FACULTY OF ENGINEERING AND ARCHITECTURE

Language Model Adaptation with Applications in Artificial Intelligence for Education

Semere Kiros Bitew

Doctoral dissertation submitted to obtain the academic degree of
Doctor of Engineering

Supervisors
Prof. Chris Develder, PhD - Prof. Thomas Demeester, PhD
Department of Information Technology
Faculty of Engineering and Architecture, Ghent University

March 2024

GHENT UNIVERSITY

ISBN 978-94-6355-818-1
NUR 984
Wettelijk depot: D/2024/10.500/23

Members of the Examination Board

Chair

Prof. Filip De Turck, PhD, Ghent University

Other members entitled to vote

Prof. Orphée De Clercq, PhD, Ghent University
Prof. Toon De Pessemier, PhD, Ghent University
Prof. Djoerd Hiemstra, PhD, Radboud Universiteit, the Netherlands
Lucas Sterckx, PhD, LynxCare

Supervisors

Prof. Chris Develder, PhD, Ghent University
Prof. Thomas Demeester, PhD, Ghent University

Acknowledgements

After spending two years in the Netherlands for my master's studies, moving to yet another foreign country far from home to pursue a PhD in a topic yet to be discovered was not something I had planned. Nonetheless, it turned out to be an immensely rewarding experience. This thesis represents the culmination of that experience, made possible by the wonderful people around me.

First and foremost, I would like to express my sincere gratitude to my supervisors, Prof. Chris Develder and Prof. Thomas Demeester. You have continuously supported, motivated and trusted me throughout my PhD journey. I have learned so many things from both of you, but the following lessons particularly stick to memory. Thomas, from you, I have learned how to think critically as a researcher, to push my limits and the importance of supporting people through their lows. You have been a true mentor, always showing interest in what I have been doing, and your empathy during my personal challenging times has profoundly impacted my professional and personal development. Chris, you have imparted the skills of writing research papers with clarity and precision (and creating beautiful figures), alongside teaching me how to strike a balance between work and life.

Besides my main supervisors, I would also like to thank Johannes Deleu for his insightful guidance throughout my PhD. Without them and without our (formal and informal) brainstorming sessions, this thesis would not have been possible.

I would like to express my heartfelt appreciation to the esteemed members of the jury: Prof. Djoerd Hiemstra, Prof. Orphee De Clerq, Prof. Toon De Pessemier, dr. Lucas Sterckx. Their valuable feedback, constructive criticism, and insightful suggestions during the defense of my Ph.D. have greatly enhanced the quality and rigor of this work. I would also like to extend my thanks to Prof. Flip De Turck, the Chair of the Ph.D. defense, for overseeing the proceedings.

A special thank you goes to Djoerd Hiemstra, who was the second reader of my master's thesis and encouraged me to pursue a PhD. His recommendation was instrumental in my decision to start my PhD journey. My interest in research, specifically in NLP and information retrieval, was undoubtedly influenced by the courses I took with him.

I wish to thank all my colleagues and friends in iGent and De Krook.

I feel fortunate to have been part of such a fantastic team: Giannis, Klim, Amir, Manu, Yiwei, Maarten, Gargya, Fabio, François, Jens-Joris, Henri, Karel, Paloma, Cédric, Tom, Felix and Seza. Their support, discussions, and shared experiences have enriched my research journey and fostered a collaborative environment crucial to my growth as a researcher. I want to thank my friends Manu, Yiwei, and Amir for lending an ear during my personal struggles and for the adventures exploring Ghent together.

I extend a heartfelt acknowledgement to my Tigraian friends here in Belgium. In the face of the genocidal war in Tigrai, my place of origin, we have shared a collective sense of profound sadness. The unparalleled challenges of pursuing my PhD studies from halfway around the world, while Tigraians, including members of my own family, faced isolation from the rest of the world, would have been impossible without the unwavering support and empathy of my Tigraian friends here. They, too, were enduring the same harrowing experiences. The toll of this conflict has been deeply personal for me, as I have mourned the loss of a younger brother and a brother-in-law during this period of violence. Their memory and the strength of the Tigraian community continue to inspire my endeavors and resilience.

