KTH ROYAL INSTITUTE
OF TECHNOLOGY

Degree Project in Computer Science and Engineering

Second cycle, 30 credits

Automatic Distractor Generation for Spanish Reading Comprehension Questions

Using language models to generate wrong, but plausible answers for multiple choice questions in Spanish

JORGE SANTIAGO ROMAN AVILA

Stockholm, Sweden, 2023

Automatic Distractor Generation for Spanish Reading Comprehension Questions

Using language models to generate wrong, but plausible answers for multiple choice questions in Spanish

JORGE SANTIAGO ROMAN AVILA

Master's Programme, Machine Learning, 120 credits
Date: December 18, 2023

Supervisor: Johan Boye
Examiner: Olov Engwall
School of Electrical Engineering and Computer Science
Swedish title: Automatisk Generering av Distraktorer för Spanska Läsförståelsefrågor
Swedish subtitle: Användning av språkmodeller för att generera felaktiga men trovärdiga svar på flervalsfrågor på spanska

Abstract

A common evaluation method for students in the context of reading comprehension is the use of Multiple Choice Questions. A student must read a text and a question, and then choose the correct answer from a set of options, one of which one is the correct answer, and the other options are wrong. The wrong options are called distractors. Creating Multiple Choice Question exams is time-consuming, and a task that is open for automation. Distractor Generation is the task of generating wrong, but plausible options for Multiple Choice Questions. It is a task that can be addressed with Machine Learning and Large Language Models. As this task has been addressed in languages such as English, and Swedish, this work addresses the task for the Spanish language. This work achieves 3 objectives. The first one is the creation of a Multiple Choice Question dataset in Spanish with distractors, by manually tagging distractors from the dataset *SQuAD-es*. The newly created dataset with distractors is called *SQuAD-es-dist*. The second one is automatically generating distractors with machine learning methods. A BERT model is fine-tuned to generate distractors, and a GPT model is used through zero-shot learning to generate distractors. The third one is a human study on the generated distractors to evaluate the plausibility and usability of the distractors. Although both methods show to be effective, yet not perfect, at generating distractors, the GPT model shows better applicability and a higher capacity to confuse students in the task of Distractor Generation.

Keywords

Distractor Generation, Multiple Choice Questions, Reading Comprehension, Spanish Language, BERT, GPT

Sammanfattning

En vanlig utvärderingsmetod för studenter i samband med läsförståelse är användningen av flervalsfrågor. En elev måste läsa en text och en fråga, och sedan välja rätt svar från en uppsättning alternativ, varav ett är rätt svar och de andra alternativen är fel. De felaktiga alternativen kallas distraktorer. Att skapa prov med flervalsfrågor är tidskrävande och en uppgift som är öppen för automatisering. Distraktorgenerering är uppgiften att generera felaktiga, men rimliga alternativ för flervalsfrågor. Det är en uppgift som kan lösas med maskininlärning och stora språkmodeller. Eftersom denna uppgift har behandlats på språk som engelska och svenska, behandlar detta arbete uppgiften för det spanska språket. Detta arbete uppnår 3 mål. Den första är skapandet av ett dataset med flervalsfrågor på spanska med distraktorer, genom manuell taggning av distraktorer från datasetet *SQuAD-es*. Det nyskapade datasetet med distraktorer kallas *SQuAD-es-dist*. Den andra metoden är att automatiskt generera distraktorer med maskininlärningsmetoder. En BERT-modell finjusteras för att generera distraktorer, och en GPT-modell används genom zero-shot-inlärning för att generera distraktorer. Den tredje metoden är en mänsklig studie av de genererade distraktorerna för att utvärdera hur rimliga och användbara distraktorerna är. Även om båda metoderna visar sig vara effektiva, men inte perfekta, för att generera distraktorer, visar GPT-modellen bättre tillämpbarhet och en högre kapacitet att förvirra studenter i uppgiften att generera distraktorer.

Nyckelord

Distraktorgeneration, Flervalsfrågor, Läsförståelse, Spanska språket, BERT, GPT

Resumen

Para evaluar a alumnos en el contexto de comprensión de lectura se usan las preguntas de opción múltiple. El alumno debe leer un texto y una pregunta y, a continuación, elegir la respuesta correcta entre un conjunto de opciones, una de las cuales es la respuesta correcta y las demás opciones son incorrectas. Las opciones incorrectas se denominan distractores. La creación de exámenes con preguntas de opción múltiple requiere mucho tiempo, y es una tarea susceptible a la automatización. La Generación de Distractores es la tarea de generar opciones erróneas, pero plausibles, para Preguntas de Elección Múltiple. Es una tarea que puede abordarse con Aprendizaje Automático y Grandes Modelos de Lenguaje. Dado que esta tarea ha sido explorada en idiomas como el inglés, y el sueco, este trabajo aplica la tarea para el idioma español. Este trabajo alcanza 3 objetivos. El primero es la creación de un conjunto de datos de preguntas de respuesta múltiple en español con distractores, etiquetando manualmente los distractores del conjunto de datos *SQuAD-es*. El nuevo conjunto de datos con distractores se denomina *SQuAD-es-dist*. La segunda consiste en generar distractores automáticamente con métodos de aprendizaje automático. Se entrena y ajusta un modelo BERT para generar distractores y se utiliza un modelo GPT mediante ”zero-shot learning” para generar distractores. El tercero es un estudio humano sobre los distractores generados para evaluar la aplicabilidad y usabilidad de los distractores. Aunque ambos métodos muestran ser eficaces, pero no perfectos, en la generación de distractores, el modelo GPT muestra una mejor aplicabilidad y una mayor capacidad para confundir a los estudiantes en la tarea de Generación de Distractores.

Palabras claves

Generación de Distractores, Preguntas de Opción Múltiple, Comprensión de Lectura, Lenguaje Español, BERT, GPT

Acknowledgments

I would like to thank Johan Boye for having guiding me through the whole process of the thesis. To Dmytro Kalpakchi for answering my questions about Textinator. To my friends in Stockholm for distracting me when distracting was needed. To my friends back home for being a WhatsApp call away. To everyone that participated in the Question Answering survey, regardless of how annoying Google Forms are. Thanks to Alex and Jimena. Thanks to both my parents. Check!

Stockholm, December 2023
Jorge Santiago Roman Avila

Contents

List of Figures

List of Tables

List of acronyms and abbreviations

BERT	Bidirectional Encoder Representations from Transformers
DG	Distractor Generation
GPT	Generative Pre-trained Transformer
LLM	Large Language Model
MCQ	Multiple Choice Question
NLP	Natural Language Processing

Chapter 1

Introduction

This chapter gives an overview of the problem addressed in this thesis. Section 1.1 introduces the background of the thesis as an introduction to the problem. Section 1.2 presents the motivation to do this project. Then Section 1.3 formally explains the problem statement. Section 1.4 and Section 1.5 present the objectives and the limitations, respectively. Section 1.6 briefly describes some methods used, and Section 1.7 contains the general structure of the succeeding chapters.

1.1 Background

In this work, different methods of Distractor Generation (DG) for Spanish language Multiple Choice Questions (MCQs) are explored. MCQs are widely used to evaluate students on several subjects, such as mathematics, reading comprehension, and vocabulary. Some widely known tests that use MCQs are the GMAT, GRE, SAT and TOEFL which are used for university, graduate admissions and as second language certificates. Typically, an MCQ is composed of a problem or question (Q), a key/correct answer (A), and 1 or more distractors, options that are incorrect ($D_1, D_2, ..., D_N$). In the context of reading comprehension, a text/passage (T) accompanies the previous elements. Students who are evaluated using MCQs are expected to read the text/passage (T), the question (Q), and from a subset of items/options that include the correct answer and the distractors ($A, D_1, D_2, ..., D_N$), choose the correct answer (A) to demonstrate comprehension, knowledge and/or understanding of the problem.

DG is the task of generating the wrong answers that accompany the context

of an MCQ. The objective is to generate the incorrect options ($D_1, D_2, ..., D_N$) given a text, question, and answer (T, Q, A). DG is important as good distractors are needed in order to have high-quality tests and evaluations [1]. DG is a task that is included in several of the guidelines written by experts to ensure that test items better assess student learning [2]. Distractors must be always wrong, but remain plausible to distract uninformed students [1, 2].

As a brief example, suppose there is a student doing a Reading Comprehension test, and the student encounters a text on the Second World War, followed by a set of MCQs. One of the questions asks *What year was the invasion of Normandy?*, to which the correct answer is *1944*. Distractors are the other options, in most cases 3 other wrong options, that must try to fool the student. A set of good distractors can not be *France*, *June*, or *Franklin D. Roosevelt*, as a student would know that these are not the correct answer. Good distractors could be *1941*, *1942*, and *1943*.

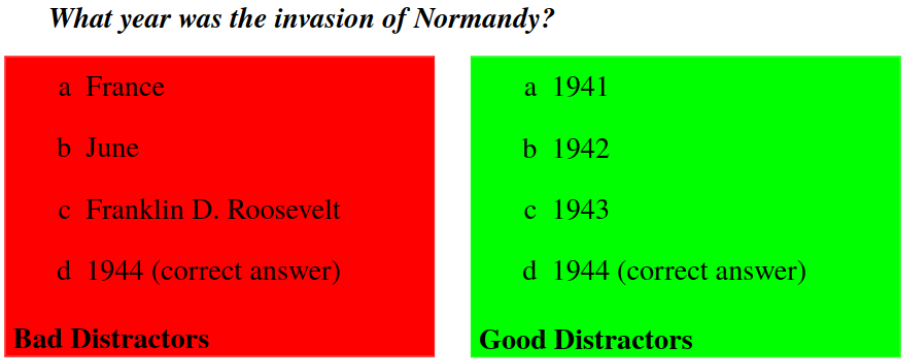

Figure 1.1: An example on good and bad distractors

Recent advances in the area of Natural Language Processing (NLP) have given rise to Large Language Models (LLMs). LLMs are able to handle complex language tasks such as text summarization, translation, text understanding, text synthesis, and question answering. Some examples of LLMs are Bidirectional Encoder Representations from Transformers (BERT) [3], T5 [4], and Generative Pre-trained Transformer (GPT) [5], with GPT-4 [6] being the most recent SoTA model. LLMs can be used and/or fine-tuned to address the current task of DG, to generate distractors automatically.

There exists currently work in Automatic DG for MCQs in languages such as English [7] and Swedish [8]. The latter one uses LLMs, specifically the Swedish BERT [9] model, to generate the distractors for MCQs. This work focused on a similar task, but using a MCQ dataset in Spanish [10] and a Spanish BERT [11] model. The methods presented in Kalpakchi and Boye [8] to generate distractors are the *left-to-right* and *u-PMLM* methods, which are used in this work and are explained in Section 2.5. Both these methods involve fine-tuning the BERT model by providing the context of the problem (T, Q, A) as input data, and the distractors $(D_1, D_2, ..., D_N)$ as output data. In this fashion, BERT can be fine-tuned to generate distractors.

This work also uses a GPT model ([5], [12], [13]) through the GPT API to generate distractors for the test set in a zero-shot fashion. The resulting distractors from GPT are included as final results of the experiments as well. The GPT model is presented with the context through a prompt, and is instructed in the prompt to generate the distractors.

This work presents a qualitative evaluation on the approaches, *left-to-right*, *u-PMLM*, and GPT. The results consist of a study performed by native Spanish speakers, that are instructed with answering the questions and to evaluate the plausibility and quality of distractors.

1.2 Motivation

This work intends to extend the work done previously in Kalpakchi and Boye [8] to evaluate the methods on a different dataset. It first plans to reproduce the methods and results in a Swedish dataset from Kalpakchi and Boye [8], and then repeat the methods on a Spanish dataset, that is built and adapted to the task from an existing Spanish MCQ dataset. The purpose is to expand these methods into different languages, and evaluate the generalization capability of these methods to other languages. Once this has been done, a new method will be proposed through a zero-shot fashion, using the GPT API.

From an applicability perspective, in a survey conducted by Merrimack College, Center [14], more than 1300 teachers from the US were asked to report the amount of time they spend a week in their jobs, with a consistent average of 54 hours a week, of which only 25 of them were dedicated to actual teaching. 5 hours were dedicated weekly for planning or preparation, and 3 hours for general administrative work. Exam creation could fall under the

planning or administrative work category, so automating the generation of distractors for MCQs is a potential tool for educators in order to gain more control of their time. As exam creation is just one of the many tasks teachers perform, this work is not intended to provide a solution to automate or displace teachers, but to provide an aiding tool. In the same survey of [14], 28% said that if they had more time, they would spend it in actual teaching time, which both benefits teachers and students.

From an NLP as a field perspective, this has tremendous value to research done in applications of LLMs, given the recent growth on interest for language modelling, and recent advances such as GPT-4. It is relevant to show how a smaller model is still able to perform in comparison to a bigger, newer model. It is important to note that at the time of start of the process of deciding the thesis topic, ChatGPT and its breakthrough had not happened yet. At the time of the development of the thesis, ChatGPT [15] and GPT-4 [6] models were made popular and available to use through an API, so the purpose is also to use the new approaches for this specific task.

1.3 Problem Statement and Research Question

To formally state the problem, creating MCQs is a time-consuming task for teachers and the problem is the task of automatically generating distractors for MCQs, which can be addressed with LLMs given their ability to generate language and solve specific language-related tasks.

The research question is: How effective are different language models, such as BERT and GPT, at automatically generating plausible distractors for MCQs in Spanish in the context of reading comprehension?

1.4 Objectives

The objective of this project is to explore current methods of DG and extend their applicability to a Spanish MCQ dataset. To the knowledge of the author, DG has been explored in the English language [7] and Swedish language [8], but DG has not been done before for the Spanish language. This has been divided into the following three sub-goals:

1. Modify/Adapt an existing MCQ dataset in Spanish to serve as a DG dataset, with all its contents correctly labelled $(T, Q, A, D_1, D_2, ..., D_N)$
2. Fine-tune and apply the chosen methodology for a Spanish BERT model and a GPT model for the task of DG.
3. Evaluate the generated distractors on Native Spanish speakers to analyze their usability as good distractors.

This work aims to finish with an analysis of the distractors in the qualitative study (Native Spanish speakers), as well as drawn conclusions between the Swedish results Kalpakchi and Boye [8], and the Spanish results obtained here.

There are some deliverables openly available, mainly this written work, and the Spanish dataset built. The BERT methods are the ones used in Kalpakchi and Boye [8].

1.5 Scope and Limitations

This work limits itself to the methods previously mentioned with a Spanish BERT (*left-to-right* and *u-PMLM*), and GPT with prompts in a zero-shot fashion. The *left-to-right* and *u-PMLM* methods are later described in Section 2.5. When using BERT, this study does not evaluate the different BERT models that have been pre-trained in Spanish, and it uses the previously chosen BETO model [11].

A big part of the project is in the data collection part. Manual tagging of questions, answers, and distractors are a crucial part of this project, as it is the input in order to fine-tune the models and serves as ground truth for the data points presented. Since tagging is a human intensive activity, it remains a limitation as to the size and quantity of data points that can be obtained from a Spanish dataset. Nevertheless, the aim is to have a dataset of similar size as done in Kalpakchi and Boye [8].

Another limitation regards the dataset and the qualitative evaluation on Spanish Native speakers. It is very important to clarify that correct answers are **found inside the context text**. This work will limit itself to just having questions in which the answers are explicitly found in the text, and will not have questions in which external knowledge or critical thinking abilities are needed to answer the text. The only tool needed to answer the question will

be the text.

Given the lack of resources, the qualitative evaluation is performed on voluntary native speakers, and the number of participants in the study is a limitation, given the time and resources available. Also, only student evaluations are performed. Teacher evaluation on distractor quality, in which a Spanish-speaking teacher analyzes the applicability of distractors from the teacher's point of view, is out of the scope of this work.

1.6 Methods

For data tagging and collection, the online tool Textinator [16] is used, which allows a user to create a dataset by highlighting parts of a text to mark a correct answer and distractors.

To fine-tune BERT, the repository associated with Kalpakchi and Boye [8] is used, and a HuggingFace [17] Spanish BERT model (BETO [11]). To use GPT in a zero-shot fashion, the python API from OpenAI is used.

1.7 Structure of the thesis

Chapter 2 presents relevant background information about NLP, LLMs, and DG. The methodology is divided into 2 chapters. Chapter 3 presents the methodology and method involved in recollecting the dataset, and also the process behind tagging the data to adapt it to distractor generation. Chapter 4 focuses on the methodology in fine-tuning BERT, generating distractors with GPT, and the human evaluation of the distractors. Chapter 5 is about the results on distractor generation from the quantitative and qualitative perspective. Chapter 6 discusses the results and make an analysis of the findings. And last but not least, Chapter 7 concludes the work, and proposes future work in the topic. References and Appendices are found in the end.

Chapter 2

Background

This chapter provides a theoretical background needed to comprehend the topic thesis. The first 2 sections, Section 2.1 and Section 2.2 explain a bit of the theory behind the language field of reading comprehension and MCQs as a teaching tool, respectively, in order to comprehend the area of application and the problem. Then Section 2.3 gives a background on the technical aspects of Language modelling to comprehend Transformers and Attention, the building blocks of GPT and BERT. Section 2.4 goes through available MCQ datasets and their characteristics, and finally section 2.5 presents related work on automatic distractor generation in other contexts.

2.1 Reading Comprehension

Reading Comprehension is the ability to interpret and understand written language as text. Different levels of reading comprehension have been studied and there exist several frameworks to measure it.

One of them is *A Simple View of Reading* by Gough and Tunmer [18]. Gough and Tunmer [18] identify 2 core components of reading that determine your reading comprehension proficiency: decoding and linguistic comprehension. Decoding, put in simple terms, refers to the word recognition abilities of a person. It is the ability to recognize the phonetic and visual representations of words and differentiate them between other words. In reading, it is the ability to read a word accurately [18]. Linguistic comprehension is the ability to make meaning out of a sequence of words, and associate them to a knowledge base in order to understand language [18]. If I am told a story, and I understand what I am being told, that is linguistic

comprehension.

To understand the *Simple View of Reading*, suppose you are learning Japanese and you are first taught to identify the symbols in the language and match them to their phonetic equivalent. You then learn to differentiate between words, based on the phonetic and visual representation of them, and you eventually can read whole pages of text in Japanese, and read them out loud accurately. With this, you gain high decoding abilities of Japanese. But you still lack the knowledge to know what the words mean, and what the meaning of the words are in a sequence. You are able to read Japanese, but you do not know what you are reading, so you lack the linguistic comprehension of Japanese. In order to have full reading comprehension of Japanese, you would need to learn the meaning of these words, besides being able to decode them and recognize them.

Decoding and linguistic comprehension are processes that can be evaluated on students individually. But to be strong in reading comprehension, one must be strong in both of these components. This framework uses a formula to determine the level of reading comprehension, in which a decoding score times a linguistic comprehension score gives the score for reading comprehension. The score for reading comprehension explains how well a student is at reading comprehension.

$$\textbf{Decoding} \times \textbf{Language Comprehension} = \textbf{Reading Comprehension} \quad (2.1)$$

Similarly, another framework for reading comprehension is the *Four Resources Model* from Firkins [19]. The *Four Resources Model* establishes four levels that act as language learner roles. This model was developed in order to aid programs designed to teach languages to students, and applies as a tool for second language instruction.

The *Four Resources Model* is a simple way of assigning different hierarchical roles in reading comprehension, and associate the skills a student must practice in these roles to improve literacy development. Each learner role allows a student to focus on the corresponding skills. The hierarchy of the four roles are highlighted in Figure 2.1. The four roles are [19]:

1. Code-Breaker: *How do I access the semiotic system of construction?* This role focuses on decoding, emphasis on sounds, vocabulary and

grammar.

2. Text Participant: *How do I understand this text?* Focus on making meaning of something, choseion genre, and register.

3. Text User: *How do I use this text?* This role focuses on action, emphasis on communicative purpose.

4. Text Analyst: *How is the text positioning me as a user?* Develop skills in understanding how to apply a text, relationship between texts and evaluate effectiveness of texts.

