A Tool for Automatic Question Generation for Teaching English to Beginner Students

1st Martín Morón
Instituto de Computación
Facultad de Ingeniería
Universidad de la República
Montevideo, Uruguay
martin.moron@fing.edu.uy

2nd Joaquín Scocozza
Instituto de Computación
Facultad de Ingeniería
Universidad de la República
Montevideo, Uruguay
joaquin.scocozza@fing.edu.uy

3rd Luis Chiruzzo
Instituto de Computación
Facultad de Ingeniería
Universidad de la República
Montevideo, Uruguay
luischir@fing.edu.uy

4th Aiala Rosá
Instituto de Computación
Facultad de Ingeniería
Universidad de la República
Montevideo, Uruguay
aialar@fing.edu.uy

***Abstract*—We present a tool for the automatic generation of questions from texts for ESL teaching, using resources and methods from the Natural Language Processing (NLP) field. The approach presented in this paper is based on symbolic methods, using hand-crafted rules. The tool uses linguistic information, such as semantic roles, co-references and named entities, to generate questions and answers from a text selected by a teacher. The generated questions are ranked in order to offer the teacher a varied set to create an exercise. All the elements can be edited in case any error is produced, since NLP tools and methods are not completely accurate. When solving an exercise, the students are given immediate feedback based on the expected answers. A primary evaluation showed promising results, especially for "what" and "who" questions.**

***Index Terms*—NLP for Language Teaching, Automatic Question Generation, Computer Assisted Language Learning**

I. Introduction

The development of applications to support teaching, besides providing an innovative and motivating resource for working with students in class, expands the possibilities of teaching in some contexts where there are not enough teachers, such as elementary schools in rural areas that require remote support from specialists in different disciplines. In these moments, moreover, this type of tools are crucial to facilitate remote work not only in the particular contexts mentioned above, since in many parts of the world face to face classes have been suspended due to the health emergency we are experiencing.

Natural Language Processing (NLP) provides different resources that can facilitate the development of these tools, when it comes to teaching languages or disciplines that require reading and comprehension of texts. This work is part of a research line that seeks to apply NLP techniques for the development of didactic applications for teaching English as a second language (ESL). The work has been oriented mainly to aid school teachers who teach English to children at beginner level in rural areas, supported remotely by specialized English teachers.

In this paper we present a prototype for automatic questions generation which allows the teacher to enter a text of their choice, from which different questions are generated. The questions are ranked in order to offer the teacher a varied set that conforms an exercise. For each question an answer is pre-calculated, so that the tool can give immediate feedback to the students. The generated questions and answers can be edited by the teacher, in case any error is produced.

The rest of the paper is structured as follows: section II describes the related work, section III gives an overview of the tool we built and its workflow, section IV provides details on how the different question/answer pairs are created and ranked, section V describes our evaluation of the generated questions, and finally VI shows some conclusions and future work.

II. Related Work

In the area of Automatic Question Generation (AQG), the traditional approach has been the definition of templates and rules, to be applied on sentences or texts pre-processed with linguistic tools [1]–[3]. Most of the work has focused on generating wh-questions from simple sentences, aiming at the evaluation of text comprehension. Some authors have researched the generation of questions from sentences with more complex structures [4] and questions aiming at the assessment of grammatical concepts [5]. In recent years, the availability of datasets for training machine learning models, mainly the SQuAD corpus [6], [7], allowed the experimentation with neural networks [8]–[11], in some cases explicitly including the expected answer, in addition to the source text, as part of the input for question generation.

Neural approaches were not explored in this work, since the texts of the SQuAD corpus are quite different from those that might be used in teaching English to children at beginner level, and we did not found an available dataset adequate for our work. The approach we present is based on manual rules for a predefined set of question types. The rules are applied on texts pre-processed with tools that provide linguistic information as semantic roles, co-references, named entity recognition and classification, and morpho-syntactic information.

978-1-6654-0956-8/21/$31.00 ©2021 IEEE

III. Description of the Tool

The tool lets the teachers create exercises comprised of a series of questions associated to a text. Once the teacher inputs a text, the tool automatically generates a set of question/answer pairs and selects some subset of the questions. The teacher has the possibility of adding more questions or answers, editing or removing questions or answers, or saving the exercise (with a name). The students can then log in to the application, select an exercise created by a teacher, and solve it. When students are solving exercises, they receive immediate feedback about their answers. Section IV-D describes how this feedback is generated. Fig. 1 shows an example of a normal use case scenario for our tool: the teacher inputs a text and selects some questions and answers for creating an exercise, and then the student solves the exercise and receives feedback.