I also want to thank my family. My parents, Embanidey and Kirosa, thank you for being a constant source of inspiration, for always being there for me, for giving me so many opportunities, and for always believing in me. To my siblings, I am grateful for your love and support and for making me feel that you are always proud of me.

My final words are to my wife, Semu, and my daughter, Lelu. Your patience, love and limitless support have sustained me through endless nights, missed holidays, and weekends spent away at the office.

Ghent, Fall 2023
Semere

"To Robela, and to my friends who paid the ultimate sacrifice in defense of our cherished homeland, Tigrai."

Their memory eternal, our eternal debt, their only gain!

Table of Contents

List of Figures

List of Tables

List of Acronyms

AI	Artificial Intelligence
AQG	Automatic Question Generation
BART	Bidirectional Auto-Regressive Transformers
BERT	Bidirectional Encoder Representations from Transformers
GPT	Generative Pre-trained Transformer
GPU	Graphical Processing Unit
LR	Logistic Regression
LSTM	Long Short Term Memory
MCQs	Multiple Choice Questions
ML	Machine Learning
MLM	Masked Language Model
MT	Machine Translation
NER	Named Entity Recognition
NLP	Natural Language Processing
PLMs	Pretrained Language Models
SEQ2SEQ	Sequence to sequence
SOTA	State-Of-The-Art
SVM	Support Vector Machine
T5	Text-to-Text Transfer Transformer
TF-IDF	Term-Frequency Inverse-Document-Frequency

Samenvatting
– Summary in Dutch –

Artificiële intelligentie (AI) wordt een steeds essentiëler onderdeel van ons dagelijkse leven, met invloed op een breed spectrum van gebieden, van geavanceerde medische diagnostiek tot gepersonaliseerde webwinkel suggesties. De mogelijkheid van AI om te leren van een enorme hoeveelheid data zoals afbeeldingen, spraak en tekst stelt het in staat om de mens in verschillende domeinen te overtreffen. Voorbeelden hiervan zijn het spelen van complexe spellen, het optimaliseren van logistiek en het bieden van directe vertalingen. Natuurlijke taalverwerking of *natural language processing* (NLP) is een interdisciplinair onderdeel van AI en taalkunde dat machines in staat stelt menselijke taal te interpreteren, te begrijpen en te genereren. Naast de bekende technologieën zoals vertaling (bijvoorbeeld Google Translate) en spraakherkenning (bijvoorbeeld Siri), strekken de toepassingen van NLP zich uit tot het ondersteunen van gepersonaliseerd onderwijs door middel van geautomatiseerde chatbots, het ondersteunen van automatische lening- of kredietaanvragen in de financiële sector en het helpen bij geestelijke gezondheidsbeoordelingen, onder andere.

Een belangrijke drijfveer achter de recente vooruitgang in NLP is de ontwikkeling van vooraf getrainde taalmodellen of *pretrained language models* (PLM's). Deze modellen vertegenwoordigen een fundamentele verschuiving in het begrip en de generatie van menselijke taal door machines. In tegenstelling tot eerdere modellen die uitgebreide handmatige feature engineering vereisten, gebruiken PLM's enorme hoeveelheden tekstuele data om taalpatronen en nuances te leren. Deze methodologie stelt hen in staat om taalcontext, grammatica en semantiek diepgaand te begrijpen, wat de basis vormt voor talrijke NLP-toepassingen. PLM's staan centraal in het creëren van geavanceerde NLP-systemen zoals ChatGPT, die bekend staan om hun conversatievaardigheden en contextueel begrip. Deze PLM's hebben de state-of-the-art in verschillende NLP-taken voortgestuwd, waaronder machinale vertaling, interpretatie van natuurlijke taal, sentimentanalyse en het genereren en beantwoorden van vragen. Hun veelzijdigheid en efficiëntie maakten innovatieve toepassingen mogelijk in verschillende velden, waaronder het onderwijs, waar ze ondersteuning bieden voor intelligente tutorsystemen, het automatisch samenvatten van tekstinhoud, en realtime

taalondersteuning voor niet-moedertaalsprekers.