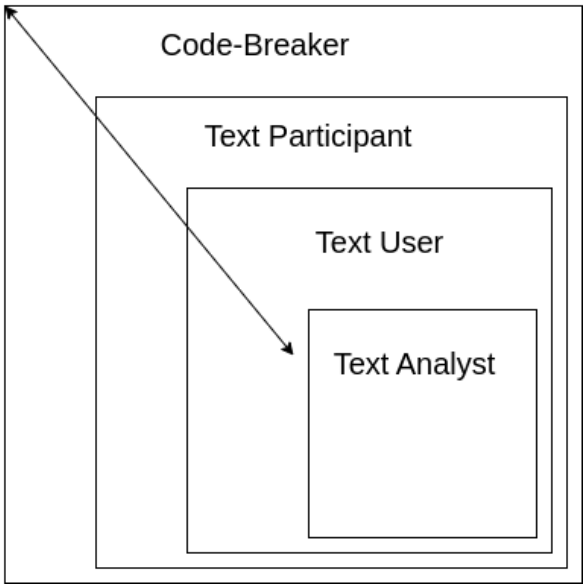

Figure 2.1: A sketch from the Four Resources Model, following Freebody and Luke (1990), found in Firkins [19]

Evaluating language abilities should then cover the different roles in this model. Duke and Pearson [20] identify several ways to test and evaluate reading comprehension in students through different effective individual strategies. Some strategies involve assigning students with the task of summarizing a text, predicting what will happen in a text, or making a visual representation of the text by drawing a picture or a diagram [20]. Another technique is simply having students read a text passage and then ask questions to the students about these text passages, and ensure they are able to answer the questions. MCQs are useful in this last strategy.

2.2 MCQs as a Teaching Tool

Asking questions about a text is an effective practice to test reading comprehension in students, and MCQs have been the default tool for this.

To the knowledge of the author, the most extensive studies on how to create multiple choice items can be attributed to Haladyna, Downing, and Rodriguez [2]. By building on their previous research, examining textbook passages and evaluating results of empirical studies with guidelines, Haladyna, Downing, and Rodriguez [2] have collected and created their own guidelines for multiple choice item creation. They created a taxonomy that contains 31 guidelines to aid on creating MCQs. The guidelines are divided into different categories.

- Concent Concerns
- Formatting Concerns
- Style Concerns
- Writing the Stem
- Writing the Choice

In category ”Writing the Stem”, the authors refer to the question (Q) and/or instructions as the ”Stem”.

The full guidelines from Haladyna, Downing, and Rodriguez [2] are found in Appendix A. Haladyna, Downing, and Rodriguez [2] performed evaluation on their guidelines based on endorsements from other authors and from doing an extensive literature study, to see which of their guidelines were cited in other works. Based on the literature and external sources found, Haladyna, Downing, and Rodriguez [2] reported a percentage of texts that were *Uncited*, *For* or *Against* their guidelines. This way, the authors evaluated the level of agreement between their proposed guidelines and the guidelines found in external literature. The objective of this was to say that their guidelines were not a ground truth, but simply tools that could or could not be used for exam creation.

For this report, the category of interest is *Writing the Choice* category, as it concerns writing the distractors. There are 14 guidelines under *Writing the Choice*, and some guidelines on distractors address the number of distractors, and the plausibility of distractors. The definition of *plausible* in distractor

context is not provided. This work also contains descriptions on the different formats, including conventional multiple choice (choose 1), true or false, multiple true-false, matching and context-dependent item set. Context-dependent item sets are the format in which there exists a scenario, table, chart, reading passage, or visual material, followed by a MCQ. MCQs for reading comprehension, are context-dependent item sets. [2].

Tarrant and Ware [1] provide test development practices in the context of nursing students, and collect guidelines from researched literature to increase the quality of MCQ exams. One specific technique regards the creation of plausible distractors. Tarrant and Ware [1] describe that distractors must be effective, and should center on widespread errors around the correct answer, meaning that a misconception or confusion of a characteristic of the correct answer, might lead an unprepared student to choose one of the distractors [1]. When creating distractors, a common mistake is to come up with 2 good distractors and then create ”filler” options, which the authors refer to as bad distractors, but are put there to ”fill” spots in the multiple choice options. ”Filler” options just make the set of multiple choice options have more items, at the cost of having bad quality items. Tarrant and Ware [1] suggest that it is better to have an MCQ instance with 2 high quality distractors, than having an instance with 4 items, but only 2 high quality distractors and 2 ”filler” options. Quality over quantity.

Some strategies to make distractors more plausible are to use the students' mistakes and misunderstandings to create items and to use verbal associations with the correct answer and the context (the question or the problem) [1]. The distractors should be homogenous and parallel, which can be translated as semantically similar and grammatically consistent (if asking about medical treatments, include only medical treatments). Good distractors should also be similar in length to the correct answer. As a general goal, Tarrant and Ware [1] state that distractors should be attractive to the uninformed, but distractors should not trick, confuse or mislead the participants who are knowledgeable.

Other techniques mentioned in these practices are to write a sufficient number of items and to distribute correct answers randomly and evenly in the ordering of the options [1].

In Vyas and Supe [21], the authors explore the effect of changing the number of options in the context of MCQs in medical tests in India. Tests

are usually composed of 4 to 5 options, including the correct option, and their study identifies that using 3 options is as effective as 4 and 5. In this study, they also identify writing plausible distractors as time-consuming and the most difficult part of preparing MCQs. The main drawbacks in creating distractors are implausibility, more than one correct answers, the longest option being the correct one, and also the faulty use of ”all of the above” and ”none of the above” as options. [21]

The importance of generating good distractors is backed up by these approaches on defining effective testing by using MCQs. Generating good distractors is challenging and a problem that can be addressed through computation and linguistics.

2.3 Natural Language Processing & Transformers

NLP is a multidisciplinary field that combines computer science, machine learning and linguistics, that addresses language problems by modelling and enabling computers to understand, and generate human language.

This section introduces the area of language modelling, to understand the basic concept of NLP, and some of the technical background that needed to understand the models used in this work, such as Transformers, GPT, and BERT. Attention and Tranformers are the main building blocks of the most recent language models. Understanding how these components and language models function is relevant to the thesis and comprehension of this work.

2.3.1 Language Modelling

NLP tasks are solved with language models. Language models define a probability distribution over sequences of words, and they predict the next words based on these probabilities [22, 23]. The model receives a certain sequence of words, and the model's task is to predict the words that follow the input sequence. This is true for tasks such as language generation, text summarization, question answering and translation. As an example, in the context of text completion as a NLP task, a language model must finish sequences and predict the next words of a sequence, based on the previous part of the sequence. It is more likely that a language model would complete

the phrase *"I want to break ..."* with the word *"free"*, rather than with the words *"orange"* or *"August"*.

From a probability perspective, a language model builds a probability distribution over all the possible next words in a sequence conditioned on the context sequence, and chooses the word with the highest probability as the next word. In the example below, a language model would find the probability of the word *"free"*, given the context *"I want to break"*.

$$P(\text{free}|\text{I want to break}) \tag{2.2}$$

Language is temporal in nature. Spoken and written text can be seen as a sequence of data points, that follow a temporal relationship, in which the order of the sequence is important [23]. Language models can be simple statistical models like N-gram models and Hidden Markov Models [22, 23], and can also be more complex models such as neural networks. For this work, we will focus on Transformers [24], which use an attention-based mechanism. The original Transformer architecture was designed as a sequence-to-sequence model. The first concept to grasp is sequence-to-sequence models.

2.3.2 Sequence-to-Sequence

Sequence-to-sequence models are composed of an encoder and a decoder, hence they are often also called **encoder-decoder** models [23]. The encoder takes as an input a variable-length sequence $\{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N\}$, and learns to output a fixed-length latent representation. Generating the fixed-length latent representation at the end of the sequence is the whole objective of the encoder, as it has comprised all the information of the input sequence [23]. The latent representation is often called the *context* $\mathbf{c}$, and is produced from the sequence of hidden vectors $\{\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_N\}$.

The decoder takes the context vector $\mathbf{c}$, and learns to output a variable-length sequence $\{\mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_T\}$ [25]. The decoder generates every token in the output sequence in an auto-regressive manner, one element at a time, having the last output $\mathbf{y}_{t-1}$ and the last hidden state $\mathbf{h}_{t-1}$ as input for every subsequent time step t, until an end-of-sequence token is output [23]. At every step, the full context $\mathbf{c}$ from the encoder can be made available as well.

Figure 2.2 shows a detailed process of an encoder-decoder model. The encoder just ensures to create the context $\mathbf{c}$, which in a vanilla encoder-decoder,

it is the last hidden state $\mathbf{h}_N$ of the encoder. Then at every step of the decoder, we use the original context from the encoder $\mathbf{c}$, the output of the decoder at previous time step $\mathbf{y}_{t-1}$ and the hidden state at previous time step $\mathbf{h}_{t-1}$ to generate the next token $\mathbf{y}_t$.

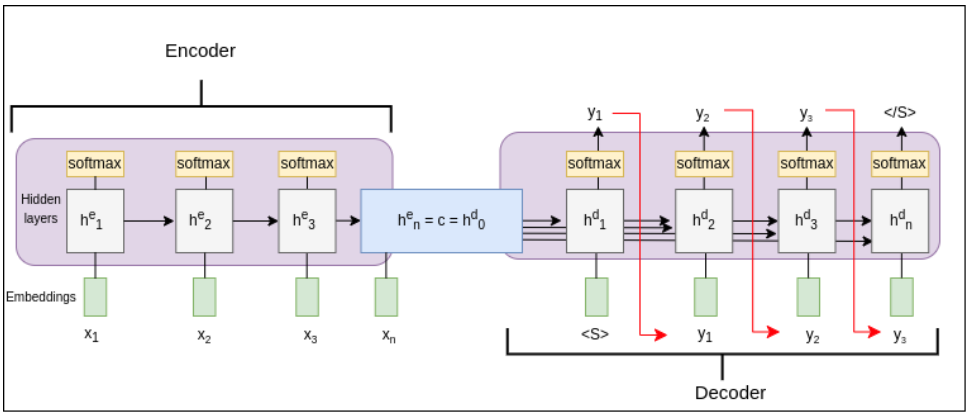

Figure 2.2: Detailed process of a vanilla encoder-decoder model. Following the model from Jurafsky and Martin [23]

Encoder-Decoder models are used for NLP tasks such as language translation, text summarization, and question answering. For instance, translating the English phrase "I like cheese", into Spanish "Me gusta el queso", means the input has a length of 3 words, while its translation to Spanish has a length of 4 words.

Under the field of statistical learning, these models are under transductive learning models. Transduction is any task that involves transforming an input sequence into a different output sequence [26]. In the context of NLP, language translation i.e. translating a text from English to Spanish, would be an example of transduction.

2.3.3 Limitations of Sequence-to-Sequence models

Traditionally, NLP tasks were performed by training sequence-to-sequence models that initially showed some limitations:

1. Long-range dependencies: The information in the network's hidden representation was kept for a number of cycles that was dependent on the

weights and the data that was input in future time steps. By the end of the encoding sequence, the context had compressed all the information of the sequence, which inevitably led to information loss.

2. Parallelization: Vanilla encoder-decoder models were sequential in nature, as every element in the input sequence generated a latent state that was needed in the next time step. This created a computational bottleneck, as training needed to be sequential, and was difficult to optimize in parallel computation hardware such as GPUs. The number of sequential operations of a forward pass increases linearly as the size of the sequence increases. [24]

3. Vanishing & Exploding Gradients: During training, as the gradients travel back in time, the gradients might suffer large increase of its norm (exploding gradients) or going to norm 0 (vanishing gradients) [27], making them hard to train networks for long sequences.

To counteract these limitations, attention mechanisms and later Transformer networks were developed.

2.3.4 Attention

In a vanilla encoder-decoder model, the context vector contains all the information about the encoded sequence. This may lead to having less information about the beginning of the sequence, compared to the information at the end of the sequence. This misrepresentation of the whole input sequence acts as a bottleneck, as the decoder only receives information that is available in the context. [23, 28].

Bahdanau, Cho, and Bengio [28] introduced a solution for the long-range dependencies' loss, as an *align and translate* scheme. Instead of having a fixed-length context, the context is now a sequence of vectors, which the decoder can "choose" adaptively in every time step to generate the appropriate word. It "aligns" to the part of the context that is most relevant, and then "translates".

In Bahdanau, Cho, and Bengio [28], the encoder-decoder model generates a forward sequence of hidden states $\overrightarrow{\mathbf{h}}_1, ..., \overrightarrow{\mathbf{h}}_N$, and a backward sequence of hidden states $\overleftarrow{\mathbf{h}}_1, ..., \overleftarrow{\mathbf{h}}_N$. Then every hidden state is a concatenation of the forward and backward hidden state such that for every step n, we have a hidden

state $\mathbf{h}_n = [\overrightarrow{\mathbf{h}_n}, \overleftarrow{\mathbf{h}_n}]$[28]. The motivation for this is so that at every step, we have information concerning to the left and to the right of the current word. The decoder uses in every step t a time-step specific context vector $\mathbf{c}_t$. This vector is computed by the decoder when generating output $\mathbf{y}_t$.

First the decoder computes an *attention score*, which in Bahdanau, Cho, and Bengio [28] is referred to as the output of an *alignment model* $a()$. The *alignment model* computes a vector $\mathbf{e}_{tn}$ based on the last decoding latent state $\mathbf{h}_{t-1}$, and every n hidden state of the encoder $\mathbf{h}_n$. The score is just a measure of how well all the N hidden states align with the output at position t. In Bahdanau, Cho, and Bengio [28], the *alignment model* is a trainable feed forward network, but it can be a simple similarity score such as a dot-product [23].

$$\mathbf{e}_{tn} = a(\mathbf{h}_{t-1}, \mathbf{h}_n) \tag{2.3}$$

The attention scores are normalized by a softmax function.

$$\alpha_{tn} = softmax(\mathbf{e}_{tn}) \tag{2.4}$$

And then the context vector $\mathbf{c}_t$ a time t is created from a weighted sum of the softmax scores and the encoder hidden states $\mathbf{h}_n$

$$\mathbf{c}_t = \sum_{n=1}^{N} \alpha_{tn} \mathbf{h}_n \tag{2.5}$$

If we represent the output probability of the decoder as function g, then the output conditional probability for $\mathbf{y}_t$ is a function of the specific context vector $\mathbf{c}_t$ for timestep t, the last output $\mathbf{y}_{t-1}$, and the last hidden state $\mathbf{h}_{t-1}$ [28].

$$p(\mathbf{y}_t | \mathbf{y}_1, ..., \mathbf{y}_{t-1}, \mathbf{X}) = g(\mathbf{y}_{t-1}, \mathbf{h}_{t-1}, \mathbf{c}_t) \tag{2.6}$$

They called this an *attention mechanism*, as the probability α_{tn} obtained, shows the importance of the encoder hidden state $\mathbf{h}_n$ with respect to the previous decoder hidden state $\mathbf{h}_{t-1}$ in producing the current decoder hidden state $\mathbf{h}_{t-1}$, and producing the current decoder output $\mathbf{y}_t$. The decoder is deciding at what part of the encoder to look at [28].

Following Bahdanau, Cho, and Bengio [28] and Luong, Pham, and Manning [29], in Figure 2.3 an attention mechanism in translation is simulated and exemplified. Attention tells what the different parts of the encoder to focus

on different parts of the decoder. In this example, we translate *The cat is black* into its French counterpart *Le chat est noir*. The weights are simulated just to visually show how the different words would have a different attention score with respect to its translation. For instance, the word *cat* could have a large attention score with respect to *chat*, but would still hold an attention score with *Le* and *est* as it still holds a relationship with those words (the article and the verb that are attached to the noun).

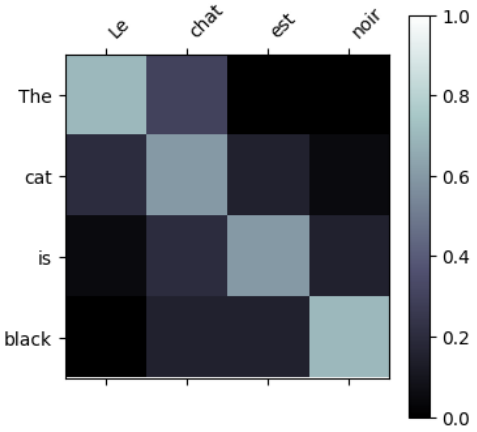

Figure 2.3: Attention example. Weights were synthetically created. The purpose of image is to exemplify attention.

Later in Luong, Pham, and Manning [29], they would call this *global attention*, as at every generative step of the decoder, the network looks at the attention scores of the whole input sequence.

2.3.5 Transformers

Attention solved the problem of long-range dependencies that past encoder-decoder models had. Still, the problem of parallelized computation remained. In 2017, Vaswani et al. [24] introduced the Transformer network. The Transformer network is an encoder-decoder architecture, that relies solely on attention mechanisms, to build global dependencies between input and output. The encoder and decoder *attention blocks* are composed of multi-head attention mechanisms, and feed-forward networks. The Transformer network is able to process all the tokens in a sequence at the same time, allowing computation to be parallelized, and making it a well-suited choice

for processing long sequences [24].

Figure 2.4 shows the Transformer architecture from Vaswani et al. [24]. The component on the left of Figure 2.4 is the encoder, and the component of the right is the decoder.

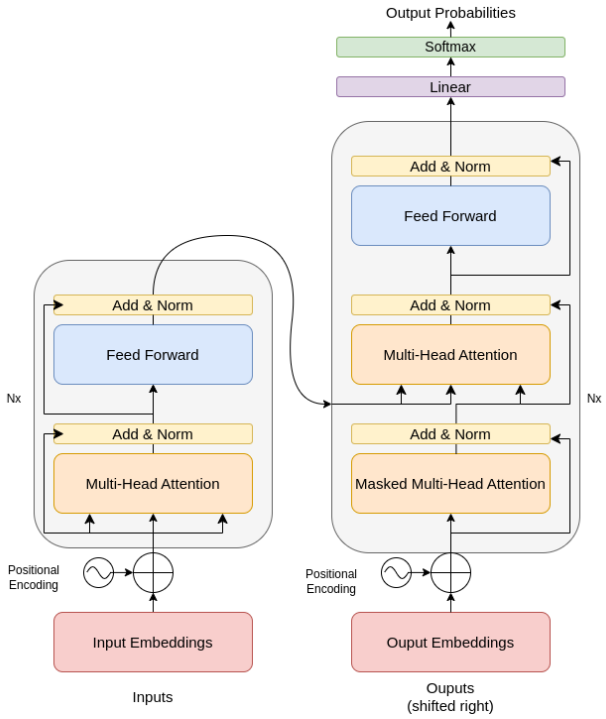

Figure 2.4: Transformer Architecture. Following the model from Vaswani et al. [24]

A key component for both the decoder and encoder is the multihead self-attention. Self-attention is the weight that an element in a sequence has within the other elements of the sequence [24, 30]. This is in contrast to prior attention mechanisms Bahdanau, Cho, and Bengio [28], that compute attention from elements in the encoder hidden states to the elements in the decoder hidden states. It is called multihead, because it is done several times, with the purpose that every head learns a different relationship between words. In self-attention, we compute a weighted average of each word embedding based on the other weights of the sequence, and the weights are called the attention weights. The embeddings of this type are often called *contextualized embeddings*. To give as

an example, the word ”drain” might have several meanings, it could be a verb or a noun. But by creating contextualized embeddings from the other elements in the sequence, then the embeddings have a different form if the word is in phrases like ”I feel an energy drain” or ”The drain is clogged”. Formally, new embeddings $x'_1, ..., x'_n$ are created out of multiplying the original embeddings $x_1, ..., x_n$ by the attention weights W. [30].

$$x'_i = \sum_{j=1}^{n} w_{ji} x_j \tag{2.7}$$

In Vaswani et al. [24], they use scaled dot-product attention, which uses a query, key, and value vector to compute the attention weights W. The query and key vector are multiplied to see their similarity (dot-product), then the vector is scaled and passed through a softmax function. The weights are obtained by multiplying the output of the softmax by the value vector. The attention-blocks do this multiple times, not only having one transformation, and hence having multiple attention weights. This is referred to as multihead self-attention, and the purpose is to have several aspects of similarity, and different relationships between words. [24, 30]. The attention heads are concatenated and passed through a feed forward network. The multihead attention, and the feed forward network compose an attention block. Figure 2.5 shows a diagram of the scaled dot-product attention, and the multi-head attention block, from Vaswani et al. [24].

The encoder is composed of several encoder blocks stacked on top of each other. As input, they take the token embeddings. After every encoder block, the output are embeddings of the same size as the original embeddings, but now contain contextual information of the other parts of the sequence. Every encoder block acts as an ”update” to the token representations. [24, 30] The main job of the encoder is to create a rich numerical representation of the input sequence that contains all the information and relationships between the words of the input sequence.