IV. Generation of Questions and Answers

The question generation algorithm has the following three steps, represented as dotted boxes in Fig. 2:

1) Pre-process the text using the AllenNLP [12] library performing Part-of-Speech (POS) and morpho-syntactic tagging, Semantic Role Labeling (SRL), Co-reference Resolution, and Named Entity Recognition (NER). The information produced by this linguistic analysis will be used by the rules.
2) Process the rule modules, each module generates one type of question using as input the information generated in the previous step. Both questions and answers are generated at the same time.
3) Apply the ranking function to the generated question set. The aim of this function is to give the user an ordered subset of questions that could be used as a starting point for generating the exercise.

A. Rule modules

There are five rule modules, which are described below.

1) 'What colour' questions: These could be seen as a subset of the 'what' questions that we will see later on, but they follow a different path in our tool. Each adjective in the sentence is checked against the WordNet [13] ontology to see if it is a hyponym of the synset *"colour"*. Then we analyze each verb in the sentence that has the colour in one of its arguments and the semantic role associated to that argument. A 'what colour' question can be created in one of several ways. There are three types of rules: rules for the verb *"to be"* (*"The ball is red"*), rules for the verb *"to have"* (*"John has brown eyes"*), and rules for other verbs (*"Mary washes her red car every Saturday"*).

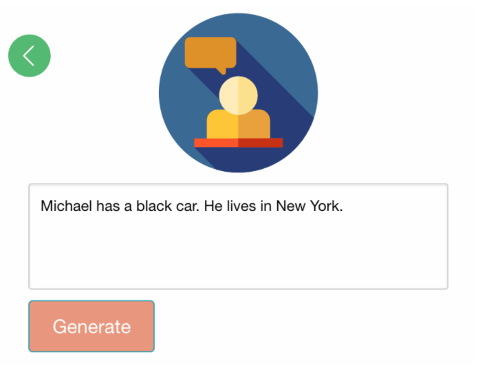
For example, for the sentence *"John has brown eyes"*, we have the following analysis:

$[_{ARG0}$ John] $[_{V}$ has] $[_{ARG1}$ brown$_{adj}$ eyes]

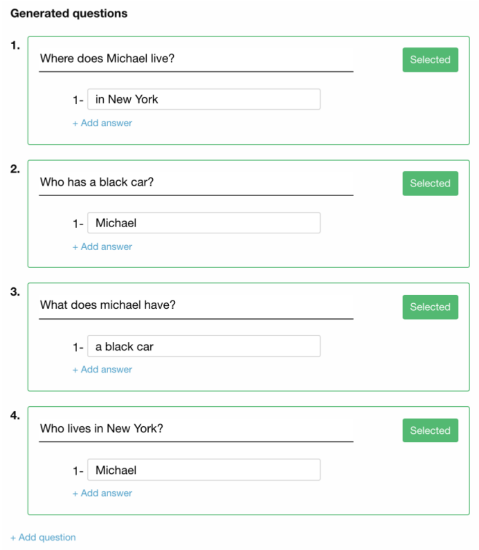
Which can be processed with a 'what colour' rule for the verb *"to have"*, that takes the ARG0, ARG1 and adjective from the sentence and generates the following question/answer pair:

Q: *"What colour are John's eyes?"*

A: *"brown"*


Michael has a black car. He lives in New York.

(a) The teacher inputs the text.


Generated questions

1. Where does Michael live? — Selected
1- in New York
+ Add answer

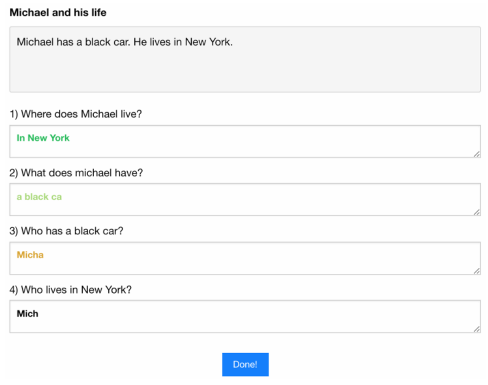
2. Who has a black car? — Selected
1- Michael
+ Add answer

3. What does michael have? — Selected
1- a black car
+ Add answer

4. Who lives in New York? — Selected
1- Michael
+ Add answer

+ Add question

(b) The application creates a series of questions and the teacher selects a set for the exercise.