Het algemene thema van mijn proefschrift is het *aanpassen van taalmodellen*, voornamelijk voor toepassingen in AI in het onderwijs, om automatisch educatief materiaal te creëren. Dit proefschrift richt zich op de uitdagingen bij het formuleren van test- en oefenvragen in educatieve settings, een taak die traditioneel aanzienlijke training, ervaring, tijd en middelen vereist. Deze taak is vooral cruciaal in high-stakes omgevingen zoals certificeringen en toetsen, waar vragen niet hergebruikt kunnen worden. Mijn primaire onderzoek is met name gericht op twee educatieve taken: *generatie van afleiders* en *generatie van invul opgaven*. Generatie van afleiders verwijst naar het genereren van plausibele maar onjuiste antwoorden in meerkeuzevragen, terwijl generatie van invul opgaven verwijst naar het induceren van goed gekozen gaten in bestaande teksten, zodat deze dienst kunnen doen als grammatica oefening. Deze taken, hoewel reeds uitgebreid onderzocht, bieden onontgonnen mogelijkheden die aangepakt kunnen worden via de recente vooruitgang in taalmodellen. Als *secundair doel* onderzoek ik hoe coreference resolution kan aangepast worden aan nieuwe talen. Coreference resolution heeft als doel het groeperen van vermeldingen in de tekst aan de hand van de entiteiten waarnaar ze verwijzen in de echte wereld. Deze sleuteltaak in NLP is essentieel voor het begrijpen en genereren van samenhangende taal.

Na het vergelijken van klassieke machine learning modellen en taalmodelgebaseerde zoekmodellen voor de taak van afleidergeneratie (Hoofdstuk 2), richt ik me op het verbeteren van deze oplossing via uitbreiding naar grote taalmodellen zoals ChatGPT (Hoofdstuk 3). Vervolgens concentreer ik me op het aanpassen van een taalmodel voor het genereren van invul opgaven (Hoofdstuk 4). Ten slotte richt ik me op het aanpassen van coreference resolution (Hoofdstuk 5) aan nieuwe talen, ditmaal niet met toepassing in het onderwijs. In de volgende alinea's volgt een kort overzicht van elk hoofdstuk, telkens met nadruk op de belangrijkste bijdragen.

In Hoofdstuk 1 bied ik een kort overzicht van de eerdere literatuur over pretrained language models en toepassingen van AI in het onderwijs om de lezer in staat te stellen de in de volgende hoofdstukken beschreven termen te begrijpen.

De concrete onderzoekscontributies van deze thesis beginnen in Hoofdstuk 2 met het voorstellen van een neurale netwerkarchitectuur die een meertalig pretrained language model aanpast voor de taak van afleider rangschikking. We gebruiken dit model om slim afleiders te hergebruiken uit een bestaande set van handmatig gecreëerde antwoorden en afleiders, met als doel leraren te helpen efficiënt nieuwe meerkeuzevragen te creëren. Deze vragen hebben betrekking op een verscheidenheid aan domeinen, onderwerpen en talen. We tonen aan dat dit model in staat is om afleiders van betere kwaliteit te genereren in vergelijking met verschillende baselines, zowel wat betreft geautomatiseerde metrieken als een gebruikersstudie met leraren die wij op poten zetten.

In Hoofdstuk 3 benutten we grote taalmodellen of large language models (LLM) zoals ChatGPT om vrij afleiders te genereren, in plaats van bestaande afleiders te rangschikken. Hier introduceren we een nieuwe strategie om LLM's te begeleiden bij het genereren van plausibele afleiders. Dit omvat het aansturen van de LLM met vraagitems die automatisch zijn opgehaald uit vraag databanken, met behulp van lokale modellen gebouwd in Hoofdstuk 2. We laten zien dat de combinatie van lokale modellen met LLM's afleiders van hogere kwaliteit produceert.

In Hoofdstuk 4 verschuiven we onze focus naar een meer gespecialiseerde educatieve taak: het genereren van grammatica oefeningen met invul opgaven. We schetsen de creatie van een real-world dataset van Franse grammatica oefeningen met invul opgaven, die verschillende grammaticale aspecten omvatten. We kaderen deze taak als een voorbeeld van example-aware prediction, waarbij geschikte gaten in teksten worden geïdentificeerd op basis van gedeeltelijk geannoteerde gegevens. We stellen een nieuw neuraal netwerk voor, waarbij een pretrained language model aangepast wordt voor de voorspellingstaak. We demonstreren hoe de effectiviteit aanzienlijk verhoogt wanneer we de output van het model conditioneren op een voorbeeldoefening, in tegenstelling tot een baseline model dat onafhankelijk van deze voorbeelden werkt. Daarnaast analyseren we het inherente vermogen van het model om onderscheid te maken tussen elementaire types oefeningen zonder dat het expliciet getraind is om dit te doen. We benadrukken hierbij de herkenningscapaciteiten voor veelvoorkomende types in de testset.