The decoder is also composed of attention blocks, but they have 2 main differences between encoder blocks. The first difference in decoder blocks is that they have a *masked* multi-head self-attention layer. The masked component makes sure the decoder does not look into the future to generate a token, which would make the task trivial. It only looks at past tokens. [24,

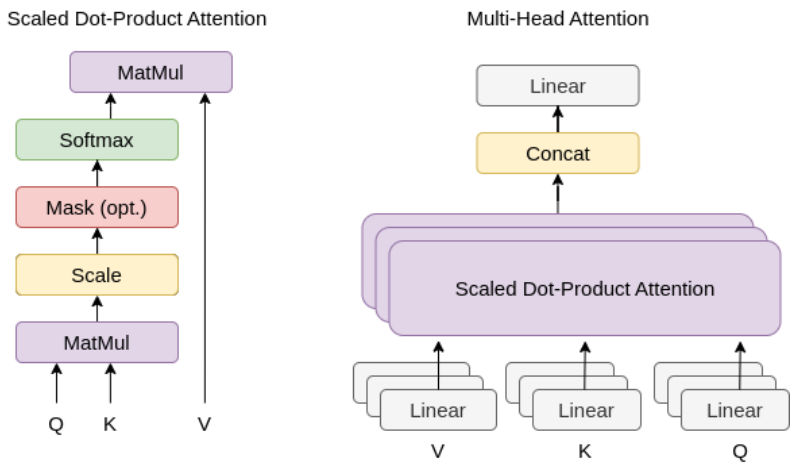

Figure 2.5: Scaled Dot-product attention, and multi-head attention. Following the model from Vaswani et al. [24]

30] The second difference is that after the masked sel-attention, there is an encoder-decoder attention layer, which does multi-head attention between the output of the encoder and the intermediate representations of the decoder. [24, 30] After the encoder-decoder attention layer, the token embeddings go through a feed forward network, just like in the encoder blocks.

The original Transformer architecture was an end-to-end encoder-decoder model, and trained on a supervised manner, on a corpus of German/English paired sentences for the task of translation [24]. It is composed of 8 heads in the encoder and 8 heads in the decoder, and 6 stacked encoder blocks, and 6 stacked decoder blocks [24]. Nowadays, there exist different implementation of language models based only on Encoder architectures, only on Decoder architectures and Encoder-Decoder architectures [30]. The family of GPT [5] models belong to the Decoder-only architectures, and BERT [3] and its variants, belong to the Encoder-only architectures. T5 [4] is a model that belong to encoder-decoder architectures, which are used to map from one sequence of text to another, just as the original transformer. New models used pre-training to take advantage of unlabeled data, to later be fine-tuned to specific tasks.

2.3.6 GPT

Decoder models are great at finding out the next word in a sequence, and GPT [5] was an autoregressive model introduced by OpenAI [31] based on the decoder part of the Transformer. GPT introduced the concept of ”generative pretraining”, which is an unsupervised method in which the model is trained on huge amounts of text, and it learns to predict the next word in a phrase or sentence based on these texts. It was trained on the BookCorpus dataset [5]. After pretraining, the GPT model can be fine-tuned with a supervised objective to several tasks. The original GPT model consisted on 12 decoder-only attention blocks and 12 attention heads, with approximately 117 million parameters. [5]

Further improvements to the GPT model were introduced by GPT-2 [12], which increased to 1.5 billion parameters and later GPT-3 [13] which increased to 175 billion parameters. The GPT-3 model leveraged a bigger compute, a bigger model, and a bigger dataset, and was a model that could now create more complex text passages and showed few-shot learning capabilities. *Few-shot learning* is the capability of a model to be provided with a few examples of a task, and then the model is able to generalize and complete the task in a new scenario. This differs from *zero-shot learning*, in which a model is able to complete a task without showing it any previous examples. The GPT-3.5 model is available to prompt through the OpenAI API. [31] More recently, the GPT-4 [6] model was also released, which improves on models GPT-3 and GPT-3.5. An extensive evaluation of the model’s capabilities are provided [6], but the details on the model’s technical details such as architecture have yet to be fully released.

2.3.7 BERT

On the side of Encoder-only models, Google introduced BERT [3]. As GPT, BERT made use of pre-training on a large unlabeled corpus of text, to later be fine-tuned for specific tasks on smaller amounts of labelled data. [3] BERT improved upon GPT [5] which was an earlier implementation of a pre-trained language model. Besides GPT being a ”decoder based” model, and BERT being an ”encoder based” model, a main difference between them was that GPT was uni-directional, meaning that the architecture was made left-to-right, and BERT was bi-directional. A left-to-right functioning is also called autoregressive, every generated token can only access the past tokens in the self-attention components, and bi-directional means it can access the context

before and after, at the left and right of the token.

BERT was the first deep bidirectional unsupervised trained language model. Being bidirectional had implications on the contextual embeddings built by the encoder blocks. Since it is bidirectional, it builds the contextual embeddings by looking at the whole context, left, and right of the word. As an example, in the 2 phrases:

1. I am arriving to the bank to exchange money.
2. I am arriving to the bank with my fishing pole.

An autoregressive approach builds the same contextual embedding for the word ”bank”, since it can only access the words to the left, but the contextual embeddings are different on a bidirectional approach, since it can access words to the left and to the right.

BERT is pretrained in an unsupervised manner with 2 objectives, the first one being masking. Masking is a technique in which during training, a word is substituted with the special token *[MASK]*, and the objective is a cross-entropy loss to predict the masked word [3]. Besides being trained with a masking objective, BERT is also trained with a ”Next Sentence Prediction” objective, which given 2 sentences, A and B, the model predicts True or False if sentence B is after sentence A. [3].

BERT is a masked language model, and it was introduced in 2 different versions, the base version which had 12 encoder transformer blocks, and 12 attention heads, adding up to 110 million parameters, and the large version which had 24 encoder transformer blocks, 16 attention heads and added up to 340 million parameters [3]. Upon the release of BERT, Devlin et al. [3] also released results on 11 NLP tasks, that were competitive results compared to the state-of the art approaches, and included the task of Question Answering on the Stanford Question Answering Dataset, or SQUAD.

Language models such as BERT and GPT can be used and/or fine-tuned to specific tasks. A BERT model can use masking to learn to produce plausible distractors, by providing it with the distractors and learn to predict the word that best fits the masked position, given the provided context to the model. The task of Distractor Generation uses Multiple Choice Questions datasets as context.

2.4 Multiple Choice Questions

As mentioned, the characteristics of the MCQ dataset for this task has to contain a text, a question, and a set of plausible distractors. There is a variety of MCQ datasets that are used for the task of Question Answering and for Distractor Generation, some common ones are RACE [32], which is a dataset of English reading comprehension originally intended to test Chinese students from middle and high school, and the SQuAD dataset [33], the Stanford Question Answering Dataset, a dataset of Wikipedia text extracts and more than 100,000 questions (there are multiple questions per text), in which the correct answer to the question appears in the text. Even though the SQuAD dataset contains answers in the text, it does not contain any distractors.

There exist more limited options in Spanish, but some datasets found were Xquad [34], which is a human professional translation of 240 texts and 1190 question-answers into 10 different languages, one of them including Spanish. Another option is SQuAD-es [10], which is an automatic translation of the SQuAD dataset. SQuAD-es is translated using a TAR method, Translate-Align-Retrieve. In Carrino, Costa-jussà, and Fonollosa [10], the original source text, question, and answer from the original SQuAD dataset is translated using a trained Transformer model, then the source context and target context are aligned at a sentence level, and then the answer is retrieved from the source in order to have the correct position of the target answer in the target text. Since SQuAD-es is an automatic translation of SQuAD, the dataset does not contain distractors. Another more recent Spanish dataset for MCQ was used in IberLEF 2022 for the ReCoRES task [35]. It is a MCQ dataset in Spanish with texts, questions, answers and distractors, with the objective of answering a question given reading comprehension, and then generating the reasons why the picked option was chosen. ReCoRES is a dataset of 439 texts and 1822 questions. Since ReCoRES objective is focused on reasoning, the correct answer and distractors are not in the text. In order to answer questions in ReCoRES, a student requires abilities beyond reading comprehension, but language understanding and critical thinking.

In Kalpakchi and Boye [8], *SweQUAD-MC*, a small-scale Swedish MCQ dataset is built, with an approximate size of 1200 questions and texts (multiple questions per text) for the task of Distractor Generation. The dataset is manually tagged and built using Textinator [16], an open-source text-annotating tool. The Textinator tool allows you to write a question for a text,

and then tag through a highlighting tool a text extract to label it as either the answer or a distractor.

MCQs can be used to address the task of Question Answering, which is not the focus of this work. But MCQs can also be used for a much less explored task, such as DG. DG has received less attention, but nevertheless, has been studied and implemented in different scenarios and contexts.

2.5 Distractor Generation & Related Work

In Susanti et al. [36], they explore distractor generation in the context of a multiple-choice English vocabulary question. They explore it in the context of standardized language tests, such as the TOEFL and the IELTS, which are exams for ”English as a second language” tests. The questions in English learning exams are mostly asking closest-in meaning vocabulary questions. These questions show a passage and a highlighted word, and the student is asked to choose a word from a set of options that mostly resembles the highlighted word. Their method consists on getting a set of candidates from a list of synonyms and siblings of the target word from the WordNet taxonomy, which is a lexical database. From this list of plausible candidate distractors from these sources, filtering is performed based on question writing guidelines. The remaining distractors are ranked by cosine similarity of their GloVe embeddings to choose the top distractors.

In Susanti et al. [36], they also provide with a thorough evaluation method that is divided into 2 parts, a student evaluation and an expert (teacher) evaluation. The student evaluation consists on asking English learners to answer questions, in which the reading passage, the question and the correct answer are the same, they only differ from the distractor set. The distractor sets are either human-made, from the baseline method, or from the challenger method (proposed in this study). In the teacher evaluation, a teacher evaluates all the sets from the perspective of an educator and an item writer.

Qiu, Wu, and Fan [7] address distractor generation also for English standardized tests (TOEFL and SAT), and propose an EDGE framework, qu**E**stion and answer guided **D**istractor **GE**neration. In this work, they create a sequence-to-sequence network to generate the distractors. They first attempt to encode the reading passage, the question, and the answer all together to extract the contextual semantic representation of all the text through three

matrices, and then use an attention mechanism based on the scaled dot product to enrich this encoded semantic representation. After a reforming module, they use an attention-based network to generate the distractors. The specific focus is on the RACE [32] dataset. They make great focus on the generated distractors quality, and assure that the distractors generated are both *incorrect*, and *plausible*.

Zhou, Luo, and Wu [37] create a co-attention hierarchical network, which is an encoder-decoder model. The encoder is hierarchical because it first gets word-level encoded representations from the text, and then based on these representations gets sentence-level encoded representations. Both word-level and sentence-level representations are included in the decoder. It is called a co-attention network because after the hierarchical encoding, they incorporate attention from "reading passage-to-question" and also attention from "question-to-reading passage", which provides with richer output vectors that model better the interaction between question and reading passage. The decoder uses the word-level and sentence-level representations from the encoder at every decoding step. They add to the loss function, an additional semantic similarity loss based on the cosine similarity to ensure that the generated distractors gave a high semantic relevance with the reading passage. In this work, they focus specifically on the RACE [32] dataset as well, just as in Qiu, Wu, and Fan [7].

Offerijns, Verberne, and Verhoef [38] use GPT and BERT to generate distractors. Their proposed method has 3 separate models, a question generator, a distractor generator and a Question-Answer filter. In the question generator part, they fine-tune a GPT-2 model on the English SQuAD [33] v2 dataset to be able to create questions given the context and the answer as input. Once questions generated and initially filtered as "answerable", then for the distractor generation module, they fine-tune a second GPT-2 model on the RACE dataset [32], that takes as input the context, answer and the previously generated question, to generate 3 plausible distractors. In this part they ensure that the distractors are non-repeated, non-empty strings by applying a repetition penalty and by repeating the generation step until three unique distractors were generated. The final part is the Question-Answering filtering, which is done by a DistilBERT model [39]. The DistilBERT model is a smaller version of BERT that uses knowledge distillation in the pretraining phase, to obtain competitive results related to BERT and reducing its size by 40%. The Question-Answering filtering is a classification task by the DistilBERT model

in which the model is trained to answer correctly the question given the context, question and answers, correct and distractors. The context-question-answer-distractor groups are filtered out depending if the model is able to answer them or not. If the model is unable to answer the question, it is assumed that there is something wrong with the question of the distractors, so the group is filtered out.

Chung, Chan, and Fan [40] use a BERT model and fine-tune it to perform DG. They provide as input the reading passage, question and correct answer, and refer to this as the context **C**. They generate the distractor in an autoregressive fashion by using masking prediction on the next token. Table 2.1 shows an example of the autoregressive generation, in which the mask [MASK] is always generated after every iteration and a token [S] separated the context from the distractors, and the distractors from each other.

Iter	Input Sequence	Predict
1	[C] C [S] [MASK]	Because
2	[C] C [S] Because [MASK]	Henry
3	[C] C [S] Because Henry [MASK]	didn't
4	[C] C [S] Because Henry didn't [MASK]	want
5	[C] C [S] Because Henry didn't want [MASK]	to
6	[C] C [S] Because Henry didn't want to [MASK]	go
7	[C] C [S] Because Henry didn't want to go [MASK]	.
8	[C] C [S] Because Henry didn't want to go. [MASK]	[S]

Table 2.1: Autoregressive Distractor Generation. Following model from Chung, Chan, and Fan [40]

Besides the autoregressive DG they improve the quality of the generated distractors by also jointly training a parallel masked language model and creating a multitask architecture, and also by applying negative answer regularization, which penalizes distractors that are close to the correct answer and could be confused with the correct answer. [40]

The work done in Kalpakchi and Boye [8] serves as a base for the current work. In [8], they make use of a BERT model trained on a Swedish corpus to perform DG. The Swedish BERT model was trained by Malmsten, Börjeson, and Haffenden [9] and made available through Hugging Face platform [17]. This same work recollects the MCQ dataset *SweQUAD-MC* presented in Section 2.4. Similar to Chung, Chan, and Fan [40], the context which is composed of the test, question and answer, is fed into the BERT model. On top of the BERT model, there are 2 trainable linear layers. Distractors are generated using 2 techniques, the *left-to-right* method (autoregressive) and the *u-PMLM* method, as shown in Table 2.2 and Table 2.3, respectively. Table 2.4 has the same example from Table 2.1 but with the u-PMLM scheme.

Input Sequence for left-to-right (autoregressive)	Target
[CLS] CTX [SEP] [MASK]	D11
[CLS] CTX [SEP] D11 [MASK]	D12
[CLS] CTX [SEP] D11 D12 [MASK]	[SEP]
[CLS] CTX [SEP] D11 D12 [SEP] [MASK]	D21
[CLS] CTX [SEP] D11 D12 [SEP] D21 [MASK]	D22
[CLS] CTX [SEP] D11 D12 [SEP] D21 D22 [MASK]	D23
[CLS] CTX [SEP] D11 D12 [SEP] D21 D22 D23 [MASK]	[SEP]

Table 2.2: Left-to-Right Distractor Generation. Following model from Kalpakchi and Boye [8]

Input Sequence for u-PMLM	Target
[CLS] CTX [SEP] D11 [MASK]	D12
[CLS] CTX [SEP] [MASK] D12	D11
[CLS] CTX [SEP] D11 D12 [SEP] D21 [MASK] [MASK]	D22, D23
[CLS] CTX [SEP] D11 D12 [SEP] D21 [MASK] D23	D22
[CLS] CTX [SEP] D11 D12 [SEP] [MASK] D22 [MASK]	D21, D23

Table 2.3: u-PMLM Distractor Generation. Following model from Kalpakchi and Boye [8]

Iter	Input Sequence	Index	Predict
1	[C] C [S] [M] [M] [M] [M] [M] [M] [M]	1	Henry
2	[C] C [S] [M] Henry [M] [M] [M] [M] [M]	4	to
3	[C] C [S] [M] Henry [M] [M] to [M] [M]	5	go
4	[C] C [S] [M] Henry [M] [M] to go [M]	2	didn't
5	[C] C [S] [M] Henry didn't [M] to go [M]	0	Because
6	[C] C [S] Because Henry didn't [M] to go [M]	3	want
7	[C] C [S] Because Henry didn't want to go [M]	7	.
8	[C] C [S] Because Henry didn't want to go.	8	[S]

Table 2.4: u-PMLM Distractor Generation. Using example from Chung, Chan, and Fan [40], method is from Liao, Jiang, and Liu [41]

The autoregressive variant works similarly to the method in Chung, Chan, and Fan [40], in which the tokens are generated one after the next one by predicting the [MASK] token. The *u-PMLM* variant method has the motivation from Liao, Jiang, and Liu [41]. In Liao, Jiang, and Liu [41], a probabilistic masking scheme is proposed, in which a sequence $S = \{s_1, s_2, ..., s_N\}$ has all tokens as [MASK] tokens, and an index n is chosen from a uniform prior distribution (randomly). Then, the language model is input with the sequence *S* and is tasked to only predict the token at position n. The predicted word is placed in its corresponding index in sequence *S*, and the process is repeated for the remaining indexes. It is called a probabilistically masked language model (PMLM), with a uniform prior (*u-PMLM*). Probabilistically masked language models are the bridge between masked and autoregressive language models, and look to assimilate the text generation capabilities of autoregressive models in masked models.

There are extensive qualitative and quantitative evaluation schemes in

this work. They propose custom quantitative metrics to assess distractor generation, and divert from the traditional BLEU and ROUGE scores. The motivation is that a distractor might still be plausible and of good quality, even though the BLEU and ROUGE scores are low, and vice versa, a high BLEU and ROUGE score but an invalid distractor given a repeated word [8]. Some proposed custom quantitative metrics were used in this work, such as *Distractor Recall*, which is simply the ability to recall the ground truth distractors. The qualitative evaluation is based on student and teacher tests, similarly done in Qiu, Wu, and Fan [7]. Kalpakchi and Boye [8] also contains a more comprehensive survey with related work on distractor generation.

Chapter 3

Methodology: Dataset

The research process and methodology after the literature review is summarized in the next steps:

Step 1 Collect texts and MCQs in Spanish.

Step 2 Tag questions, answers, and distractors to create a dataset for distractor generation.

Step 3 Apply the BERT-based methods, *left-to-right*, and *u-PMLM* to automatically generate distractor.

Step 4 Use GPT model in a zero-shot fashion to generate distractors.

Step 5 Perform human testing of the distractors to have a qualitative evaluation.

Step 6 Analyze results.

As data collection was very central to this work, this chapter contains the first 2 steps regarding building the dataset. Section 3.1 describes the process in deciding the base dataset to use for the task of DG. This section builds on the background from Section 2.4 to assess the datasets found and their usability for this work. Based on the chosen MCQ dataset, Section 3.2 focuses on the process of manually building and adapting the MCQ dataset to be a DG dataset. Section 3.2 also explains the principles behind the creation of the current DG dataset. We will refer to a *data point*, as a set consisting of a text T, a question Q, a correct answer A, and a series of distractors $D_1, D_2, ..., D_N$.

3.1 Data Collection

To fine-tune a BERT model on the task of Distractor Generation, a Spanish MCQ dataset was needed. As mentioned in section 2.4, there exist limited options on Spanish MCQ datasets. For this task, the Spanish MCQ dataset needed to comply with the following requirements:

1. Be in Spanish language.
2. Have a similar size in training, development, test splits as in Kalpakchi and Boye [8].
3. Have the correct answer and distractors in the text.

Recall that the objective scenario of this work is in reading comprehension, where we seek to generate good-enough distractors to confuse Spanish learners. The SQuAD [33] dataset is a dataset that fits the DG task because it contains the correct answers in the text, and is big in data point quantity. Both SQuAD-es [10] and XQuAD [34], are translations of the SQuAD dataset into Spanish. The XQuAD dataset is a professional human translation, making it small in size, with around 1000 data points. When the tagging is done for distractors, a lot of texts do not qualify as usable for the DG task, because some texts are too short or simply do not contain plausible ground truth distractors in the text. This means that the final number of data points if XQuAD was used would be less than 1000 data points, which is less than the size used in Kalpakchi and Boye [8]. For this reason, XQuAD is not a viable solution to pick as a source of texts, questions, and answers.

SQuAD-es contains over 18000 texts. In Carrino, Costa-jussà, and Fonollosa [10], results related to the correctness of the translation from the original SQuAD to SQuAD-es are presented. The reported errors are of 2 types: A misaligned span and an overlapping span. [10]. It is mentioned that both errors depend on the translation by the language model, and also the alignment algorithm. By looking at the results, it is not possible to see the exact correctness of the quality of the translation only. Given the big size of the dataset, when performing manual tagging of the distractors, the Spanish tagger was able to evaluate if a text had a translation, spelling or grammatical mistake. In that case, the text was just discarded, and another text could be loaded. Given then the characteristics of SQuAD-es, SQuAD-es was the final dataset used for its texts to proceed into tagging.