Michael and his life

Michael has a black car. He lives in New York.

1) Where does Michael live?
In New York

2) What does michael have?
a black ca

3) Who has a black car?
Micha

4) Who lives in New York?
Mich

(c) The student completes the exercise and receives immediate feedback.

Fig. 1: Screenshots from the tool as seen by (a, b) the teacher and (c) the student.

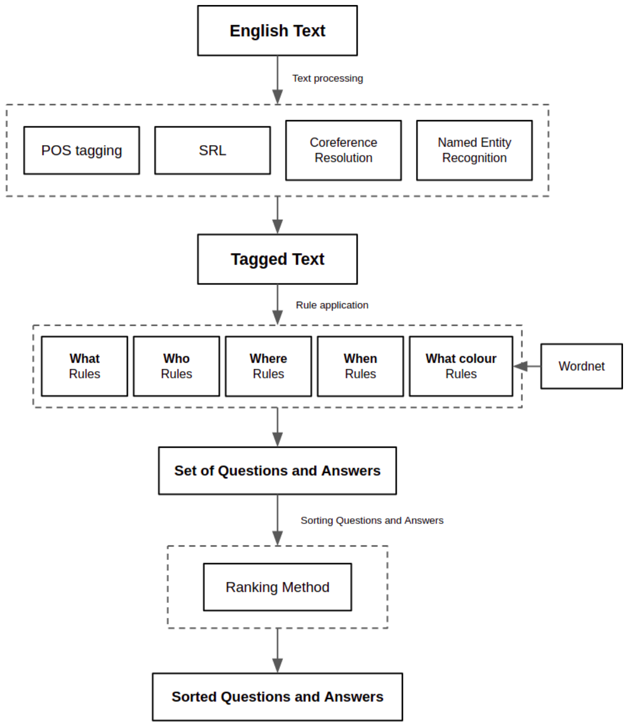


Fig. 2: Overview of the question-answer pairs extraction process.

```
set_of_questions = [ ]

for each verb in the sentence :

   if the verb has ARG0 or ARG1 then

      if ARG0 exists then ARG = ARG0
      else ARG = ARG1

      if ARG is a named entity then
         if ARG is PERSON then
            int_pronoun = "Who"
         else
            int_pronoun = "What"
      else
         if ARG is in {the..., it, that} then
            int_pronoun = "What"
         else if ARG is a subject personal pronoun then
            int_pronoun = "Who"

      question = int_pronoun + rest_of_sentence (without ARG)
      answer = ARG
      set_of_questions += (question, answer)

return set_of_questions
```

Fig. 3: Pseudocode for the ‘Who’ module.

2) ‘Who’ questions: This type of questions are some of the most common questions in level A1 and A2 tests. The rules for these questions analyze the SRL and NER structure of the sentence. Either ARG0 or ARG1 must be present in the sentence, and will be the answer of the generated question. If both exist, ARG0 is prioritized. The rules must decide which interrogative pronoun should be used: ”Who” or ”What”. ”Who” is used when the agent is a named entity classified as PERSON by the NER module, or when it is a personal pronoun.

For example, consider the sentence *“Mike has been sleeping for hours”*, which has the following analysis:

[$_{ARG0}$ Mike] has been [$_{V}$ sleeping] [$_{ARG-TMP}$ for hours]

With this sentence we can generate a ‘who’ question/answer pair like the following:

Q: *“Who has been sleeping for hours?”*

A: *“Mike”*

Fig. 3 shows a pseudocode for the rules in the ’Who’ questions module. Notice that this module can also generate some ‘What’ questions when the expected answer is not a person.

3) ‘What’ questions: This is the most heterogeneous group of questions. The rules look for the presence of ARG0 and ARG1 in the sentence, and take into account different aspects like the verb tense and the arguments for generating the questions. There are specific rules for different verb forms, specially verbs with auxiliaries, and there are rules for questions with the verb ”to do”.

For example, the sentence *“Mike will play footbal”* has the following analysis:

[$_{ARG0}$ Mike] [$_{ARGM-MOD}$ will] [$_{V}$ play] [$_{ARG1}$ football]

From this sentence, a rule can create a question/answer pair like the following:

Q: *“What will Mike play?”*

A: *“football’*

4) ‘When’ questions: The ‘when’ questions look for temporal adjuncts (ARGM-TMP) and require the sentence to have at least one of the ARG0 or ARG1 arguments. They also use information about verb tense and analyze different cases to generate the question template, deciding whether to use the auxiliary ”to do” or auxiliaries already present in the sentence. If the sentence includes other arguments or modifiers, they are included in the question as well.