In tegenstelling tot eerdere hoofdstukken die zich richtten op de educatieve taken van afleidergeneratie en grammaticaoefeningen met invul opgaven, wijken we in Hoofdstuk 5 af naar het aanpassen van de fundamentele NLP-taak coreference resolution aan nieuwe talen. We presenteren het aantrekkelijke idee om vertaaltools te gebruiken voor het bootstrappen van coreference resolution in talen met beperkte middelen of low-resource talen. Specifiek presenteren en analyseren we twee strategieën: (i) vertaal de train data in een high-resource taal naar de doeltaal en train een coreference-model en (ii) vertaal de test data naar een high-resource taal (bijvoorbeeld Engels) en gebruik een getraind coreference-model voor inferentie. We voeren ook een rigoureuze analyse uit naar de bron van fouten voor deze twee strategieën, waarbij de kwaliteit van machinale vertaalmodellen optreedt als de primaire beperkende factor.

Ten slotte bieden we in Hoofdstuk 6 een beknopte maar volledige samenvatting van de belangrijke onderzoeksbijdragen die in deze thesis gemaakt werden. Deze samenvatting belicht de belangrijkste inzichten die uit het onderzoek naar voren zijn gekomen, zodat de lezer een duidelijk begrip krijgt van de belangrijkste bevindingen van de thesis. Daarnaast verstrekken we aanbevelingen voor toekomstige onderzoeksrichtingen op basis van de conclusies van deze studie, waarbij we het belang van voortgezette exploratie en ontdekking in het veld benadrukken.

Summary

Artificial Intelligence (AI) has increasingly become a vital part of our everyday lives, impacting a broad spectrum of areas, from advanced medical diagnostics to personalized shopping suggestions. AI's ability to learn from a vast amount of data like images, speech and text allows it to outperform humans in various domains. Examples include playing complex games, optimizing logistics, and providing instant language translations. Natural language processing (NLP) is an interdisciplinary subfield of AI and linguistics that allows machines to interpret, understand, and generate human language. Beyond the familiar technologies such as text translation (e.g., Google translate) and voice recognition (e.g., Siri), NLP's applications extend to supporting personalized learning in education through automated chatbots, supporting automatic loan/credit applications in finance and aiding in mental health assessments, among others.

A key driver behind the recent progress in NLP is the development of pretrained language models (PLMs). These models represent a fundamental shift in machine understanding and generation of human language. Unlike earlier models that required extensive manual feature engineering, pretrained models use vast text data to learn language patterns and nuances. This methodology enables them to grasp language context, grammar, and semantics deeply, forming the basis for numerous NLP applications. PLMs are central to creating cutting-edge NLP systems such as ChatGPT, which are known for their conversational abilities and contextual understanding. These PLMs have pushed the state-of-the-art in several NLP tasks like machine translation, natural language inference, sentiment analysis, and question generation and answering etc. Their versatility and efficiency have enabled innovative applications across various fields, including education, where they support intelligent tutoring systems, content summarization automation, and instant language assistance for non-native speakers.

The overall theme of my dissertation is in *adapting language models* mainly for applications in AI in education to automatically create educational content. It addresses the challenges in formulating test and exercise questions in educational settings, which traditionally require significant training, experience, time, and resources. This is particularly critical in high-stakes environments like certifications and tests, where questions cannot be reused. In particular, the primary research is focused on two educational tasks:

distractor generation and *gap-filling exercise generation*. Distractor generation task refers to generating plausible but incorrect answers in multiple-choice questions, while gap-filling exercise generation refers to inducing well-chosen gaps to generate grammar exercises from existing texts. These tasks, although extensively researched, present unexplored avenues that recent advancements in language models can address. As a *secondary objective*, I explore the adaptation of *coreference resolution* to new languages. Coreference resolution is a key NLP task that involves clustering mentions in a text that refer to the same real-world entities, a process vital for understanding and generating coherent language.