The ReCoRES [35] dataset has a size of +1800 questions and +400 texts. This dataset checks the first 2 requirements, it is a Spanish MCQ dataset and is similar in size to the dataset in Kalpakchi and Boye [8]. But the correct answer and distractors are not in the text. The ReCoRES dataset was used in the context of question answering and reasoning on the chosen answer, so it went beyond the desired scope of this work which is reading comprehension, as it lies more in reading understanding and critical thinking. A learner must read the text and draw conclusions from it, in order to answer the question. Generating distractors for texts that do not contain the distractors remains as future work.

3.2 Manual DG Dataset Construction

The texts from SQuAD-es 2.0 version from the training set were imported into Textinator [16] to build the DG dataset. Textinator is able to first load the context texts, and then provide a text input to write a question and the ability to the user to highlight in green the text that is the correct answer and highlight in red the text that is a distractor.

In this section, the verb *"tagging"* is referred to as the process of manually building the DG MCQ dataset, since the user of Textinator is *"tagging"* phrases as either correct answers or distractors. The person constructing the questions and the distractors through Textinator is referred to as the *"tagger"*. A screenshot of the Textinator interface can be seen in Appendix B.

Through Textinator, the tagger read the texts, came up with questions about the text, and then tagged inside the text, the correct answer, and the plausible distractors. From this tagging procedure, the dataset ***SQuAD-es-dist*** was built.

The type of answers and distractors vary in their nature and the type of information they contain. A question is usually answering one or more of *What?, Who?, When?, Where?, Why?* and *How?*. Even though the texts of the SQuAD-es dataset included questions and answers, the questions and answers were not always correct, and if correct, the text still needed to have the distractors inside it. This means not all the questions and texts in SQuAD-es, were viable to include in the dataset. Nevertheless, the original questions and answers from SQuAD-es served as a great starting point for the tagger to come up with a question, its correct answer, and a set of distractors, when manually tagging in Textinator.

For more complicated and longer texts, the ChatGPT [15] interface was used to aid the tagger by introducing a text from the dataset and a prompt that instructed ChatGPT to provide with 3 sets of good questions, answers, and distractors that could apply to the text. The answers by ChatGPT were faulty, as some were grammatically incorrect, the correct answer was wrong, and/or the distractors were not from the reading passage. Nevertheless, the tagger used ChatGPT's answers as a starting point for the tagger to come up with a correct set of question, correct answer and distractors for a given text. The prompt used to aid the tagger in ChatGPT is included in Appendix C.

3.2.1 Tagging Principles

Besides following the guidelines provided in Section 2.2, the tagger followed some lines of thought to build correct question, answer, and distractor sets. When using Textinator, the tagger first got showed a text, and the first thing to determine was if the text was long enough to proceed with finding a question and answer for the text. There was no established rule, but the tagger usually skipped the text if it was less than 5 or 6 sentences long.

After determining if the text was long enough, the tagger proceeded to read the text, and try to come up with a question that answers something that is in the text. Since the texts are from different topics, from biology to history, and art to medicine, some texts took more time to digest and comprehend. With complicated texts, the tagger used the ChatGPT aid prompt (Appendix C) as described previously as a guide. If the tagger was unable to find a question with an answer in the text after reading the text and using ChatGPT, then the tagger skipped the question. Once the tagger came up with a question and an answer, the tagger wrote correctly the question using a language tool to ensure correct grammar, and also highlighted in green the correct answer.

After finding a question, and an answer in the text, the tagger did the last step of finding at least 2 distractors in the text, and highlight the distractors in red. If the tagger was unable to find at least 2 distractors, then the text was skipped. After this step, the tagger submits the question, and gets shown a different text to start the process again. The tagging process is visually seen in Figure 3.1

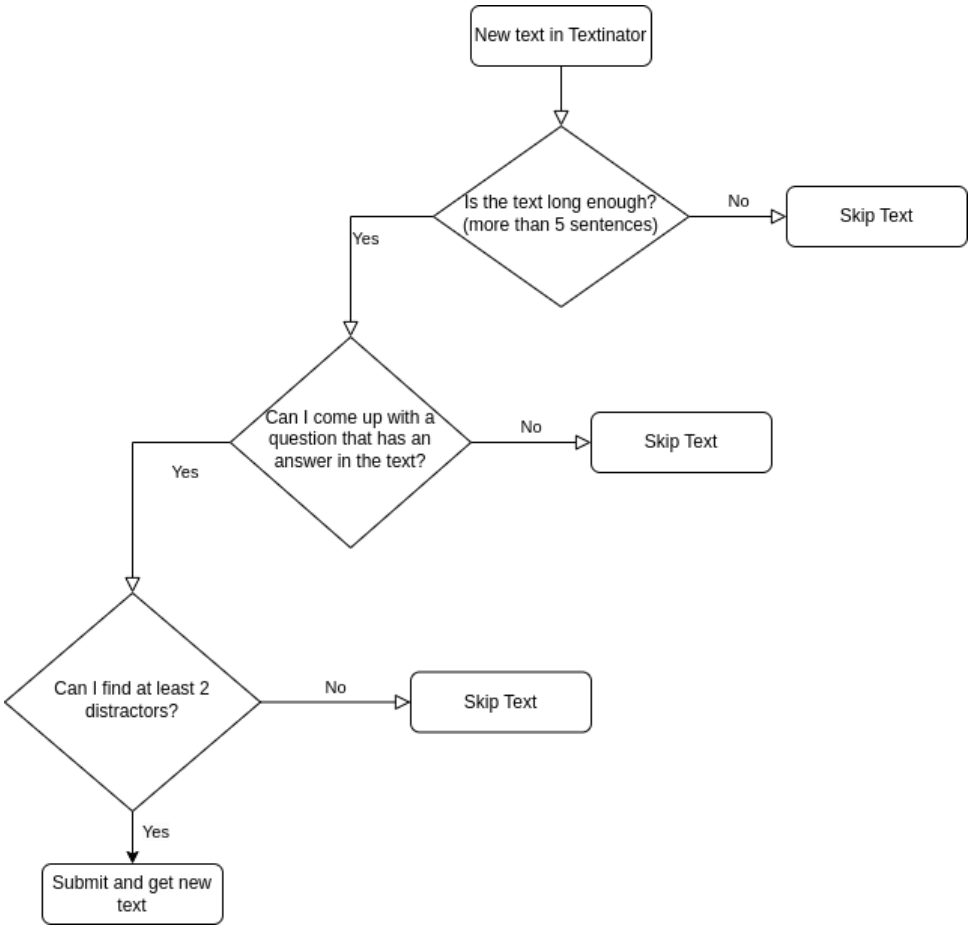

Figure 3.1: Flowchart of tagging process

Distractors have a grammatical aspect as well. As a student, one could identify grammatical details that could give away the correct answer, such as verb tense, capitalization of letters, length, and semantic group between the correct answer and incorrect answer. One can infer that a difference in capitalization could mean giving away the correct answer or an evidently erroneous distractor. These are all points that must be taken into account.

As an example, let's pretend the question "What factor was the main reason for the French army's defeat in the invasion of Russia?" belongs to a MCQ dataset, and has the following options, which contain grammatical differences between each other:

Incorrect Multiple Choice Options

1. The weather was brutal for the French army, since they were not prepared for the cold conditions.
2. the diseases decimated half of the french forces, and obligated a quick retreat.
3. The lack of food was very important for the retreat of the French army.
4. The unclear leadership will be devastating for the French plans on the invasion of Russia.

Let's assume the correct answer is the first option (1). Although the distractors (2, 3, 4) remain in a logical and semantic field, a student could notice the variations in the grammatical format of the options, and discard some options. Option 2 does not start with a capital letter and has the word "french" uncapitalized, while the other 3 options do. While this does not mean that option 2 is wrong, this could lead to a student into having intuitions that this option is wrong, by simply assessing the grammatical differences. A similar case applies to option 4, in which the verb tense is the future tense. Since the question asks in the past tense, the options should all follow the past tense. Option 4 being in future tense could give an intuition to the student that this answer is wrong. A more grammatically correct set of options, would be composed of better distractors for students:

Correct Multiple Choice Options

1. The weather was brutal for the French army, since they were not prepared for the cold conditions.
2. The diseases decimated half of the French forces, and obligated a quick retreat.
3. The lack of food was very important for the retreat of the French army.
4. The unclear leadership was devastating for the French plans on the invasion of Russia.

If the tagger encountered possible distractions that were not in the correct grammatical form, the tagger could make a posterior editing to the distractor, to ensure that it was of the same grammatical form as the correct answer (e.g.

changing it from singular to plural).

In Section 1.1, an example to introduce the concept of distractors is shown in Figure 1.1. The question *What year was the invasion of Normandy?* has a correct answer *"1944"*; has bad distractors when the distractors are *"France"*, *"June"*, and *"Frankin D. Roosevelt"*; and has good distractors when the distractors are *"1941"*, *"1942"*, and *"1943"*. The bad distractors are all semantically incorrect to the right answer, because the distractors are not a year, and a student would know that the correct answer must be a year. For effective and correct distractor tagging, the tagger needs to consider that good distractors must be semantically similar to the correct answer.

But another dimension that the tagger must take into account is the possible "common knowledge" that a student could have. Suppose generated distractors were *"1023"*, and *"2023"*. Even though these 2 options are semantically correct to the right answer, "common knowledge" could help a student identify that these are not the correct answer. The invasion of Normandy happened in the second World War, which a student could know that it did not happen in *"1023"*, and much less in *"2023"*. Including "common knowledge" as a principle makes the task of manually tagging distractors more complex. Nevertheless, it was a consideration taken at the moment of tagging data points, that could not always be followed to the rule, given that "common knowledge" is in the end a subjective matter.

3.2.2 Types of Multiple Choice Options

When tagging, a tagger can encounter different ways to approach a question, that also defines the difficulty of a question. In Kalpakchi and Boye [42], there are factors that affect the difficulty of a question: Type of Information, Type of Match, Plausibility of Distractors.

- Type of Information: How abstract the stem is. Easier difficulty if it asks for concrete things, like places or people, and harder as it asks about more abstract concepts, like themes.[42]
- Type of Match: Level of inference. Easier difficulty if it requires string matching, so just reading the text, and harder as it critical thinking and reading between the lines is required. [42]
- Plausibility of Distractors: the closeness of the distractors to the correct answer. [42]

As seen in Section 2.1, where frameworks for reading comprehension were presented, the Four Resources model [19] explains reading comprehension in 4 levels. The 2 first levels were Code-Breaker and Text Participant, which focus mainly on understanding the text, and diverge from more critical tasks such as critical thinking. The Four Resources model and the identified determiners for question difficulty seen in Kalpakchi and Boye [42], served as a way of explaining intuition on the type of information, and served as a posterior guide to the process of tagging. When tagging, the tagger mostly focused on type of information, and that the type of information in the distractors just matches the one in the correct answer.

For this work, the tagger identified 3 different categories that ranged in the Type of Information category, that would also determine the difficulty of the question.

- Dates and Numbers: Answering questions like *When?*, *How much?*, *How many?*. These answers and distractors have everything related to numbers, including percentages, fractions, amounts, economic currency of some kind, and time-related items, such as years, months, and complete dates. Possible examples of these items are: ”five”, ”$400”, ”13%”, ”1st of January”, ”1999”, etc.

- Proper Names: Answering questions like *Who?*, and *Where?*. These answers and distractors are names of people, of countries and of cities. Possible examples of these items are ”John”, ”Canada”, ”Stockholm”.

- Short and Long Phrases: Answering questions like *Why?* and *How?*. These answers and distractors are extracts of the text that ask about what is happening in the text, an action someone or something took, and descriptions on something explained on the text. Possible examples of these items are ”because she took the wrong way”, ”three crowns and a pigeon”, and ”by conquering the Roman Empire”.

As general guidance, the tagger tried to keep the proportion of questions in the next percentages. While ”Dates and Numbers” and ”Proper Names” answers have its own challenges, the tagger deemed more important the category on ”Short and Long Phrases”, since it poses a higher difficulty than the other 2 categories, hence a bigger proportion in the total percentage.

- Dates and Numbers: 20%

- Proper Names: 20%
- Short and Long Phrases: 60%
- Total: 100%

To see an example of one of each of the 3 categories of the type of answers in Spanish, and with its English translation, see Appendix D.

3.2.3 Tagging Results

Since this is a work closely related to the work done in Kalpakchi and Boye [8], the Spanish dataset is desired to have a similar size in training, development, and test set to the Swedish dataset used in Kalpakchi and Boye [8]. Table 3.1 are the computed statistics of the train, development, and test set of the dataset in Kalpakchi and Boye [8].

The final size of the Spanish distractor dataset *SQuAD-es-dist* had 980 questions, spread out in 953 texts, 27 of the texts contained 2 questions. In the 3 categories, the distribution of MCQs is as following:

- Dates and Numbers: 135
- Proper Names: 216
- Short and Long Phrases: 629
- Total: 980

Table 3.2 contains the computed statistics of *SQuAD-es-dist*.

	training	dev	test
# of texts	434	64	45
# of MCQs	962	126	102
# of D	2.15 ± 0.54	2.13 ± 0.38	2.05 ± 0.22
Len(Text)	380.7 ± 332.8	352.1 ± 232.4	353.4 ± 263.7
Max(Text)	2142.0	2140.0	1591.0
Len(A)	4.19 ± 3.44	4.45 ± 3.73	4.58 ± 4.63
Max(A)	23.0	19.0	28.0
Len(D)	4.32 ± 3.84	4.09 ± 3.92	3.90 ± 3.63
Max(D)	24	29	21
|Len(A) - Len(D)|	2.06 ± 2.12	2.04 ± 1.99	2.02 ± 2.61
Max(|Len(A) - Len(D)|)	13.5	10.0	19.0

Table 3.1: Statistics on the Swedish MCQ dataset from Kalpakchi and Boye [8]

	training	dev	test
# of texts	810	95	48
# of MCQs	835	95	50
# of D	3.16 ± 0.74	3.23 ± 0.69	3.20 ± 0.75
Len(Text)	158.2 ± 54.9	158.5 ± 49.5	158.2 ± 60.8
Max(Text)	453.0	297.0	449.0
Len(A)	4.36 ± 4.49	5.45 ± 4.70	4.76 ± 4.37
Max(A)	26.0	22.0	19.0
Len(D)	4.59 ± 4.97	5.62 ± 5.53	4.08 ± 3.46
Max(D)	43	32	17
|Len(A) - Len(D)|	1.69 ± 2.28	2.08 ± 2.52	1.85 ± 2.00
Max(|Len(A) - Len(D)|)	16.66	10.33	8.0

Table 3.2: Statistics on the *SQuAD-es-dist* dataset

The differences are several. The number of questions is higher for the training, development and test set in *SweQUAD-MC*, compared to *SQuAD-es-dist*. There are almost 150 questions more for the training set in *SweQUAD-MC* compared to *SQuAD-es-dist*, which could have implications in the fine-tuning quality of the BERT model, but it is leveraged by the number of texts, which is much higher in *SQuAD-es-dist* compared to the number of texts in *SweQUAD-MC*. There are around 2 questions per text in *SweQUAD-MC*, and close to 1 question per text in *SQuAD-es-dist*. Another difference is in the average number of distractors, which is close to 3 in *SQuAD-es-dist*, and close to 2 in *SweQUAD-MC*. A last significant difference is the length of the texts. On average, the texts in *SweQUAD-MC* are over twice as long as the texts in *SQuAD-es-dist*.

Texts being shorter in *SQuAD-es-dist* was a main characteristic of the tagging procedure, as shorter texts raised the difficulty of tagging. A shorter text means there are fewer questions to ask about the text, and also fewer distractors to find inside the text. Multiple texts were discarded just because their length provided no content in order to come up with a question that had the answer and distractors in the text. From the tagging perspective, the length of the texts had an implication in the way the BERT model was trained as well.

The *SQuAD-es-dist* dataset and other related resources are publicly available as a GitHub repository. See Appendix E.

Chapter 4

Methodology: Implementation

Once the *Squad-es-dist* dataset was built, the next steps were to perform DG with BERT and DG with a GPT model. In Section 4.1, the method to generate distractors with BERT through the *left-to-right* and *u-PMLM* models is presented. In section 4.2, the method to generate distractors with a GPT model through zero-shot learning is presented.

4.1 Distractor Generation with BERT

The first step before proceeding to fine-tune using *SQuAD-es-dist* and a pretrained Spanish BERT, was to reproduce the fine-tuning of *SweQUAD* with the pretrained Swedish BERT.

In Kalpakchi and Boye [8], the method for DG is by fine-tuning a pretrained BERT in Swedish language. The chosen BERT for the Swedish approach was to use the pretrained base model from Malmsten, Börjeson, and Haffenden [9], which is available in the Hugging Face [17] platform, under the name of *KB/bert-base-swedish-cased*. Kalpakchi and Boye [8] add 2 linear layers and a layer normalization layer on top of BERT, as shown in Figure 4.1. BERT was fine-tuned in the 2 different schemes, *left-to-right* and *u-PMLM*. In both schemes, the network is fine-tuned by the cross-entropy loss of choosing the correct masked word, conditioned on the context. The context consists of the text, question, and correct answer.

BERT has a context size of 512 tokens [3]. This means the text, question, and correct answer, as well as the [MASK] tokens to generate the distractors, need to fit the 512 tokens. As many texts are more than 512 tokens, Kalpakchi

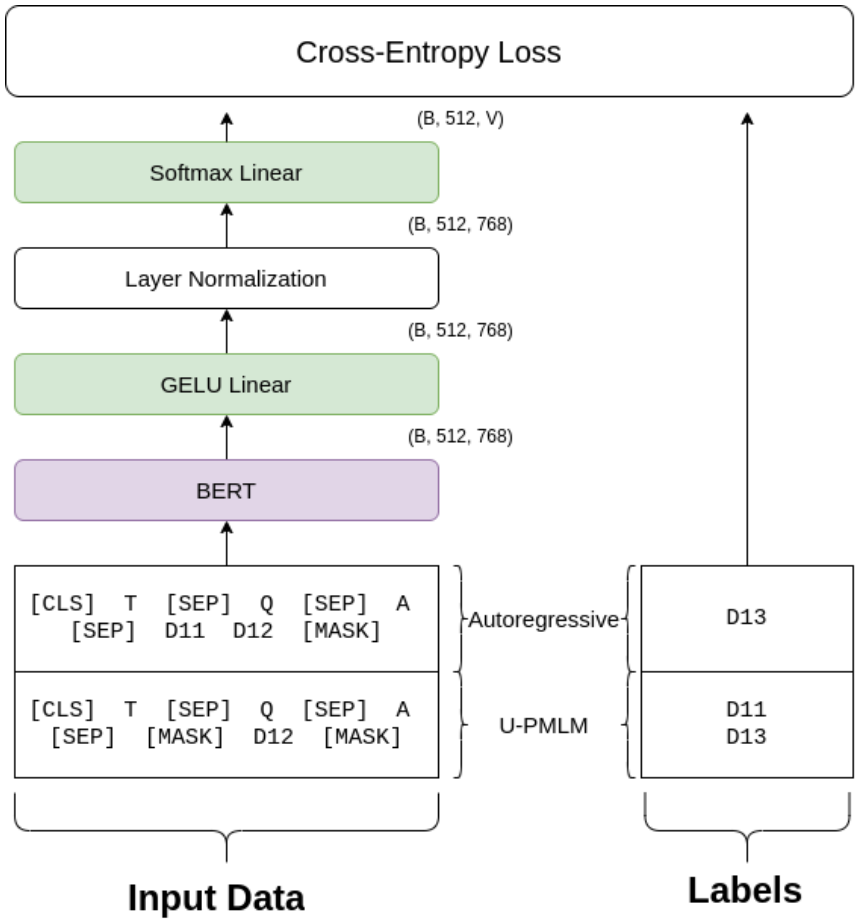

Figure 4.1: The DG scheme with BERT. Following the model from Kalpakchi and Boye [8]

and Boye [8] truncate the text to 384 tokens. This number comes from the average length of the texts, as computed in Table 3.1. The limitation is that for some texts, the entire context of information will not be available for training and inference, since only the first 384 tokens are available. For the *u-PMLM* training scheme, Kalpakchi and Boye [8] also set the number of [MASK] tokens per distractor to 20. The original repository for DG training using a pre-trained BERT from Kalpakchi and Boye [8] is found in the *SweQUAD-MC* Github repository, found in Appendix E.

