented in Table 1), which might be too few for creating an engaging crossword. We are working on improving the extraction process to generate more clues, but a combination with other methods such as including dictionary definitions of related words, especially short ones that could fill crossword gaps, would be advisable to build more complete and interesting crosswords.

Besides our heuristic patterns, we made some experiments to extract clues with more modern large language models (LLM) techniques which seemed promising. However, due to the limitations of our application servers, we decided to use more traditional methods because they are less demanding in terms of computational resources.

Acknowledgements

We want to thank ReadWorks for letting us use their simple English texts in the context of this research.

References

Gonzalo Berger, Tatiana Rischewski, Luis Chiruzzo, and Aiala Rosá. 2022. Generation of english question answer exercises from texts using transformers based models. In *2022 IEEE Latin American Conference on Computational Intelligence (LA-CCI)*, pages 1–5. IEEE.

Steven Bird. 2006. NLTK: the natural language toolkit. In *Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions*, pages 69–72.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the association for computational linguistics*, 5:135–146.

Adi Botea. 2007. Crossword grid composition with a hierarchical csp encoding. In *Proceeding of the 6th CP Workshop on Constraint Modelling and Reformulation, ModRef-07*.

Luis Chiruzzo, Laura Musto, Santiago Góngora, Brian Carpenter, Juan Filevich, and Aiala Rosá. 2022. Using nlp to support english teaching in rural schools. In *Proceedings of the Second Workshop on NLP for Positive Impact (NLP4PI)*, pages 113–121.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Jennifer Esteche, Romina Romero, Luis Chiruzzo, and Aiala Rosá. 2017. Automatic definition extraction and crossword generation from spanish news text. *CLEI Electronic Journal*, 20(2):6–1.

Christiane Fellbaum. 1998. *WordNet: An electronic lexical database*. MIT press.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke Zettlemoyer. 2018. AllenNLP: A deep semantic natural language processing platform. In *Proceedings of Workshop for NLP Open Source Software (NLP-OSS)*, pages 1–6, Melbourne, Australia. Association for Computational Linguistics.

Marti A Hearst. 1992. Automatic acquisition of hyponyms from large text corpora. In *COLING 1992 volume 2: The 14th international conference on computational linguistics*.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, Adriane Boyd, et al. 2020. spaCy: Industrial-strength natural language processing in python.

Anisia Katinskaia, Javad Nouri, and Roman Yangarber. 2018. Revita: a Language-learning Platform at the Intersection of ITS and CALL. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Kamran Kowsari, Kiana Jafari Meimandi, Mojtaba Heidarysafa, Sanjana Mendu, Laura Barnes, and Donald Brown. 2019. Text classification algorithms: A survey. *Information*, 10(4):150.

Gary Meehan and Peter Gray. 1997. Constructing crossword grids: Use of heuristics vs constraints. *Proceedings of Expert Systems*, 97:159–174.

George A Miller. 1995. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41.

Martín Morón, Joaquín Scocozza, Luis Chiruzzo, and Aiala Rosá. 2021. A tool for automatic question generation for teaching english to beginner students. In *2021 40th International Conference of the Chilean Computer Science Society (SCCC)*, pages 1–5. IEEE.

Wiwat Orawiwatnakul. 2013. Crossword puzzles as a learning tool for vocabulary development. *Electronic Journal of Research in Education Psychology*, 11(30):413–428.

Martha Palmer, Daniel Gildea, and Paul Kingsbury. 2005. The proposition bank: An annotated corpus of semantic roles. *Computational linguistics*, 31(1):71–106.

Analía Percovich, Alejandro Tosi, Luis Chiruzzo, and Aiala Rosá. 2019. Ludic applications for language teaching support using natural language processing. In *2019 38th International Conference of the Chilean Computer Science Society (SCCC)*, pages 1–7. IEEE.

Leonardo Rigutini, Michelangelo Diligenti, Marco Maggini, and Marco Gori. 2012. Automatic generation of crossword puzzles. *Int. J. Artif. Intell. Tools*, 21.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Sasha Spala, Nicholas Miller, Franck Dernoncourt, and Carl Dockhorn. 2020. SemEval-2020 task 6: Definition extraction from free text with the DEFT corpus. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 336–345, Barcelona (online). International Committee for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Bingsheng Yao, Dakuo Wang, Tongshuang Wu, Zheng Zhang, Toby Li, Mo Yu, and Ying Xu. 2022. It is AI's turn to ask humans a question: Question-answer pair generation for children's story books. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 731–744, Dublin, Ireland. Association for Computational Linguistics.