After comparing classical machine learning approaches and language model-based retrieval models (Chapter 2) for the distractor generation task, I focused on improving the solution by extending it to large language models such as ChatGPT (Chapter 3). Then I focused on adapting a language model to the gap-fill exercise generation (Chapter 4). Finally, moving away from the application in education, I focused on adapting coreference resolution to new languages. In the paragraphs that follow, I offer a brief overview of each chapter, highlighting their key contributions.

In Chapter 1, I provide a brief overview of the previous literature on pre-trained language models and applications of AI in education to allow the reader to understand the terms described in subsequent chapters.

The achieved research contributions of this thesis start in Chapter 2 with proposing a neural network architecture that adapts a multilingual pre-trained language model for the task of distractor ranking. We use this model to smartly reuse distractors from a large existing set of manually created answers and distractors for questions over a variety of domains, subjects, and languages to help teachers create new MCQs. We show that this model is able to generate better quality distractors compared to baselines using automated metrics and a user study with teachers we conducted.

In Chapter 3, we leverage instruction-tuned large language models (LLMs) such as ChatGPT to freely generate distractors as compared to ranking existing distractors. Here, we introduce a novel strategy for guiding LLMs in generating plausible distractors. This involves prompting the LLMs with question items automatically retrieved from question banks, using local models we built in Chapter 2. We show that combining local models with LLMs produces higher quality distractors.

In Chapter 4, we shift our focus to the more specialized educational task of generating gap-fill grammar exercises. We outline the creation of a real-world dataset of French gap-filling exercises, covering various grammatical aspects. We frame the task as an example-aware prediction challenge, where suitable gaps in texts are identified based on partially annotated data. We propose and train a novel neural network by adapting a pretrained language model for the prediction task. We demonstrate that conditioning the model's output on an example exercise significantly increases its ef-

fectiveness compared to a baseline model that operates independently of examples. Additionally, we analyze the model's inherent ability to differentiate between elementary exercise types without being explicitly trained to do so, highlighting its recognition capabilities for commonly occurring types in the test set.

Unlike previous chapters that focused on the educational tasks of distractor generation and gap-fill grammar exercise generation, in Chapter 5 we switch the topic to adapting the fundamental NLP task of coreference resolution to new languages. We present the appealing idea of leveraging translation tools for bootstrapping coreference resolution in languages with limited resources. Specifically, we propose and analyze two strategies (i) translate the training data in high-resource language to the target language and train a coreference model and (ii) translate the test data into high-resource source language (e.g., English) and use a trained coreference model for inference. We also perform rigorous analysis on the source of errors for these two strategies, indicating the quality of machine translation models as the primary limiting factor.

Finally, in Chapter 6, we provide a concise yet comprehensive summary of the significant discoveries made throughout this thesis. This summary highlights the key insights that emerged from the research, providing the reader with a clear understanding of the thesis's main findings. Additionally, we provide recommendations for future research directions based on the conclusions drawn from the study, underscoring the importance of continued exploration and discovery in the field.

1 Introduction

"Those [...] who had been around for a long time, can see old ideas reappearing in new guises [...]. But the new costumes are better made, of better materials, as well as more becoming: so research is not so much going round in circles as ascending a spiral."

— Karen Spärck Jones

Artificial Intelligence (AI) systems refer to software in computers or machines that are used to perform tasks that usually require human intelligence (e.g., learning, reasoning, memorization) [1]. We use AI in our daily lives, sometimes even without realising it, in applications such as search engines, smart assistants, personalized social media feeds, chatbots [2], navigation apps, weather forecasting [3], movie/friend recommendation [4] etc. AI is also used in more complex applications such as autonomous vehicles [5], logistics optimization [6] etc. Many of these AI systems are built using Machine Learning (ML) algorithms that give computers the ability to learn from data and improve over time without being explicitly programmed [7].

As a subfield of AI, Natural Language Processing (NLP) combines computational linguistics with ML and statistical models to create systems that are capable of automatically processing the human language as if it "understands" it [8]. NLP is at the heart of several technologies that interact with human language, ranging from psycholinguistics applications such as suicide prediction [9] and personality type detection [10] to facilitating language translation [11] and sentiment analysis [12] in broader domains.