Training was done in a 10GB NVIDIA GeForce RTX 3080. It was performed for 6 epochs, with a training batch size of 4. Training for each method (*left-to-right* and *u-PMLM*) took approximately 1 hour for each training scheme. Results for the reproduction of fine-tuning BERT for *SweQUAD-MC* on the development set are in Appendix F.1.

After analyzing results, and checking that they were comparable results to the ones achieved in Kalpakchi and Boye [8], the focus switched to the new dataset, and to use the same methods to replicate DG but with *SQuAD-es-dist* and a Spanish BERT.

For the Spanish approach, the chosen pretrained BERT model was BETO, from Cañete et al. [11]. BETO has the same architecture as BERT, but it is entirely pre-trained on a Spanish corpus. There is a cased and un-cased version [11]. The model is publicly available in the Hugging Face [17] platform, under the name of *dccuchile/bert-base-spanish-wwm-cased*.

Besides working with a different language (Spanish), a different dataset (SQuAD-es-dist), and using a different pretrained model (BETO - Spanish BERT), there were 2 modifications between the Swedish fine-tuning and the Spanish fine-tuning. The size of the text was truncated to 350 tokens, instead of 384. This is because the texts in *SQuAD-es-dist* were significantly smaller than the texts in *SweQUAD-MC* (158 average size v.s. 380 average size). By decreasing the size of texts from 384 tokens to 350, the second modification was to increase the number of [MASK] tokens to be 30, and we allow for longer answers to be generated in the *u-PMLM* variant.

The training was also done in a 10GB NVIDIA GeForce RTX 3080, with the same hyperparameters, for 6 epochs, training batch of 4, and an approximate training time of 1 hour for both schemes. Results for fine-tuning Spanish BERT using *SQuAD-es-dist* on the development set are in Appendix F.2.

4.2 Distractor Generation with GPT-4

Once a Spanish BERT was fine-tuned on *SQuAD-es-dist*, a second method of DG was done through zero-shot prompting and the GPT API. In this step, first several prompts were created and experimented with to generate distractors.

The GPT API provides with a *ChatCompletion* endpoint, in which a user can send a prompt, and the model will use the chosen model for text generation, generating tokens given the previous tokens [43].

The prompt designed included directions for the model to perform its task, and it contained the context of a data point, so the text, the question, and the correct answer. The model chosen was the *gpt-4* model. The temperature chosen was 0, since it is a retrieval task. The model only needs to retrieve plausible distractors from the text provided. Given that prompt engineering is a relative new field, extensive search for the optimal prompt was not done, but the general guidelines that are put together by OpenAI were used [44], such as adopting a persona, being specific with the task details, specifying the steps required, and using delimiters to indicate the distinct parts of the input. The prompt first explained what distractors were, what the task of DG consisted of, and then it asked the model to generate distractors given a text, question, and correct answer. The prompt also included specifics of the task, such as creating the distractors in a similar grammatical form to the correct answer, and to be sure that the generated distractors were not correct.

Experimentation was done with the development set, and once the prompt was refined and decided on a final version of the prompt, it was used to generate distractors on the test set. The Python file for DG through the GPT API is found in the *SQuAD-es-dist* Github repository, in Appendix E. The final prompt used in Spanish and its translation to English is found in Appendix G.

Chapter 5

Results and Evaluation

After generating distractors with BERT and GPT, evaluation can be performed. There were 2 types of evaluations performed. In section 5.1, the quantitative results for DG using BERT are presented. In section 5.2, the human evaluation performed using distractors from both BERT and GPT are presented.

5.1 Quantitative Evaluation of BERT results

Traditional language evaluation metrics, such as BLEU [45], focus on the overlap of words between a generated text and a reference text. In the case of DG, it would mean measuring distractor quality by the overlap of words between the generated distractors and the ground truth distractors. This creates an issue as these metrics become insufficient for the desired purpose, as a high score could still mean that the distractor quality is low, and vice versa.

For example, let us suppose the case where the model generates the distractor ”Because there is no water left in the Grand Canyon”, and the ground truth is ”Because there is water left in the Grand Canyon”. The generated distractor is the complete opposite of the ground truth distractor, which could mean it being a very bad distractor. Even though the generated distractor is a bad one, the BLEU score against the ground truth distractor is 0.65, which is an adequate score.

For this reason, Kalpakchi and Boye [8] define their own evaluation metrics, that are more fit for the specific task of DG. In this work, some evaluation metrics from Kalpakchi and Boye [8] are used. The evaluation

metrics are the following:

1. *Distractor Recall*: The percentage of distractors that were generated that match the ground truth distractors. The higher the percentage, the better.

2. *Any of the generated distractors matches with a gold one*: Percentage of MCQs that contain at least one distractor from the ground truth distractors. The higher the percentage, the better.

3. *Any of the distractors is in its own context*: Percentage of MCQs that contain at least one distractor from the context text. The higher the percentage, the better.

4. *The correct answer is among generated distractors*: Percentage of MCQs that contain the correct answer among the generated distractors. The lower the percentage, the better.

5. *Any (but not all) generated distractors are the same*: Percentage of MCQs that contain at least one distractor that is repeated from another distractor. The lower the percentage, the better.

6. *All generated distractors are the same*: Percentage of MCQs that all the generated distractors are the same. The lower the percentage, the better.

7. *Any of the distractors contains repetitive words*: Percentage of MCQs that contains at least one distractor with repeated contiguous words. The lower the percentage, the better.

8. *Any of the distractors is an empty string*: Percentage of MCQs that contains at least one distractor which is an empty string. The lower the percentage, the better.

9. *Any of the distractors is a distractor from training data*: Percentage of MCQs that contains at least one distractor that is from the training data ground truth distractors. The lower the percentage, the better.

After training for 6 epochs on both *left-to-right* and *u-PMLM* schemes, the best chosen model was decided based on the development set results, found in Appendix F.2. From the results in the development set, overall the *u-PMLM* variant is able to recall more distractors. The best checkpoint of the *left-to-right* variant (iteration 24000) is able to recall 27.54% of the distractors, while the best *u-PMLM* checkpoint (iteration 20000) is able to recall 31.48%

of the distractors. Although the *left-to-right* is better in certain metrics, such as distractors containing repetitive words, and any of the distractors in its own context, the overall performance is better for *u-PMLM*, as it is able to recall more distractors (Distractor Recall). Table 5.1 shows the test results on the best *u-PMLM* model.

	Best u-PMLM model
Total	50
Distractor recall	35.62%
Any of the generated distractors matches with a gold one	70.0%
Any of the distractors is in its own context	94.0%
The correct answer is among generated distractors	6.0%
Any (but not all) generated distractors are the same	2.0%
All generated distractors are the same	0.0%
Any of the distractors contains repetitive words	4.0%
Any of the distractors is an empty string	0.0%
Any of the distractors is a distractor from training data	0.0%

Table 5.1: Test Results for best u-PMLM model

The best BERT model is able to recall 35.62% of distractors, and shows great performance in the other metrics, such as having no MCQs with same distractors, empty string and distractors from training data, and also having a very low percentage of MCQs with faulty distractors, such as repeated words, or the correct answer among the generated distractors.

Even though the fine-tuned BERT model is able to recall less than 50% of the ground truth distractors, this does not mean that the generated distractors are not good ones, or implausible. Hence, the human qualitative evaluation is needed to evaluate distractor quality.

Quantitative results are not available for the GPT generated distractors, but in order to compare both, the qualitative evaluation was done on Spanish speakers.

5.2 Qualitative Evaluation on Spanish speakers

Once both models had generated distractors for the test set, the final evaluation consisted of a study with Native Spanish Speakers, to evaluate the performance of distractors in a real scenario. The study consisted of asking Spanish speakers to respond to the test questions with the different distractors, the ones generated from BERT and the ones generated by GPT. But, the students did NOT have access to the associated text to respond to the question. The motivation behind not showing the associated text is the following. Since it is reading comprehension, the assumption is that a Spanish speaker could easily respond to any question, if the text is present, regardless of the distractors. In order to really evaluate the performance of the distractors, the questions are shown without context, in order to force the respondents to answer the question by just looking at the multiple choice options. Good distractors would force the respondents to choose answers at random. But bad distractors would make respondents discard some options, and skew the accuracy positively.

As an example, suppose there is a test that consists of questions with 4 multiple choice options each, 1 correct answer and 3 distractors. A test with perfect distractors would make respondents answer all the questions randomly, having on average a 25% accuracy. Let's suppose now, that for every question, there is one distractor that upon reading, a student would know that the bad distractor is not the correct answer. So when choosing randomly, a student would now choose from 3 options instead of 4, having on average a 33.33% accuracy. In the case of 2 bad distractors, choosing randomly would yield a 50% accuracy, and so on. So the assumption is that the set with the worst distractors, makes the accuracy of the students go higher, while the better the distractors, makes the accuracy of students tend to the theoretical average when choosing randomly.

The test set had a size of 50 questions, each of them with 1 correct answer, and 3 distractors. From those 50 questions, the questions that had 2 ground truth distractors were filtered, and then from the remaining, 40 questions were randomly selected. After selecting the 40 questions, 2 different tests were built. Test 1 had the first 20 questions with distractors generated by BERT, and the second 20 questions with distractors generated by GPT. Test 2 had the opposite, the first 20 questions with distractors by BERT, and the second 20

questions with distractors generated by GPT. By doing this, we can compare the average results of the first 20 questions, GPT against BERT, and then compare the second 20 questions, BERT against GPT. It eliminates the factor that a certain set of questions would be harder or easier, and that the students from one group are more capable of responding to the questions than the other group.

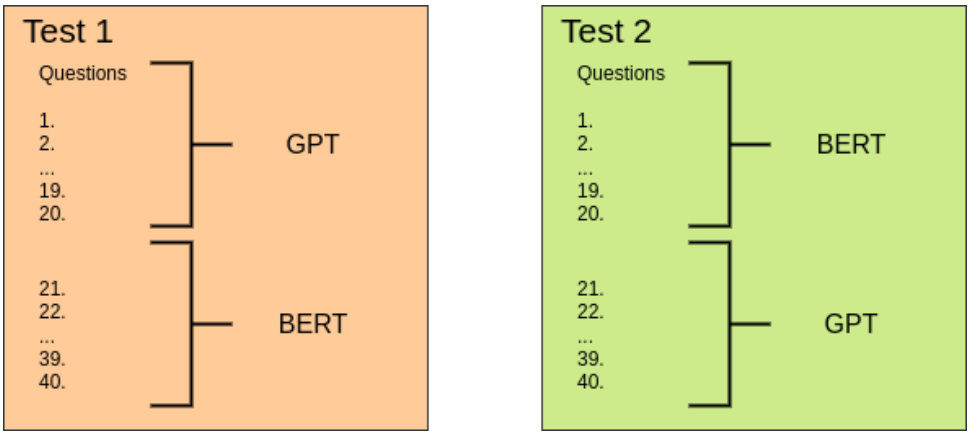

Figure 5.1: Setup of the Spanish-speaking tests

The respondents were provided with instructions that emphasized the importance of answering questions to the best of their knowledge without consulting external sources. While the instructions clarified that the test did not aim to evaluate their individual knowledge, they did not explicitly mention that the primary objective was to assess the automatically generated distractors. Following this, respondents answered two initial questions about their native language and age. This followed by responding to the 40 questions, presented in a random order. The test finished with a text box, in which respondents could freely write their final thoughts on the activity. The original instructions and the English translation is found in Appendix H.

The 2 tests were sent to Spanish speakers: Test 1 had 14 respondents and Test 2 had 15 respondents. The results were collected and analyzed. Table 5.2 shows the average accuracy of each of the tests in the corresponding part of the test.

Both exams, Test 1 and Test 2, have a lower accuracy for the GPT variant, as seen in Table 5.2. This means that the GPT variant through zero-shot learning,

Average accuracy of respondents on tests.	
Test 1 Questions 1-20 GPT	**31.79%** Accuracy
Test 2 Questions 1-20 BERT	45.99% Accuracy
Test 1 Questions 21-40 BERT	51.79% Accuracy
Test 2 Questions 21-40 GPT	**41.00%** Accuracy

Table 5.2: Average accuracy of respondents in their corresponding part of the test

is able to generate better distractors, as it is able to confuse students more by making them choose more at random, compared to the BERT variant.

Chapter 6

Discussion

This chapter discusses the results and draws distinctions between them. Section 6.1 compares the quantitative results in Spanish DG from this work with the results in Swedish DG from Kalpakchi and Boye [8]. Section 6.2 analyzes the results from the human evaluation and presents some specific examples of the generated distractors and the performance with the human evaluation.

6.1 Swedish and Spanish Distractor Generation

The test results obtained by fine-tuning a Spanish BERT, as seen in Table 5.1 can be compared to the results obtained in Kalpakchi and Boye [8], to explore the applicability of the method in different languages, and compare and contrast the differences between both results.

The first similarity that can be drawn is that in both, the results from this work and the results from Kalpakchi and Boye [8], the best performing model is the *u-PMLM*. This shows a general advantage between the *u-PMLM* and the *left-to-right* training scheme, that could be further explored with other languages, and other datasets.

Between the 2 approaches, the Spanish DG shows better performance than the Swedish DG. While the Distractor Recall in Kalpakchi and Boye [8] is 15.31%, the Distractor Recall for Spanish DG is of 35.62%. There is a significant difference in the metric *Any of the generated matches with a gold*

one, in which the Spanish DG is better, with 70% of the MCQs having at least one distractor which matches the ground truth ones. In Swedish DG, this corresponds to only 26.47%.

The Swedish DG and Spanish DG have similar performance in other metrics, such as having a low percentage of MCQs that contain the correct answer among the distractors, a low percentage of MCQs with faulty distractors (empty strings, same distractors, and repeated words), and a low percentage of MCQs that contain distractors from the training data.

While at first glance, the Spanish DG appears to perform better than the Swedish DG, there are several factors that influence the results. The first factor is the length of the texts. The average length of texts in Spanish DG (158 words) is smaller than the average length of texts in Swedish DG (380 words). During training time, a smaller portion of Spanish texts get truncated to the predetermined max length of texts (384 for Swedish, 350 for Spanish). This means that a higher proportion of data points in Spanish DG have the entire text available for the model to generate the distractors. This could be a reason for the distractor recall being higher in Spanish DG compared to Swedish DG. A higher ratio between tokens and average length of words, means less texts are truncated during training and inference, so the model has available more of the context to generate the correct distractors.

DG	Avg length of texts	Allowed tokens in the text	Ratio
Swedish	380 words	384 tokens	1.010
Spanish	158 words	350 tokens	2.215

Table 6.1: Text length distinction between Swedish and Spanish

The difficulty of the questions is also a factor that has an influence in the results. Around 40% of the questions in Spanish DG are under the category of Numbers and Proper Names, which are questions of a lower difficulty than Short Phrases. It is unknown what is the proportion of questions in these categories (Numbers, Proper Names, Short Phrases) for the Swedish DG, though it remains as a possible factor that affects the results. These are all factors that are independent of the target language, hence it is unfair to claim that the difference in results is due to the difference in the languages, rather than the differences in the dataset. A more thorough analysis of the differences

between the results of DG in different languages and datasets remains as future work.

6.2 GPT and BERT Human Evaluation

As previously mentioned, an accurate way to evaluate distractors is by doing a human evaluation and asking people to answer questions with the generated distractors. The real performance of distractors is evaluated on a real testing scenario because the purpose of having good distractors is to be able to confuse students into choosing the incorrect answer. Section 5.2 describes the evaluation performed on a test group of Spanish-speaking people, in which people were shown questions with distractors generated by BERT and GPT, and they were asked to answer those questions. The assumption is that the model that is better at generating distractors is the model that makes the students have less accuracy and forces students to choose at random. Only a handful of questions will be presented here, since the analysis of each individual question results in a more extensive work.

Table 5.2 shows the average accuracy of the respondents of each test, on each part of the test, the first 20 and the last 20 questions. Table 5.2 shows that the questions with GPT generated distractors have a lower accuracy than the questions with BERT generated distractors. This argument can be used to claim that GPT DG outperforms BERT DG.

To gain further insights, one can analyze the individual questions, their generated distractors, and the average response rate for the questions to see the differences between both approaches. By analyzing individual questions, intuition can be built as to why one model is better than the other one when evaluating on students, and why the results from Table 5.2 were obtained. It is important to note that the number of evaluated students is low, 14 students for Test 1 and 15 students for Test 2. So while the average choice rate of questions can lead to intuition on the results, in order to assure concluding results and statistical significance, a bigger population is needed to be tested upon.

Take as an example a question from the dataset, as shown in Table 6.2 in Spanish, and its translation to English in Table 6.3. The table shows the question at the top, as well as the correct answer and distractors of each method, BERT and GPT. Recall that a good set of distractors will lead to a student choosing at random. By looking at the generated distractors by GPT, the percentage of choice is seemingly random among the respondents, the correct answer being chosen 26.6% of the times, and the other distractors between 20%-26%. When looking at the distractors, they are all in the same

Spanish		
	¿Por qué serían recordados los Cachorros en el último cuarto de la temporada?	Percentage Choice
GPT	haber perdido 17 juegos notables en la clasificación (correct)	26.66%
	haber lanzado un juego sin hits el 19 de agosto	26.66%
	haber construido una ventaja sustancial en la División Este de la Liga Nacional	26.66%
	haber perdido el juego final de una serie en Cincinnati	20%
BERT	haber perdido 17 juegos notables en la clasificación (correct)	79%
	jugar los resurgentes K Jr.	7%
	los resurgentes Piratas de Pittsburgh	7%
	jugar una serie en Cincinnati	7%

Table 6.2: Example Question 1 for BERT and GPT - Spanish (Original)

English		
	Why would the Cubs be remembered the last quarter of the season?	Percentage Choice
GPT	to have lost 17 notable games in the classification (correct)	26.66%
	to have thrown a game without hits the 19th of august	26.66%
	to have built a substantial advantage in the East Division of the National League	26.66%
	to have lost the final game of a series in Cincinnati	20%
BERT	to have lost 17 notable games in the classification (correct)	79%
	playing the resurgents K Jr.	7%
	the resurgents Pittsburgh Pirates	7%
	playing a series in Cincinnati	7%

Table 6.3: Example Question 1 for BERT and GPT - English (Translation)

grammatical form as the correct answer, as they all start with the verb ”haber” (to have), they are all similar in length to the correct answer, and the semantic content of the distractors are similar to the correct answer too. This is an example of effective generation of distractors, as they led students to choose at random. By contrast, the BERT approach is highly skewed against the correct answer, hinting that the generated distractors were not of good quality. They are all shorter and different in verb tense to the correct answer. These factors

may have led to the distractors being ineffective.

Table 6.4 and Table 6.5 show another example from the study, in Spanish and its English translation, respectively. Both approaches seem to have good distractors. Distractors from both approaches are semantically similar to the correct answer, and of similar length. The only thing to note was the last distractor generated by BERT. In Spanish, pollen is in singular form, while the generated distractor generates ”los polen”, which is the plural form. A correctly generated distractor would have been ”el polen”. The fact that it is in an incorrect form, makes it a bad distractor. Even though it is a wrong distractor, the percentage of choice does not reflect any significant effect given a badly generated distractors, as still 13.33% of respondents of that test chose this answer, even if it was grammatically incorrect. Again, a bigger population participating in the study is needed, and could possibly prove the effect of this distractor in the accuracy of the respondents.

A last example can be presented in which both approaches have an accurate generation of distractors in Table 6.6 and Table 6.7, in original Spanish and its English translation, respectively. In this example, both BERT DG and GPT DG generate distractors that are similar in length to the correct answer, similarly semantically, and no grammatical errors. The distribution of choices seems evenly distributed, with the correct answer for both approaches being chosen 27% and 29% of the times, for GPT and BERT, respectively. They both even generate the same distractor (”compression techniques”). This example is to show how, even though there are examples where GPT seems to have better distractors than BERT, there are still examples where they both perform well.

All the presented examples are examples of Short Phrases, and for future work, a more extense analysis can be made given the type of question (Proper Name, Numbers, Short Phrase), and the outcome of choice of respondants.