For example, for the sentence *“Bill and Mary will play tennis next week”* we can have the following analysis:

[$_{ARG0}$ Bill and Mary][$_{ARGM-MOD}$ will] [$_{V}$ play][$_{ARG1}$ tennis][$_{ARGM-TMP}$ next week]

From this analysis, the rules generate the following question/answer pair:

Q: *“When will Bill and Mary play tennis?”*

A: *“next week”*

5) ‘Where’ questions: The ‘where’ questions are similar to the ‘when’ questions, but look for location adjuncts in the sentence. For example the analysis for the sentence *“Mary plays basketball at school”* would be:

$[_{ARG0}$ Mary$]$ $[_{V}$ plays$]$ $[_{ARG1}$ basketball$]$ $[_{ARGM-LOC}$ at school$]$

The rules could generate the following question/answer pair for this sentence:

Q: *"Where does Mary play basketball?"*

A: *"at school"*

B. Use of coreferences

The tool uses the coreference analysis to augment the number of questions it can generate. For example, if the text contains the pair of sentences *"Mike is going to school. He will play football."*, the coreference between *"He"* and *"Michael"* is used to generate the question/answer pair *"What will Mike play?/football"*

C. Questions ranking

The rules generate a large number of questions for each sentence in a document, producing some repetitive questions. We created a ranking system that randomly chooses a set of sentences for creating an exercise based on the document, considering the following factors: the questions are generated from different positions in the text (i.e. different sentences) and the questions are of as many different types as possible. The ranking algorithm first shuffles the list of questions and calculates a score for each question like shown in equation 1.

$$score = \frac{1}{tp + sp} \cdot \frac{1}{tt + st} \tag{1}$$

Where:

tp: Total number of questions generated for that position in the document.

sp: Number of questions for that position selected so far.

tt: Total number of questions of that type generated in the document.

st: Number of questions of that type selected so far.

This criteria tries to balance questions of varied types and from different locations in the original document. The user that is creating the exercise will be shown all the generated questions, but by default n questions chosen by the ranking algorithm will be selected.

D. Answers grading

When an exercise is being solved, students can receive immediate feedback from the tool that indicates how close to the real answers they really are. This is done by calculating the Levenshtein edition distance between the answer typed by the student and all possible answers for a question. In our case, when an answer consists of more than one word, word order is ignored and we take the minimum of the distances between each possible pair of words in the candidate and gold answers. It was implemented in this way after consulting with English teachers that are in charge of classes for beginners, as they value that the students can find the right words more than their syntactic skills. Furthermore, extra words that do not align to the expected answer are ignored if the correct words are found in the typed answer. The distance between the typed answer and the expected answer is shown using a colour code from orange to green.

TABLE I: Colour-coded feedback for some answers.

Answer	Distance
it is blue	4
it is r	2
it is re	1
it is red	0
red is it	0
red	0
He has a red ball	0

For example, assume we have a question that has three possible answers: *"red"*, *"is red"* and *"it is red"*. Table I shows some examples of typed answers with different distances and their feedback colours in the tool.

V. Evaluation

We carried on an initial evaluation of the question generation process. We analyzed a set of 20 simple English texts that could be used in the context of an English class for beginners, and manually annotated questions and answers that could be extracted from them focusing in the types of questions we developed. The initial corpus of 20 texts was separated in 80% for development and 20% for testing. In total we generated 271 question/answer pairs. The development set was used to analyze and improve the rules that generate questions, while the test set was held out and only used for evaluation purposes. We kept the same distribution of question types for both sets.

Table II shows the evaluation of questions generated using the rules. As can be seen in the table, it is much more likely to generate a 'what' or 'who' question than a 'where', 'when' or 'what colour' question. Although the numbers for the 'what' and 'who' questions are promising, the number of questions for the rest of the types in the test set is very low, so a larger corpus is necessary in order to do a better validation for these types. The last two columns in Table II show the evaluation of the answers generated by the rules. In the corpus, we annotated each question with one or more possible answers, so for the evaluation we consider an answer correct if it is any of the possible answers annotated for that question.