Another difference between GPT DG and BERT DG was that GPT DG never generated the correct answer as a distractor, while BERT DG had a few data points in which the answer was generated as a distractor. There were 2 instances in the test set for BERT DG, in which the correct answer was among the options. In one question, BERT generated the correct answer as a distractor once, leaving 2 same correct answers to choose from, and another question where it generated the correct answer as a distractor twice, leaving

Spanish		
	¿De qué es la aparición repentina que se explica por la genética de islas?	Percentage Choice
GPT	las plantas con flores (correct)	29%
	adaptaciones radicales	50%
	una avispa hipótetica	14%
	las abejas	7%
BERT	las plantas con flores (correct)	40%
	las abejas	33.33%
	las avispas	13.33%
	los polen	13.33%

Table 6.4: Example Question 2 for BERT and GPT - Spanish (Original)

English		
	What suddenly appeared that is explained by the island's genetics?	Percentage Choice
GPT	the plants with flowers (correct)	29%
	radical adaptations	50%
	a hypothetical wasp	14%
	the bees	7%
BERT	the plants with flowers (correct)	40%
	the bees	33.33%
	the wasps	13.33%
	the pollen	13.33%

Table 6.5: Example Question 2 for BERT and GPT - English (Translation)

3 same correct answers to choose from. It was expected to see differences in choice percentages between GPT and BERT in these 2 specific questions, but no apparent effect was seen in the choices, which could be correctly analyzed with a bigger population sample of the study.

In general, the distractors generated by GPT are more grammatically and semantically similar to the correct answer, of similar length, appear to confuse students more in the performed test.

Spanish		
	¿Qué característica de LaserDiscs lo diferencia del DVD?	Percentage Choice
GPT	video analógico (correct)	29%
	proceso Macroblocking de vídeo	29%
	técnicas de compresión	28%
	codificadores patentados asistidos por humanos	14%
BERT	video analógico (correct)	27%
	técnicas de compresión	53%
	completamente digital	13%
	marcado de contraste	7%

Table 6.6: Example Question 3 for BERT and GPT - Spanish (Original)

English		
	What chararcteristic of LaserDiscs differentiates it from DVD?	Percentage Choice
GPT	analog video (correct)	29%
	Macroblocking video process	29%
	compression techniques	28%
	patented codifiers assisted by humans	14%
BERT	analog video (correct)	27%
	compression techniques	53%
	completely digital	13%
	contrast marking	7%

Table 6.7: Example Question 3 for BERT and GPT - English (Translation)

It might not seem as a surprise that the GPT model outperforms the fine-tuned BERT model, as the model used was GPT-4, which is current state-of-the-art at language generation, and BERT is a relatively old, and smaller (110 million parameters) LLM. Nevertheless, it is important to explore the different ways one can approach a task, and see the advantages and disadvantages of the models. While GPT DG seems to outperform BERT DG, BERT DG is a much smaller and versatile model that can also successfully perform DG using considerably less resources than the current state of the art GPT-4 language completion model.

Chapter 7

Conclusions and Future work

7.1 Conclusions

This work explored the language task of DG, which aims to generate wrong, but plausible options for MCQ datasets. The target dataset is the newly built and openly available *SQuAD-es-dist* dataset, a Spanish MCQ dataset with texts, questions, correct answers, and distractors.

The original objectives of the work were 3:

1. Modify/Adapt an existing MCQ dataset in Spanish to serve as a DG dataset, with all its contents correctly labelled ($T, Q, A, D_1, D_2, ..., D_N$)
2. Fine-tune and apply the chosen methodology for a Spanish BERT model and a GPT model for the task of DG.
3. Evaluate the generated distractors on Native Spanish speakers to analyze their usability as good distractors.

It can be concluded that all 3 objectives were achieved. The construction of *SQuAD-es-dist* was carried out successfully, after searching for base datasets, and a meticulous tagging and modification of a chosen base dataset. Once *SQuAD-es-dist* was built, the second objective was achieved by performing automatic DG. DG was made by first fine-tuning a BERT model for the desired task with the *SQuAD-es-dist* dataset. A Spanish BERT model was fine-tuned in 2 training schemes, the *left-to-right* and *u-PMLM* scheme. After performing DG with BERT, the state-of-the-art GPT model (GPT-4) was used to generate distractors in a zero-shot fashion. This second goal aligns with the motivation

of performing DG in a language other than the previously explored by other works (Swedish [8] and English [7]), to a new language.

The third objective was achieved by generating distractors on the test set with both, BERT and GPT, to later perform a human study, in which Spanish-speaking volunteers were asked to answer the questions with the generated distractors. The objective was to see the performance of the generated distractors by analyzing the percentage response rate of such questions.

A conclusion can be drawn that both models achieve successful DG, but further analysis is needed to conclude which one is truly better, though the initial results show a better performance of the GPT DG model. This work also does not achieve enough results to claim that DG can be performed 100% automatically. At this point of the application, the generated distractors need to be validated by a teacher once generated. Although not a proven fully automatized process, LLM and the methods in this study can be of great aid in the process of DG and can help teachers in the future by reducing the time spent performing exam creation and DG.

To the knowledge of the author, this work is the first of its kind in the chosen language, Spanish. The author hopes this work is of value to the Spanish-speaking community interested in the topics spoken in this work, and hopes it ignites other research works in Spanish.

7.2 Future work

There remains an extense amount of directions that this work could continue in the future. For starters, this work is specific to one dataset in one language. It would be of great value to continue exploring different datasets and new languages, and analyze the possible differences between them.

The tagging process was a limitation, since all the questions from *SQuAD-es-dist* were tagged by the author of this work. Hence, the number of data points tagged was small. For future work, a bigger dataset might improve the generalization capabilities of the model, and improve quantitative metrics such as Distractor Recall, and also distractor plausibility when evaluating on human speakers. Tagging remains as the most intensive time-consuming task, therefore collaboration would be needed to have more resources.

The created dataset *SQuAD-es-dist*, comes from the dataset *SQuAD-es* [10], that is an automatically translated version of *SQuAD* [33]. The answers are always in the text and the difficulty of answering the questions is not high. A second dataset was found, ReCoRES [35], that contains higher difficulty questions, since critical thinking is required to answer them. It remains as future work, using a different dataset such as ReCoRES, in order to also test the ability to create distractors for a dataset of higher difficulty. For this, both approaches BERT and GPT can be used, although GPT is most likely to outperform BERT, given that critical thinking as a skill is introduced.

When focusing only on BERT, there exists future work to test the different pretrained Spanish models that have been released. This work used BETO [11], but there exists a wide variety of Spanish pre-trained BERT models, such as BERTIN [46]. It would be of value to test the performance of the task when changing the different pre-trained models.

A wider and more thorough focus on the human evaluation is future work as well. The total number of participants in the human evaluation was 29, 14 for Test 1 and 15 for Test 2. It is impossible to control factors such as people's focus when doing the study, how close they follow instructions, if they cheated, their capacity to answer questions without context, and just their general interest in participating in the study. To counteract all the possible random factors, a bigger size on the test population is recommended, and hence it remains as future work, to do a more extense human evaluation. For this work, the 29 volunteers were the most participants that could be recruited given the available time resources for this work. With more participants, a statistical significant result as to which approach is better at generating distractors would be obtained.

When constructing the *SQuAD-es-dist* dataset, there was a special focus on maintaining the proportion of questions as explained in Section 3.2.2, to ensure that the questions were of varied difficulty, and that it contained enough in all categories, Numbers, Proper Names, and Short Phrases. It remains as future work to make the evaluation and analysis by segmenting given the type of question according to its category, in order to pinpoint possible flaws in the tagging process, and also to evaluate a more fine-grained source of which type of distractors make students fail the most.

It is important to note, that at the time of establishing the topic

of this work, the recent breakthroughs of ChatGPT [15] and GPT-4 [6] had not been released yet, and a myriad of other LLMs ever since (LlaMA, Claude, PaLM). Since these models have outperformed any past fine-tuned models, it is part of possible future work to drop any fine-tuned model (BERT) and focus on zero-shot or few-shot DG methods, and perform analysis on these bigger and newer models only.

7.3 Reflections

This work is a pristine example of how fast-changing the areas of Machine Learning, NLP, and LLMs are in the more recent years. Throughout the duration of this work, countless new language models were released, and several benchmarks were broken by every consecutive model. It will be interesting to see where the field is in a couple of months, and years.

This work is not only subject to the areas of Machine Learning and NLP, but it was interesting to research on the ares of education, reading understanding, language, and also sociology, when you involve the human evaluation.

There were a lot of small parts to this work, that add up nicely to tell a compelling story. Not all parts were enjoyable, specifically the tagging process was arduous, mentally tiring, and crushed my motivation. But seeing things in retrospective, it was a beautiful struggle, that finished with an honest piece of work.

This work marks a very long effort, that started in March, 2023 and finishes in November of the same year. There was a lot of development of different skills in the area of research and science, but also a lot of personal learnings in patience and compassion.

References

[1] Marie Tarrant and James Ware. "A Framework for Improving the Quality of Multiple-Choice Assessments." In: *Nurse educator* 37 (May 2012), pp. 98–104. DOI: 10.1097/NNE.0b013e31825041d0.

[2] Thomas M. Haladyna, Steven M. Downing, and Michael C. Rodriguez. "A Review of Multiple-Choice Item-Writing Guidelines for Classroom Assessment." In: *Applied Measurement in Education* 15.3 (2002), pp. 309–333. DOI: 10.1207/S15324818AME1503_5. eprint: https://doi.org/10.1207/S15324818AME1503_5. URL: https://doi.org/10.1207/S15324818AME1503_5.

[3] Jacob Devlin et al. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." In: *CoRR* abs/1810.04805 (2018). arXiv: 1810.04805. URL: http://arxiv.org/abs/1810.04805.

[4] Colin Raffel et al. "Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer." In: *CoRR* abs/1910.10683 (2019). arXiv: 1910.10683. URL: http://arxiv.org/abs/1910.10683.

[5] Alec Radford et al. "Improving language understanding by generative pre-training." In: (2018).

[6] OpenAI. *GPT-4 Technical Report*. 2023. arXiv: 2303.08774 [cs.CL].

[7] Zhaopeng Qiu, Xian Wu, and Wei Fan. "Automatic Distractor Generation for Multiple Choice Questions in Standard Tests." In: *CoRR* abs/2011.13100 (2020). arXiv: 2011.13100. URL: https://arxiv.org/abs/2011.13100.

[8] Dmytro Kalpakchi and Johan Boye. “BERT-based distractor generation for Swedish reading comprehension questions using a small-scale dataset.” In: *CoRR* abs/2108.03973 (2021). arXiv: 2108.03973. URL: https://arxiv.org/abs/2108.03973.

[9] Martin Malmsten, Love Börjeson, and Chris Haffenden. “Playing with Words at the National Library of Sweden - Making a Swedish BERT.” In: *CoRR* abs/2007.01658 (2020). arXiv: 2007.01658. URL: https://arxiv.org/abs/2007.01658.

[10] Casimiro Pio Carrino, Marta R. Costa-jussà, and José A. R. Fonollosa. “Automatic Spanish Translation of the SQuAD Dataset for Multilingual Question Answering.” In: *CoRR* abs/1912.05200 (2019). arXiv: 1912.05200. URL: http://arxiv.org/abs/1912.05200.

[11] José Cañete et al. “Spanish Pre-Trained BERT Model and Evaluation Data.” In: *PML4DC at ICLR 2020*. 2020.

[12] Alec Radford et al. *Language models are unsupervised multitask learners*. 2019. URL: https://www.semanticscholar.org/paper/Language-Models-are-Unsupervised-Multitask-Learners-Radford-Wu/9405cc0d6169988371b2755e573cc28650d14dfe (visited on 01/06/2023).

[13] Tom B. Brown et al. *Language Models are Few-Shot Learners*. 2020. arXiv: 2005.14165 [cs.CL].

[14] EdWeek Research Center. *1st Annual Merrimack College Teacher Survey: 2022 Results*. Merrimack College. (Accessed: 2023/07/01). 2022. URL: https://www.edweek.org/products/todays-teachers-are-deeply-disillusioned-survey-data-confirms.

[15] *ChatGPT*. https://chat.openai.com/.

[16] Dmytro Kalpakchi and Johan Boye. “Textinator: an Internationalized Tool for Annotation and Human Evaluation in Natural Language Processing and Generation.” In: *Proceedings of the Thirteenth Language Resources and Evaluation Conference*. Marseille, France: European Language Resources Association, June 2022, pp. 856–866. URL: https://aclanthology.org/2022.lrec-1.90.

[17] Hugging Face. *Hugging Face*. Available at https://huggingface.co/ (2023).

[18] Philip B. Gough and William E. Tunmer. “Decoding, Reading, and Reading Disability.” In: *Remedial and Special Education* 7 (1986), pp. 10–6.

[19] Arthur Firkins. “The Four Resources Model: A Useful Framework for Second Language Teaching in a Military Context.” In: Jan. 2015.

[20] Nell Duke and P. Pearson. “Effective Practices for Developing Reading Comprehension.” In: *What research has to say about reading instruction* 3 (Jan. 2002). DOI: 10.1598/0872071774.10.

[21] Rashmi Vyas and Avinash Supe. “Multiple choice questions: A literature review on the optimal number of options.” In: *The National medical journal of India* 21 (Nov. 2007), pp. 130–3.

[22] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. Adaptive computation and machine learning. MIT Press, 2016. ISBN: 9780262035613. URL: https://books.google.co.in/books?id=Np9SDQAAQBAJ.

[23] Daniel Jurafsky and James H. Martin. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. 1st. USA: Prentice Hall PTR, 2000. ISBN: 0130950696.

[24] Ashish Vaswani et al. “Attention Is All You Need.” In: *CoRR* abs/1706.03762 (2017). arXiv: 1706.03762. URL: http://arxiv.org/abs/1706.03762.

[25] Kyunghyun Cho et al. “Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation.” In: *CoRR* abs/1406.1078 (2014). arXiv: 1406.1078. URL: http://arxiv.org/abs/1406.1078.

[26] Alex Graves. “Sequence Transduction with Recurrent Neural Networks.” In: *CoRR* abs/1211.3711 (2012). arXiv: 1211.3711. URL: http://arxiv.org/abs/1211.3711.

[27] Razvan Pascanu, Tomás Mikolov, and Yoshua Bengio. “Understanding the exploding gradient problem.” In: *CoRR* abs/1211.5063 (2012). arXiv: 1211.5063. URL: http://arxiv.org/abs/1211.5063.

[28] Dzmitry Bahdanau, Kyunghyun Cho, and Y. Bengio. “Neural Machine Translation by Jointly Learning to Align and Translate.” In: *ArXiv* 1409 (Sept. 2014).

[29] Minh-Thang Luong, Hieu Pham, and Christopher D. Manning. "Effective Approaches to Attention-based Neural Machine Translation." In: *CoRR* abs/1508.04025 (2015). arXiv: 1508.04025. URL: http://arxiv.org/abs/1508.04025.

[30] Lewis Tunstall, Leandro von Werra, and Thomas Wolf. *Natural Language Processing with Transformers: Building Language Applications with Hugging Face*. O'Reilly Media, Incorporated, 2022. ISBN: 1098103246. URL: https://books.google.ch/books?id=7hhyzgEACAAJ.

[31] OpenAI. *OpenAI*. Available at https://openai.com/ (2023).

[32] Guokun Lai et al. "RACE: Large-scale ReAding Comprehension Dataset From Examinations." In: *CoRR* abs/1704.04683 (2017). arXiv: 1704.04683. URL: http://arxiv.org/abs/1704.04683.

[33] Pranav Rajpurkar et al. "SQuAD: 100, 000+ Questions for Machine Comprehension of Text." In: *CoRR* abs/1606.05250 (2016). arXiv: 1606.05250. URL: http://arxiv.org/abs/1606.05250.

[34] Mikel Artetxe, Sebastian Ruder, and Dani Yogatama. "On the Cross-lingual Transferability of Monolingual Representations." In: *CoRR* abs/1910.11856 (2019). arXiv: 1910.11856. URL: http://arxiv.org/abs/1910.11856.

[35] Marco A. Sobrevilla Cabezudo et al. *Overview of ReCoRES at IberLEF 2022: Reading Comprehension and Reasoning Explanation for Spanish*. Sept. 2022.

[36] Yuni Susanti et al. "Automatic distractor generation for multiple-choice English vocabulary questions." In: *Research and Practice in Technology Enhanced Learning* 13 (Oct. 2018). URL: https://rptel.apsce.net/index.php/RPTEL/article/view/2018-13015.

[37] Xiaorui Zhou, Senlin Luo, and Yunfang Wu. "Co-Attention Hierarchical Network: Generating Coherent Long Distractors for Reading Comprehension." In: *CoRR* abs/1911.08648 (2019). arXiv: 1911.08648. URL: http://arxiv.org/abs/1911.08648.

[38] Jeroen Offerijns, Suzan Verberne, and Tessa Verhoef. "Better Distractions: Transformer-based Distractor Generation and Multiple Choice Question Filtering." In: *CoRR* abs/2010.09598 (2020). arXiv: 2010.09598. URL: https://arxiv.org/abs/2010.09598.

[39] Victor Sanh et al. “DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter.” In: *CoRR* abs/1910.01108 (2019). arXiv: 1910.01108. URL: http://arxiv.org/abs/1910.01108.

[40] Ho-Lam Chung, Ying-Hong Chan, and Yao-Chung Fan. “A BERT-based Distractor Generation Scheme with Multi-tasking and Negative Answer Training Strategies.” In: *CoRR* abs/2010.05384 (2020). arXiv: 2010.05384. URL: https://arxiv.org/abs/2010.05384.

[41] Yi Liao, Xin Jiang, and Qun Liu. “Probabilistically Masked Language Model Capable of Autoregressive Generation in Arbitrary Word Order.” In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, July 2020, pp. 263–274. DOI: 10.18653/v1/2020.acl-main.24. URL: https://aclanthology.org/2020.acl-main.24.

[42] Dmytro Kalpakchi and Johan Boye. “Quasi: a synthetic Question-Answering dataset in Swedish using GPT-3 and zero-shot learning.” In: *Proceedings of the 24th Nordic Conference on Computational Linguistics (NoDaLiDa)*. Tórshavn, Faroe Islands: University of Tartu Library, May 2023, pp. 477–491. URL: https://aclanthology.org/2023.nodalida-1.48.

[43] OpenAI. *API Reference Guide - ChatCompletion*. Available at https://platform.openai.com/docs/guides/text-generation (2023).

[44] OpenAI. *Prompt Engineering*. Available at https://platform.openai.com/docs/guides/prompt-engineering (2023).

[45] Kishore Papineni et al. “BLEU: A Method for Automatic Evaluation of Machine Translation.” In: *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*. ACL ’02. Philadelphia, Pennsylvania: Association for Computational Linguistics, 2002, pp. 311–318. DOI: 10.3115/1073083.1073135. URL: https://doi.org/10.3115/1073083.1073135.

[46] Javier de la Rosa et al. *BERTIN: Efficient Pre-Training of a Spanish Language Model using Perplexity Sampling*. 2022. arXiv: 2207.06814 [cs.CL].

Appendix A

MCQ Guidelines

The following are the 31 guidelines from Haladyna, Downing, and Rodriguez [2]

A.1 A revised Taxonomy of Multiple-Choice (MC) Item-Writing Guidelines

Content Concerns

1. Every item should reflect specific content and a single specific mental behavior, as called for in test specifications (two-way grid, test blueprint).
2. Base each item on important content to learn; avoid trivial content.
3. Use novel material to test higher level learning. Paraphrase textbook language or language used during instruction when used in a test item to avoid testing for simply recall.
4. Keep the content of each item independent from content of other items on the test.
5. Avoid over specific and over general content when writing MC items.
6. Avoid opinion-based items.
7. Avoid trick items.
8. Keep vocabulary simple for the group of students being tested.

Formatting Concerns

9. Use the question, completion, and best answer versions of the conventional MC, the alternate choice, true-false (TF), multiple true-false (MTF),

matching, and the context-dependent item and item set formats, but AVOID the complex MC (Type K) format.

10. Format the item vertically instead of horizontally.

Style Concerns

11. Edit and proof items.
12. Use correct grammar, punctuation, capitalization, and spelling.
13. Minimize the amount of reading in each item.

Writing the Stem

14. Ensure that the directions in the stem are very clear.
15. Include the central idea in the stem instead of the choices.
16. Avoid window dressing (excessive verbiage).
17. Word the stem positively, avoid negatives such as NOT or EXCEPT. If negative words are used, use the word cautiously and always ensure that the word appears capitalized and boldface.

Writing the Choices

18. Develop as many effective choices as you can, but research suggests three is adequate.
19. Make sure that only one of these choices is the right answer.
20. Vary the location of the right answer according to the number of choices.
21. Place choices in logical or numerical order.
22. Keep choices independent; choices should not be overlapping.
23. Keep choices homogeneous in content and grammatical structure.
24. Keep the length of choices about equal.
25. *None-of-the-above* should be used carefully.
26. Avoid *All-of-the-above*.
27. Phrase choices positively; avoid negatives such as NOT.
28. Avoid giving clues to the right answer, such as

    (a) Specific determiners including always, never, completely, and absolutely.

(b) Clang associations, choices identical to or resembling words in the stem.
(c) Grammatical inconsistencies that cue the test-taker to the correct choice.
(d) Conspicuous correct choice.
(e) Pairs or triplets of options that clue the test-taker to the correct choice.
(f) Blatantly absurd, ridiculous options.

29. Make all distractors plausible.
30. Use typical errors of students to write your distractors.
31. Use humor if it is compatible with the teacher and the learning environment.

Appendix B

Textinator Screenshot

The following is an example of a tagged data point using the Textinator [16] application.

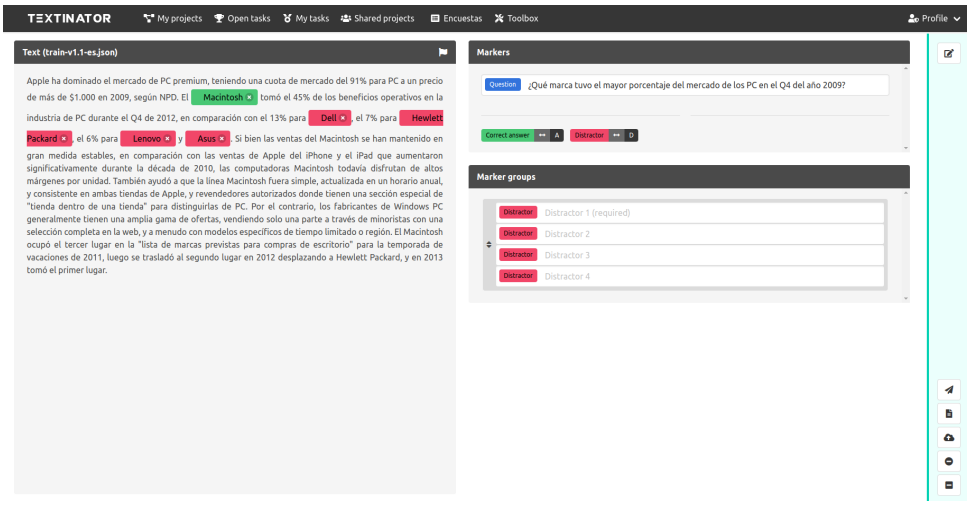

Figure B.1: Tagged data point from Textinator app

Appendix C

ChatGPT prompt for tagging

The following prompt was used in ChatGPT to aid the tagger in the process of tagging:

> Given a text, I want to generate a question, a correct answer, and 2 or 3 several distractor answers. A distractor answer is a wrong but plausible answer that is meant to confuse a student or user. When reading the question, the correct answer and the distractor answers, the text must be necessary in order to answer the question correctly. I will give you a text, and you should try to generate a question, with its correct answers and 2 or 3 distractor answers. A requirement is that the correct answer and the distractor answers, must be found in the text, you can't make up the distractor answers, they need to be found inside the text. The correct and distractor answers can be short phrases (1-4 words) and also long phrases. Try to avoid answers and distractors that are only numbers, or only proper names. Of course, numbers and proper names and can be included in the answer, but try to have them included in longer phrases. It is better if the questions answer the questions "What?", "Why?", and "How?", rather than "When?" and "Who?". The text will be in Spanish. You will provide with 3 different sets of questions, correct answer and distractors. Do you understand the task?

Appendix D

Tagging Examples

All examples are from training set. They are presented as a JSON object.

D.1 Number Example

D.1.1 Spanish


```
{
  "context": "El␣ingreso␣promedio␣neto␣de␣los␣hogares␣(
     ↪ édespus␣de␣las␣contribuciones␣sociales,␣de␣
     ↪ pensiones␣y␣de␣seguros␣de␣salud)␣en␣íPars␣fue␣de
     ↪ ␣36.085␣euros␣para␣2011.␣Su␣rango␣oscila␣entre␣
     ↪ 22.095␣€en␣el␣XIX␣Distrito␣a␣82.449␣€en␣el␣VII␣
     ↪ Distrito.␣El␣ingreso␣medio␣imponible␣para␣2011␣
     ↪ fue␣de␣alrededor␣de␣25.000␣euros␣en␣íPars␣y␣
     ↪ 22.200␣euros␣para␣Île-de-France.␣En␣étrminos␣
     ↪ generales,␣los␣ingresos␣son␣mayores␣en␣la␣parte␣
     ↪ occidental␣de␣la␣ciudad␣y␣en␣los␣suburbios␣
     ↪ occidentales␣que␣en␣las␣partes␣norte␣y␣este␣de␣
     ↪ la␣zona␣urbana.␣El␣desempleo␣se␣óestim␣en␣un␣
     ↪ 8,2%␣en␣la␣ciudad␣de␣íPars␣y␣un␣8,8%␣en␣la␣
     ↪ óregin␣de␣Île-de-France␣en␣el␣primer␣trimestre␣
     ↪ de␣2015.␣óOscil␣entre␣el␣7,6%␣en␣el␣rico␣
     ↪ departamento␣de␣Essonne␣y␣el␣13,1%␣en␣el␣
     ↪ departamento␣de␣Seine-Saint-Denis,␣donde␣viven␣
     ↪ muchos␣inmigrantes␣recientes.",
  "question": "¿áCul␣es␣el␣ingreso␣promedio␣en␣el␣
     ↪ Distrito␣VII?",
```

```
  "choices": [
    {
      "text": "82.449",
      "start": 208,
      "end": 214,
      "type": "correct answer",
      "extra": null
    },
    {
      "text": "25.000",
      "start": 293,
      "end": 299,
      "type": "distractor",
      "extra": null
    },
    {
      "text": "22.200",
      "start": 317,
      "end": 323,
      "type": "distractor",
      "extra": null
    },
    {
      "text": "22.095",
      "start": 179,
      "end": 185,
      "type": "distractor",
      "extra": null
    },
    {
      "text": "36.085",
      "start": 133,
      "end": 139,
      "type": "distractor",
      "extra": null
    }
  ]
}
```


D.1.2 English

```
{
  "context": "The␣average␣net␣household␣income␣(after␣
     ↪ social,␣pension␣and␣health␣insurance␣
     ↪ contributions)␣in␣Paris␣was␣36,085␣euros␣in␣
     ↪ 2011.␣It␣ranges␣from␣€22,095␣in␣the␣XIXth␣
     ↪ arrondissement␣to␣€82,449␣in␣the␣VIIth␣
     ↪ arrondissement.␣The␣average␣taxable␣income␣for␣
     ↪ 2011␣was␣around␣25,000␣euros␣in␣Paris␣and␣22,200
     ↪ ␣euros␣for␣Ile-de-France.␣Generally␣speaking,␣
     ↪ incomes␣are␣higher␣in␣the␣western␣part␣of␣the␣
     ↪ city␣and␣in␣the␣western␣suburbs␣than␣in␣the␣
     ↪ northern␣and␣eastern␣parts␣of␣the␣urban␣area.␣
     ↪ Unemployment␣was␣estimated␣at␣8.2%␣in␣the␣city␣
     ↪ of␣Paris␣and␣8.8%␣in␣the␣Ile-de-France␣region␣in
     ↪ ␣the␣first␣quarter␣of␣2015.␣It␣ranged␣from␣7.6%␣
     ↪ in␣the␣wealthy␣Essonne␣department␣to␣13.1%␣in␣
     ↪ the␣Seine-Saint-Denis␣department,␣where␣many␣
     ↪ recent␣immigrants␣live.",
  "question": "What␣is␣the␣average␣income␣in␣District␣
     ↪ VII?",
  "choices": [
    {
      "text": "82.449",
      "start": 208,
      "end": 214,
      "type": "correct␣answer",
      "extra": null
    },
    {
      "text": "25.000",
      "start": 293,
      "end": 299,
      "type": "distractor",
      "extra": null
    },
    {
      "text": "22.200",
      "start": 317,
      "end": 323,
      "type": "distractor",
      "extra": null
    },
```

```
    {
      "text": "22.095",
      "start": 179,
      "end": 185,
      "type": "distractor",
      "extra": null
    },
    {
      "text": "36.085",
      "start": 133,
      "end": 139,
      "type": "distractor",
      "extra": null
    }
  ]
}
```


D.2 Proper Name Example

D.2.1 Spanish


```
{
  "context": "San␣Juan␣se␣encuentra␣a␣lo␣largo␣de␣la␣
     ↪ costa␣del␣éOcano␣áAtlntico,␣en␣el␣noreste␣de␣la␣
     ↪ ípennsula␣de␣Avalon␣en␣el␣sureste␣de␣Terranova.␣
     ↪ La␣ciudad␣cubre␣un␣área␣de␣446,04␣km␣²␣y␣es␣la␣
     ↪ ciudad␣áms␣oriental␣de␣éAmrica␣del␣Norte,␣con␣
     ↪ óexclusin␣de␣Groenlandia.␣áEst␣475␣km␣áms␣cerca␣
     ↪ de␣Londres,␣Inglaterra␣que␣de␣Edmonton,␣Alberta.
     ↪ ␣La␣ciudad␣de␣San␣Juan␣se␣encuentra␣a␣una␣
     ↪ distancia␣por␣aire␣de␣3636␣ókilmetros␣de␣Lorient
     ↪ ,␣Francia,␣que␣se␣encuentra␣en␣una␣latitud␣casi␣
     ↪ éidntica␣a␣la␣del␣áAtlntico␣en␣la␣costa␣
     ↪ occidental␣francesa.␣La␣ciudad␣es␣la␣áms␣grande␣
     ↪ de␣la␣provincia␣y␣la␣segunda␣áms␣grande␣de␣las␣
     ↪ Provincias␣áAtlnticas␣édespus␣de␣Halifax,␣Nueva␣
     ↪ Escocia.␣Su␣centro␣se␣encuentra␣al␣oeste␣y␣al␣
     ↪ norte␣del␣puerto␣de␣San␣Juan,␣y␣el␣resto␣de␣la␣
     ↪ ciudad␣se␣expande␣desde␣el␣centro␣al␣norte,␣sur,
     ↪ ␣este␣y␣oeste.",
```

```
  "question": "¿áCul␣es␣la␣ciudad␣áms␣oriental␣de␣
     ↪ éAmrica␣del␣Norte?",
  "choices": [
    {
      "text": "San␣Juan",
      "start": 0,
      "end": 8,
      "type": "correct␣answer",
      "extra": null
    },
    {
      "text": "Edmonton",
      "start": 306,
      "end": 314,
      "type": "distractor",
      "extra": null
    },
    {
      "text": "Lorient",
      "start": 407,
      "end": 414,
      "type": "distractor",
      "extra": null
    },
    {
      "text": "Halifax",
      "start": 630,
      "end": 637,
      "type": "distractor",
      "extra": null
    }
  ]
}
```


D.2.2 English


```
{
  "context": "San␣Juan␣is␣located␣along␣the␣coast␣of␣the
     ↪ ␣Atlantic␣Ocean,␣in␣the␣northeast␣of␣the␣Avalon␣
     ↪ Peninsula␣in␣southeastern␣Newfoundland.␣The␣city
     ↪ ␣covers␣an␣area␣of␣446.04␣²km␣and␣is␣the␣
```

```
    ↪ easternmost␣city␣in␣North␣America,␣excluding␣
    ↪ Greenland.␣It␣is␣475␣km␣closer␣to␣London,␣
    ↪ England␣than␣to␣Edmonton,␣Alberta.␣The␣city␣of␣
    ↪ San␣Juan␣is␣located␣at␣an␣air␣distance␣of␣3636␣
    ↪ km␣from␣Lorient,␣France,␣which␣is␣at␣almost␣
    ↪ identical␣latitude␣to␣the␣Atlantic␣on␣the␣French
    ↪ ␣west␣coast.␣The␣city␣is␣the␣largest␣in␣the␣
    ↪ province␣and␣the␣second␣largest␣in␣the␣Atlantic␣
    ↪ Provinces␣after␣Halifax,␣Nova␣Scotia.␣Its␣center
    ↪ ␣lies␣west␣and␣north␣of␣the␣port␣of␣San␣Juan,␣
    ↪ and␣the␣rest␣of␣the␣city␣expands␣from␣the␣center
    ↪ ␣to␣the␣north,␣south,␣east␣and␣west."
"question": "What␣is␣the␣easternmost␣city␣in␣North␣
    ↪ America?",
"choices": [
  {
    "text": "San␣Juan",
    "start": 0,
    "end": 8,
    "type": "correct␣answer",
    "extra": null
  },
  {
    "text": "Edmonton",
    "start": 306,
    "end": 314,
    "type": "distractor",
    "extra": null
  },
  {
    "text": "Lorient",
    "start": 407,
    "end": 414,
    "type": "distractor",
    "extra": null
  },
  {
    "text": "Halifax",
    "start": 630,
    "end": 637,
    "type": "distractor",
    "extra": null
```

```
      }
    ]
}
```

D.3 Short or Long Phrase Example

D.3.1 Spanish

```
{
  "context": "En␣abril␣de␣1945,␣China␣ya␣íhaba␣estado␣en
     ↪ ␣guerra␣con␣óJapn␣durante␣áms␣de␣siete␣ñaos.␣
     ↪ Ambas␣naciones␣estaban␣agotadas␣por␣ñaos␣de␣
     ↪ batallas,␣bombardeos␣y␣bloqueos.␣éDespus␣de␣las␣
     ↪ victorias␣japonesas␣en␣la␣óOperacin␣Ichi-Go,␣
     ↪ óJapn␣estaba␣perdiendo␣la␣batalla␣en␣Birmania␣y␣
     ↪ enfrentando␣constantes␣ataques␣de␣las␣fuerzas␣
     ↪ nacionalistas␣chinas␣y␣guerrilleros␣comunistas␣
     ↪ en␣el␣lado␣del␣ípas.␣El␣éejrcito␣éjapons␣ócomenz
     ↪ ␣los␣preparativos␣para␣la␣Batalla␣de␣Hunan␣
     ↪ Occidental␣en␣marzo␣de␣1945.␣Los␣japoneses␣
     ↪ movilizaron␣las␣divisiones␣34,␣47,␣64,␣68␣y␣116,
     ↪ ␣ías␣como␣la␣ª86␣Brigada␣Independiente,␣para␣un␣
     ↪ total␣de␣80.000␣hombres␣para␣apoderarse␣de␣los␣
     ↪ óaerdromos␣chinos␣y␣asegurar␣ferrocarriles␣en␣el
     ↪ ␣oeste␣de␣Hunan.␣En␣respuesta,␣el␣Consejo␣
     ↪ Militar␣Nacional␣chino␣óenvi␣al␣º4.␣éEjrcito␣del
     ↪ ␣Frente␣y␣al␣º10.␣y␣º27.␣Grupo␣de␣éEjrcitos␣con␣
     ↪ He␣Yingqin␣como␣comandante␣en␣jefe.␣Al␣mismo␣
     ↪ tiempo,␣ótransport␣por␣aire␣a␣todo␣el␣Nuevo␣º6.␣
     ↪ Cuerpo␣chino,␣un␣cuerpo␣equipado␣con␣
     ↪ estadounidenses␣y␣veteranos␣de␣la␣Fuerza␣
     ↪ Expedicionaria␣de␣Birmania,␣desde␣Kunming␣a␣
     ↪ Zhijiang.␣Las␣fuerzas␣chinas␣totalizaron␣110.000
     ↪ ␣hombres␣en␣20␣divisiones.␣Fueron␣apoyados␣por␣
     ↪ unos␣400␣aviones␣de␣las␣fuerzas␣éareas␣chinas␣y␣
     ↪ estadounidenses.␣Las␣fuerzas␣chinas␣lograron␣una
     ↪ ␣victoria␣decisiva␣y␣lanzaron␣un␣gran␣
     ↪ contraataque␣en␣esta␣ñcampaa.␣Al␣mismo␣tiempo,␣
     ↪ los␣chinos␣lograron␣repeler␣una␣ofensiva␣
     ↪ japonesa␣en␣Henan␣y␣Hubei.␣éDespus,␣las␣fuerzas␣
```

```
    ↪ chinas␣retomaron␣las␣provincias␣de␣Hunan␣y␣Hubei
    ↪ ␣en␣el␣sur␣de␣China.␣China␣ólanz␣una␣
    ↪ contraofensiva␣para␣retomar␣Guangxi,␣que␣fue␣la␣
    ↪ última␣gran␣fortaleza␣japonesa␣en␣el␣sur␣de␣
    ↪ China.␣En␣agosto␣de␣1945,␣las␣fuerzas␣chinas␣
    ↪ retomaron␣con␣éxito␣Guangxi.␣[Cita␣requerida]",
"question": "¿éQu␣hizo␣el␣Consejo␣Militar␣Nacional␣
    ↪ chino␣ante␣la␣ómovilizacin␣de␣los␣japoneses␣de␣
    ↪ 80.000␣hombres?",
"choices": [
  {
    "text": "óenvi␣al␣º4.␣éEjrcito␣del␣Frente␣y␣al␣
       ↪ º10.␣y␣º27.␣Grupo␣de␣éEjrcitos␣con␣He␣
       ↪ Yingqin␣como␣comandante␣en␣jefe",
    "start": 755,
    "end": 862,
    "type": "correct␣answer",
    "extra": null
  },
  {
    "text": "retomaron␣las␣provincias␣de␣Hunan␣y␣Hubei
       ↪ ␣en␣el␣sur␣de␣China",
    "start": 1415,
    "end": 1475,
    "type": "distractor",
    "extra": null
  },
  {
    "text": "ólanz␣una␣contraofensiva␣para␣retomar␣
       ↪ Guangxi,␣que␣fue␣la␣última␣gran␣fortaleza␣
       ↪ japonesa␣en␣el␣sur␣de␣China",
    "start": 1483,
    "end": 1590,
    "type": "distractor",
    "extra": null
  },
  {
    "text": "movilizaron␣las␣divisiones␣34,␣47,␣64,␣68
       ↪ ␣y␣116,␣ías␣como␣la␣ª86␣Brigada␣
       ↪ Independiente",
    "start": 499,
    "end": 585,
```

```
      "type": "distractor",
      "extra": null
    },
    {
      "text": "apoderarse␣de␣los␣óaerdromos␣chinos␣y␣
         ↪ asegurar␣ferrocarriles␣en␣el␣oeste␣de␣Hunan"
         ↪ ,
      "start": 624,
      "end": 705,
      "type": "distractor",
      "extra": null
    }
  ]
}
```