VI. Conclusions

We developed a prototype for Automatic Question Generation implementing a set of rules for wh-questions, using texts that can be used for teaching English as a second language to children at a beginner level. The tool lets teachers input a text, and generates a set of question/answer pairs ranked by a heuristic that tries to add diversity to the set. The students can open an exercise, answer the questions, and receive immediate feedback on their answers. We performed an initial evaluation of the questions and answers generation with encouraging results, especially for the "what" and "who" questions, although more research is needed in this respect.

In future work we will include new types of questions and explore Machine Learning techniques. For this purpose, we will use the prototype and its edition functionality to build

TABLE II: Results of the execution of the questions and answers generation process compared to the questions manually annotated in the corpus.

	Questions						Answers	
Type	**Expected**	**Generated**	**Incorrect**	**Precision**	**Recall**	**F-Score**	**Generated**	**Precision**
Who	47	48	8	0.833	0.851	0.842	40	0.75
What	33	34	6	0.824	0.848	0.834	28	0.6
Where	4	2	1	0.5	0.25	0.333	1	1
When	1	2	1	0.5	1	0.667	1	1
What colour	1	1	0	1	1	1	1	1

a corpus with texts and associated questions, selecting texts adequate for ESL at beginner level. Our working strategy prioritized having a rapid prototype that implements an initial set of rules, so that it could be used for the generation of a corpus of texts with associated questions. So the next steps involve creating questions for different texts using the prototype, and then correcting errors and adding extra questions through the edition functionality.

This expanded corpus could be used to perform machine learning experiments with the aim of improving the performance of the tool, or we could try to use some of the existing corpora such as SQuAD combined with our generated questions to adapt and improve the performance of machine learning systems.

References

[1] M. Heilman, "Automatic factual question generation from text," Ph.D. dissertation, Carnegie Mellon, 2011.

[2] R. Das, A. Ray, S. Mondal, and D. Das, "A rule based question generation framework to deal with simple and complex sentences," in *2016 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*. IEEE, 2016, pp. 542–548.

[3] N. Le, T. Kojiri, and N. Pinkwart, *Automatic Question Generation for Educational Applications – The State of Art*. Springer, 2014.

[4] P. Khullar, K. Rachna, M. Hase, and M. Shrivastava, "Automatic question generation using relative pronouns and adverbs," in *Proceedings of ACL 2018, Student Research Workshop*. Melbourne, Australia: Association for Computational Linguistics, Jul. 2018, pp. 153–158. [Online]. Available: https://www.aclweb.org/anthology/P18-3022

[5] M. Chinkina and D. Meurers, "Question generation for language learning: From ensuring texts are read to supporting learning," in *Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications*. Copenhagen, Denmark: Association for Computational Linguistics, Sep. 2017, pp. 334–344. [Online]. Available: https://www.aclweb.org/anthology/W17-5038

[6] P. Rajpurkar, R. Jia, and P. Liang, "Know what you don't know: Unanswerable questions for squad," 2018.

[7] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, "Squad: 100,000+ questions for machine comprehension of text," 2016.

[8] X. Du, J. Shao, and C. Cardie, "Learning to ask: Neural question generation for reading comprehension," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Vancouver, Canada: Association for Computational Linguistics, Jul. 2017, pp. 1342–1352. [Online]. Available: https://www.aclweb.org/anthology/P17-1123

[9] J. Li, Y. Gao, L. Bing, I. King, and M. R. Lyu, "Improving question generation with to the point context," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 3216–3226. [Online]. Available: https://www.aclweb.org/anthology/D19-1317

[10] W. Zhou, M. Zhang, and Y. Wu, "Multi-task learning with language modeling for question generation," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 3394–3399. [Online]. Available: https://www.aclweb.org/anthology/D19-1337

[11] Y.-H. Chan and Y.-C. Fan, "A recurrent BERT-based model for question generation," in *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*. Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 154–162. [Online]. Available: https://www.aclweb.org/anthology/D19-5821

[12] M. Gardner, J. Grus, M. Neumann, O. Tafjord, P. Dasigi, N. Liu, M. Peters, M. Schmitz, and L. Zettlemoyer, "Allennlp: A deep semantic natural language processing platform," *arXiv preprint arXiv:1803.07640*, 2018.

[13] G. A. Miller, "Wordnet: a lexical database for english," *Communications of the ACM*, vol. 38, no. 11, pp. 39–41, 1995.