D.3.2 English

```
{
  "context": "By␣April␣1945,␣China␣had␣been␣at␣war␣with␣
     ↪ Japan␣for␣more␣than␣seven␣years.Both␣nations␣
     ↪ were␣exhausted␣by␣years␣of␣battles,␣bombardments
     ↪ ␣and␣blockades.After␣Japanese␣victories␣in␣
     ↪ Operation␣Ichi-Go,␣Japan␣was␣losing␣the␣battle␣
     ↪ in␣Burma␣and␣facing␣constant␣attacks␣from␣
     ↪ Chinese␣Nationalist␣forces␣and␣Communist␣
     ↪ guerrillas␣on␣the␣Chinese␣side.The␣Japanese␣army
     ↪ ␣began␣preparations␣for␣the␣Battle␣of␣Western␣
     ↪ Hunan␣in␣March␣1945.The␣Japanese␣mobilized␣the␣
     ↪ 34th,␣47th,␣64th,␣68th,␣and␣116th␣Divisions,␣as␣
     ↪ well␣as␣the␣86th␣Independent␣Brigade,␣for␣a␣
     ↪ total␣of␣80,000␣men␣to␣seize␣Chinese␣airfields␣
     ↪ and␣secure␣railroads␣in␣western␣Hunan.In␣
     ↪ response,␣the␣Chinese␣National␣Military␣Council␣
     ↪ sent␣the␣4th␣Front␣Army␣and␣the␣10th␣and␣27th␣
     ↪ Army␣Group␣with␣He␣Yingqin␣as␣commander-in-chief
     ↪ .␣At␣the␣same␣time,␣it␣airlifted␣the␣entire␣
     ↪ Chinese␣New␣6th␣Corps,␣a␣corps␣equipped␣with␣
     ↪ Americans␣and␣Burma␣Expeditionary␣Force␣veterans
     ↪ ,␣from␣Kunming␣to␣Zhijiang.␣The␣Chinese␣forces␣
     ↪ totaled␣110,000␣men␣in␣20␣divisions.␣They␣were␣
     ↪ supported␣by␣some␣400␣Chinese␣and␣U.S.␣air␣force
```

```
    ↪ ␣aircraft.␣The␣Chinese␣forces␣won␣a␣decisive␣
    ↪ victory␣and␣launched␣a␣major␣counterattack␣in␣
    ↪ this␣campaign.␣At␣the␣same␣time,␣the␣Chinese␣
    ↪ succeeded␣in␣repelling␣a␣Japanese␣offensive␣in␣
    ↪ Henan␣and␣Hubei.␣Chinese␣forces␣then␣retook␣
    ↪ Hunan␣and␣Hubei␣provinces␣in␣southern␣China.␣
    ↪ China␣launched␣a␣counteroffensive␣to␣retake␣
    ↪ Guangxi,␣which␣was␣the␣last␣major␣Japanese␣
    ↪ stronghold␣in␣southern␣China.␣In␣August␣1945,␣
    ↪ Chinese␣forces␣successfully␣retook␣Guangxi.␣[
    ↪ citation␣needed]",
"question": "What␣did␣the␣Chinese␣National␣Military␣
    ↪ Council␣do␣in␣the␣face␣of␣the␣Japanese␣
    ↪ mobilization␣of␣80,000␣men?",
"choices": [
  {
    "text": "sent␣the␣4th␣Front␣Army␣and␣the␣10th␣and␣
       ↪ 27th␣Army␣Group␣with␣He␣Yingqin␣as␣commander
       ↪ -in-chief",
    "start": 755,
    "end": 862,
    "type": "correct␣answer",
    "extra": null
  },
  {
    "text": "retook␣Hunan␣and␣Hubei␣provinces␣in␣
       ↪ southern␣China.",
    "start": 1415,
    "end": 1475,
    "type": "distractor",
    "extra": null
  },
  {
    "text": "launched␣a␣counteroffensive␣to␣retake␣
       ↪ Guangxi,␣which␣was␣the␣last␣major␣Japanese␣
       ↪ stronghold␣in␣southern␣China.",
    "start": 1483,
    "end": 1590,
    "type": "distractor",
    "extra": null
  },
  {
```

```
      "text": "mobilized␣the␣34th,␣47th,␣64th,␣68th,␣and
         ↪ ␣116th␣Divisions,␣as␣well␣as␣the␣86th␣
         ↪ Independent␣Brigade.",
      "start": 499,
      "end": 585,
      "type": "distractor",
      "extra": null
    },
    {
      "text": "seizing␣Chinese␣airfields␣and␣securing␣
         ↪ railroads␣in␣western␣Hunan",
      "start": 624,
      "end": 705,
      "type": "distractor",
      "extra": null
    }
  ]
}
```

Appendix E

Github Links

The original repository from Kalpakchi and Boye [8], that contains the dataset *SweQUAD-MC* and the DG methods using a pre-trained BERT are found in : https://github.com/dkalpakchi/SweQUAD-MC

The repository with the *SQuAD-es-dist* dataset and the GPT methods through zero-shot learning for DG can be found in : https://github.com/santroma1/SQuAD-es-dist

Appendix F

Results for Model Selection

F.1 Swedish Dataset

Autoregressive Model (left-to-right)

Iteration	12000	14000	16000	18000
Epoch	3.62	4.23	4.85	5.44
Total	126	126	126	126
Distractor recall	7.89%	6.39%	9.4%	9.02%
Any of the distractors matches with a gold one	14.29%	11.11%	16.67%	18.25%
(A) Any distractor is in its own context	47.62%	58.73%	65.08%	72.22%
The correct answer is among distractors	2.38%	2.38%	0.79%	2.38%
Any (but not all) distractors are the same	57.94%	47.62%	29.37%	26.19%
All generated distractors are the same	19.05%	10.32%	3.97%	5.56%
Any distractor contains repetitive words	0.79%	0.0%	0.79%	0.0%
Any distractor is an empty string	0.0%	0.0%	0.0%	0.0%
(C) Any distractor is from training data	26.98%	37.3%	40.48%	37.3%

Table F.1: Results on Development set for left-to-right model on Swedish dataset, experiment repeated from Kalpakchi and Boye [8]

u-PMLM Model

Iteration	12000	14000	16000
Epoch	4.30	5.02	5.74
Total	126	126	126
Distractor recall	15.41%	16.92%	18.05%
Any of the distractors matches with a gold one	30.16%	31.75%	34.13%
(A) Any distractor is in its own context	72.22%	69.05%	70.63%
The correct answer is among distractors	4.76%	2.38%	4.76%
Any (but not all) distractors are the same	13.49%	13.49%	11.9%
All generated distractors are the same	0.0%	0.0%	0.0%
Any distractor contains repetitive words	0.79%	1.59%	3.97%
Any distractor is an empty string	0.0%	0.0%	0.0%
(C) Any distractor is from training data	31.75%	34.92%	36.51%

Table F.2: Results on Development set for u-PMLM model on Swedish dataset, experiment repeated from Kalpakchi and Boye [8]

F.2 Spanish Dataset

Autoregressive Model (left-to-right)

Iteration	20000	22000	24000	26000
Epoch	4.29	4.71	5.14	5.57
Total	95	95	95	95
Distractor recall	22.95%	20.33%	27.54%	26.56%
Any of the distractors matches with a gold one	49.47%	47.37%	56.84%	54.74%
(A) Any distractor is in its own context	95.79%	88.42%	97.89%	96.84%
The correct answer is among distractors	3.16%	3.16%	4.21%	3.16%
Any (but not all) distractors are the same	4.21%	14.74%	5.26%	7.37%
All generated distractors are the same	3.16%	3.16%	1.05%	1.05%
Any distractor contains repetitive words	0.0%	0.0%	0.0%	0.0%
Any distractor is an empty string	0.0%	0.0%	0.0%	0.0%
(C) Any distractor is from training data	0.0%	0.0%	0.0%	0.0%

Table F.3: Results on Development set for left-to-right model on Spanish dataset

u-PMLM Model

Iteration	18000	20000	22000
Epoch	4.52	5.02	5.52
Total	95	95	95
Distractor recall	31.15%	31.48%	30.49%
Any of the distractors matches with a gold one	61.05%	60.0%	57.89%
(A) Any distractor is in its own context	90.53%	89.47%	91.58%
The correct answer is among distractors	5.26%	3.16%	3.16%
Any (but not all) distractors are the same	7.37%	5.26%	3.16%
All generated distractors are the same	3.16%	2.11%	0.0%
Any distractor contains repetitive words	1.05%	4.21%	5.26%
Any distractor is an empty string	0.0%	0.0%	0.0%
(C) Any distractor is from training data	0.0%	0.0%	0.0%

Table F.4: Results on Development set for u-PMLM model on Spanish dataset

Appendix G

Prompt for GPT DG

G.1 Spanish

The following prompt was sent as is for completion for Distractor Generation using GPT-4 in a zero-shot learning fashion.

Eres un maestro para estudiantes que están aprendiendo español. Te asignan la tarea de generar distractores para preguntas de opción múltiple, con el propósito de evaluar a tus estudiantes. La generación de distractores en el contexto de preguntas de opción múltiple es una tarea que con un texto, una pregunta, y una respuesta correcta, se busca generar las respuestas distractoras. Las respuestas distractoras son respuestas incorrectas a una pregunta, pero que tienen la capacidad de confundir a un estudiante, y que son sintácticamente similares a la respuesta correcta. Las respuestas distractoras deben ser gramáticamente similares a la respuesta correcta, por ejemplo, si la respuesta correcta empieza con una letra minúscula, las respuestas distractoras también deben empezar con una letra minúscula. Las respuestas distractoras serán mejores también si pertenecen al mismo grupo semántico que la respuesta correcta. Serás presentado con un texto, una pregunta, y la respuesta correcta, y tu tarea es generar 3 respuestas distractoras. Recuerda que debes generar buenas respuestas distractoras como las describí. Primero lee el texto. Segundo, lee la pregunta y la respuesta correcta. Y tercero, con base en el texto, la pregunta, y la respuesta correcta, extrae del texto las 3 respuestas distractoras. Las respuestas distractoras deben ser extraídas del texto, y las respuestas distractoras no deben ser explícitamente la respuesta correcta. Si haces las respuestas distractoras similares entre ellas, pero diferentes a la respuesta correcta, un estudiante podría intuir cuál es la respuesta correcta, al ver que hay una respuesta que es diferente en su forma a las demás. Genera las respuestas distractoras en un formato de lista de Python, y cada una debe empezar con letras minúsculas.
Texto: <TEXT>
Pregunta: <QUESTION>
Respuesta Correcta: <CORRECT ANSWER>

G.2 English

This is the English translation of the prompt input for zero-shot Distractor Generation using GPT-4.

You are a teacher for students learning Spanish. You are assigned the task of generating distractors for multiple choice questions, with the purpose of evaluating your students. The generation of distractors in the context of multiple choice questions is a task that, with a text, a question, and a correct answer, seeks to generate distractor answers. Distractor answers are incorrect answers to a question, but have the ability to confuse a student, and are syntactically similar to the correct answer. Distractor answers should be grammatically similar to the correct answer, e.g., if the correct answer begins with a lowercase letter, distractor answers should also begin with a lowercase letter. Distractor answers will also be better if they belong to the same semantic group as the correct answer. You will be presented with a text, a question, and the correct answer, and your task is to generate 3 distractor answers. Remember to generate good distractor responses as I described. First, read the text. Second, read the question and the correct answer. And third, based on the text, the question, and the correct answer, extract from the text the 3 distractor responses. The distractor answers must be extracted from the text, and the distractor answers must not explicitly be the correct answer. If you make the distractor answers similar to each other, but different from the correct answer, a student might intuit what the correct answer is, seeing that there is one answer that is different in form from the others. Generate the distractor answers in a Python list format, and each must begin with lowercase letters.
Text: <TEXT>
Question: <QUESTION>
Correct Answer: <CORRECT ANSWER>

Appendix H

Instructions for student evaluations

H.1 Spanish

The following text is the instructions in the evaluation form for Spanish-speaking students, to answer the questions with the distractors generated by both BERT and GPT-4.

> ¡Hola! Gracias por participar en esta actividad. A continuación te presentaré 40 preguntas de opción múltiple, y tu tarea es responder las preguntas con la mejor de tus habilidades. Las preguntas están basadas en un texto. Se necesita el texto asociado para responder correctamente a la pregunta. El texto asociado NO estará presente, así que tu tarea es responder la pregunta de opción múltiple sin el texto. Para muchas preguntas será muy difícil responder, ya que se necesita el texto para entender el contexto. Sin embargo, pon mucha atención a las opciones múltiples, que aunque no esté el texto asociado presente, puedes ser capaz de responder la pregunta correctamente. No te estamos evaluando a ti y la capacidad de una persona para responder las preguntas. ¡Haz tu mejor esfuerzo en responderla sin el texto, y sin buscar nada en el internet! Primero responde si el español es tu idioma nativo, y si tienes más de 18 años para empezar el estudio.

H.2 English

This is the English translation of the instructions to evaluate Spanish-speaking students.

Hello! Thank you for participating in this activity. Below I will present you with 40 multiple choice questions, and your task is to answer the questions to the best of your ability. The questions are based on a text. You need the associated text to answer the question correctly. The associated text will NOT be present, so your task is to answer the multiple choice question without the text. For many questions it will be very difficult to answer, as you need the text to understand the context. However, pay close attention to multiple choice, that even if the associated text is not present, you may be able to answer the question correctly. We are not testing you and a person's ability to answer the questions, so do your best to answer it without the text, and without looking anything up on the internet! First answer if Spanish is your native language, and if you are over 18 years old to begin the study.

TRITA-EECS-EX-2023:905

€€€€ For DIVA €€€€

{
”Author1”: { ”Last name”: ”Roman Avila”,
”First name”: ”Jorge Santiago”,
”Local User Id”: ”u1jy113v”,
”E-mail”: ”jsra2@kth.se”,
”organisation”: {”L1”: ”School of Electrical Engineering and Computer Science”,
}
},
”Cycle”: ”2”,
”Course code”: ”DA233X”,
”Credits”: ”30.0”,
”Degree1”: {”Educational program”: ”Master’s Programme, Machine Learning, 120 credits”
,”programcode”: ”TMAIM”
,”Degree”: ”Degree of Master of Science in Machine Learning”
,”subjectArea”: ”Computer Science and Engineering”
},
”Title”: {
”Main title”: ”Automatic Distractor Generation for Spanish Reading Comprehension Questions”,
”Subtitle”: ”Using language models to generate wrong, but plausible answers for multiple choice questions in Spanish”,
”Language”: ”eng” },
”Alternative title”: {
”Main title”: ”Automatisk Generering av Distraktorer för Spanska Läsförståelsefrågor”,
”Subtitle”: ”Användning av språkmodeller för att generera felaktiga men trovärdiga svar på flervalsfrågor på spanska”,
”Language”: ”swe”
},
”Supervisor1”: { ”Last name”: ”Boye”,
”First name”: ”Johan”,
”Local User Id”: ”u1ce6y3a”,
”E-mail”: ”jboye@kth.se”,
”organisation”: {”L1”: ”School of Electrical Engineering and Computer Science”,
”L2”: ”Computer Science” }
},
”Examiner1”: { ”Last name”: ”Engwall”,
”First name”: ”Olov”,
”Local User Id”: ”u1niocbs”,
”E-mail”: ”engwall@kth.se”,
”organisation”: {”L1”: ”School of Electrical Engineering and Computer Science”,
”L2”: ”Computer Science” }
},
”National Subject Categories”: ”10201, 10208”,
”Other information”: {”Year”: ”2023”, ”Number of pages”: ”1,97”},
”Copyrightleft”: ”copyright”,
”Series”: { ”Title of series”: ”TRITA-EECS-EX” , ”No. in series”: ”2023:0000” },
”Opponents”: { ”Name”: ”A. B. Normal & A. X. E. Normalè”},
”Presentation”: { ”Date”: ”2022-03-15 13:00”
,”Language”:”eng”
,”Room”: ”via Zoom https://kth-se.zoom.us/j/ddddddddddd”
,”Address”: ”Isafjordsgatan 22 (Kistagången 16)”
,”City”: ”Stockholm” },
”Number of lang instances”: ”3”,
”Abstract[eng]”: €€€€

A common evaluation method for students in the context of reading comprehension is the use of Multiple Choice Questions. A student must read a text and a question, and then choose the correct answer from a set of options, one of which one is the correct answer, and the other options are wrong. The wrong options are called distractors. Creating Multiple Choice Question exams is time-consuming, and a task that is open for automation. Distractor Generation is the task of generating wrong, but plausible options for Multiple Choice Questions. It is a task that can be addressed with Machine Learning and Large Language Models. As this task has been addressed in languages such as English, and Swedish, this work addresses the task for the Spanish language. This work achieves 3 objectives. The first one is the creation of a Multiple Choice Question dataset in Spanish with distractors, by manually tagging distractors from the dataset \textit{SQuAD-es}. The newly created dataset with distractors is called \textit{SQuAD-es-dist}. The second one is automatically generating distractors with machine learning methods. A BERT model is fine-tuned to generate distractors, and a GPT model is used through zero-shot learning to generate distractors. The third one is a human study on the generated distractors to evaluate the plausibility and usability of the distractors. Although both methods show to be effective, yet not perfect, at generating distractors, the GPT model shows better applicability and a higher capacity to confuse students in the task of Distractor Generation.

€€€€,
”Keywords[eng]”: €€€€
Distractor Generation, Multiple Choice Questions, Reading Comprehension, Spanish Language, BERT, GPT €€€€,
”Abstract[swe]”: €€€€

En vanlig utvärderingsmetod för studenter i samband med läsförståelse är användningen av flervalsfrågor. En elev måste läsa en text och en fråga, och sedan välja rätt svar från en uppsättning alternativ, varav ett är rätt svar och de andra alternativen är fel. De felaktiga alternativen kallas

distraktorer. Att skapa prov med flervalsfrågor är tidskrävande och en uppgift som är öppen för automatisering. Distraktorgenerering är uppgiften att generera felaktiga, men rimliga alternativ för flervalsfrågor. Det är en uppgift som kan lösas med maskininlärning och stora språkmodeller. Eftersom denna uppgift har behandlats på språk som engelska och svenska, behandlar detta arbete uppgiften för det spanska språket. Detta arbete uppnår 3 mål. Den första är skapandet av ett dataset med flervalsfrågor på spanska med distraktorer, genom manuell taggning av distraktorer från datasetet \textit{SQuAD-es}. Det nyskapade datasetet med distraktorer kallas \textit{SQuAD-es-dist}. Den andra metoden är att automatiskt generera distraktorer med maskininlärningsmetoder. En BERT-modell finjusteras för att generera distraktorer, och en GPT-modell används genom zero-shot-inlärning för att generera distraktorer. Den tredje metoden är en mänsklig studie av de genererade distraktorerna för att utvärdera hur rimliga och användbara distraktorerna är. Även om båda metoderna visar sig vara effektiva, men inte perfekta, för att generera distraktorer, visar GPT-modellen bättre tillämpbarhet och en högre kapacitet att förvirra studenter i uppgiften att generera distraktorer.

€€€€,
"Keywords[swe]": €€€€
Distraktorgeneration, Flervalsfrågor, Läsförståelse, Spanska språket, BERT, GPT €€€€,
"Abstract[spa]": €€€€

Para evaluar a alumnos en el contexto de comprensión de lectura se usan las preguntas de opción múltiple. El alumno debe leer un texto y una pregunta y, a continuación, elegir la respuesta correcta entre un conjunto de opciones, una de las cuales es la respuesta correcta y las demás opciones son incorrectas. Las opciones incorrectas se denominan distractores. La creación de exámenes con preguntas de opción múltiple requiere mucho tiempo, y es una tarea susceptible a la automatización. La Generación de Distractores es la tarea de generar opciones erróneas, pero plausibles, para Preguntas de Elección Múltiple. Es una tarea que puede abordarse con Aprendizaje Automático y Grandes Modelos de Lenguaje. Dado que esta tarea ha sido explorada en idiomas como el inglés, y el sueco, este trabajo aplica la tarea para el idioma español. Este trabajo alcanza 3 objetivos. El primero es la creación de un conjunto de datos de preguntas de respuesta múltiple en español con distractores, etiquetando manualmente los distractores del conjunto de datos \textit{SQuAD-es}. El nuevo conjunto de datos con distractores se denomina \textit{SQuAD-es-dist}. La segunda consiste en generar distractores automáticamente con métodos de aprendizaje automático. Se entrena y ajusta un modelo BERT para generar distractores y se utiliza un modelo GPT mediante ”zero-shot learning” para generar distractores. El tercero es un estudio humano sobre los distractores generados para evaluar la aplicabilidad y usabilidad de los distractores. Aunque ambos métodos muestran ser eficaces, pero no perfectos, en la generación de distractores, el modelo GPT muestra una mejor aplicabilidad y una mayor capacidad para confundir a los estudiantes en la tarea de Generación de Distractores.

€€€€,
"Keywords[spa]": €€€€
Generación de Distractores, Preguntas de Opción Múltiple, Comprensión de Lectura, Lenguaje Español, BERT, GPT €€€€,
}

acronyms.tex

```
%%% Local Variables:
%%% mode: latex
%%% TeX-master: t
%%% End:
% The following command is used with glossaries-extra
\setabbreviationstyle[acronym]{long-short}
% The form of the entries in this file is \newacronym{label}{acronym}{phrase}
%                                     or \newacronym[options]{label}{acronym}{phrase}
% see "User Manual for glossaries.sty" for the  details about the options, one example is shown below
% note the specification of the long form plural in the line below
\newacronym[longplural={Debugging Information Entities}]{DIE}{DIE}{Debugging Information Entity}
%
% The following example also uses options
\newacronym[shortplural={OSes}, firstplural={operating systems (OSes)}]{OS}{OS}{operating system}

% note the use of a non-breaking dash in long text for the following acronym

\newacronym{KTH}{KTH}{KTH Royal Institute of Technology}

\newacronym{ML}{ML}{Machine Learning}
\newacronym{NLP}{NLP}{Natural Language Processing}

\newacronym[longplural={Large Language Models}]{LLM}{LLM}{Large Language Model}

\newacronym[shortplural={MCQs}, firstplural={Multiple Choice Questions}]{MCQ}{MCQ}{Multiple Choice Question}

\newacronym{DG}{DG}{Distractor Generation}


\newacronym{BERT}{BERT}{Bidirectional Encoder Representations from Transformers}

\newacronym{GPT}{GPT}{Generative Pre-trained Transformer}
```