A key driver behind the recent progress in NLP is the development of Pretrained Language Models (PLMs). These models represent a fundamental shift in machine understanding and generation of human language. Unlike earlier models that required extensive manual feature engineering, pre-trained models use vast text data to learn language patterns and nuances. This methodology enables them to grasp language context, grammar, and semantics deeply, forming the basis for numerous NLP applications. PLMs are central to cutting-edge NLP systems such as OpenAI's ChatGPT, pushing the State-Of-The-Art (SOTA) in several tasks like machine translation, natural language inference, sentiment analysis, and question generation and answering etc. The versatility and efficiency of these models have significantly lowered the barrier to NLP implementation, fostering innovative applications in various fields, including education. In education, PLMs have been employed to facilitate intelligent tutoring systems, automate content summarization, and provide instant language support for non-native speakers.

The focus of this dissertation is in *adapting these advanced language models* mainly for applications in AI for education to automatically create educational content. This highlights the challenges in formulating test and exercise questions in educational settings, which traditionally require significant training, experience, time, and resources. This is even more critical in high-stakes environments like certifications and tests, where questions cannot be reused. The primary research involves two educational tasks: *distractor generation* and *gap-filling exercise generation*. Distractor generation task refers to generating plausible but incorrect answers in multiple-choice questions, while gap-filling exercise generation refers to inducing well-chosen gaps to generate grammar exercises from existing texts. These tasks have seen a lot of research in the past. However, existing methods have left room for further exploration. Recent advancements in pre-trained language models have opened new possibilities for research in these domains. In parallel, as a secondary and less pronounced objective, this dissertation also explores the adaptation of *coreference resolution* to new languages. Coreference resolution, a crucial task in NLP, involves identifying references in a text that relate to the same entities, a process vital for understanding and generating coherent language. In educational settings, using coreference resolution tools can improve the creation of high-quality, effective, and educationally sound content. For example, in distractor generation for Multiple Choice Questions (MCQs) based on reading comprehension, coreference resolution can help create distractors that are contextually relevant to the question's content. If the question is about a specific character or event, the system can avoid distractors about unrelated characters or events. Similarly, in

systems for creating gap-filling exercises, a coreference resolution tool can help choose sentences where filling a gap (replacing a pronoun or noun that refers to something mentioned earlier) makes the exercise challenging but achievable. This ensures the exercise tests understanding of the whole text.

The dissertation unfolds as follows: after comparing classical ML approaches and language model-based retrieval models (Chapter 2) for the distractor generation task, I focused on improving the solution by extending it to instruction-tuned large language models such as ChatGPT (Chapter 3). Then, I focused on adapting a language model to the gap-fill exercise generation (Chapter 4). The final part, diverging slightly from educational applications, delves into adapting coreference resolution (Chapter 5) to new languages, underlining its significance in the broader NLP landscape.

In this chapter, we describe (i) language model evolution (ii) language model adaptation techniques for NLP applications and (iii) the integration of AI in the education domain in the context of this thesis. By offering a concise overview and relevant literature references, the aim is to familiarize the reader with the terminologies and concepts essential for understanding the subsequent chapters. This chapter contains four sections:

- In 1.1, we outline the evolution of language models and describe various architectures and components frequently used in NLP.
- In 1.2, we describe the role of AI in the education domain.
- In 1.3, we highlight the main contributions of our work.
- In 1.4, we list the publications produced during my PhD.

1.1 Language models in NLP

1.1.1 Foundations of Language models

A language model is a probabilistic model of the human language [13]. Language models compute the likelihood of an entire sequence of words (e.g., a sentence, a paragraph) or the related task of predicting the probability of an upcoming word. For example, such a model could predict that the sequence *"The sky is the limit"* has a much higher probability of appearing in text than the same sequence of words with different ordering *"limit The is sky the"*. Moreover, in the sentence *"I like cats more than ..."*, such a model is expected to assign a higher probability for the word *"dog"* to follow, rather than the words *"lunch"* or *"laugh"*.

But why bother predicting upcoming words or assigning probabilities to sentences? The answer is simple: the utility of these predictions has several

real-life applications. They were first successfully used for automatic speech recognition, where identifying words in a noisy input is important [14]. In writing, language models' importance spans simple tasks such as correcting spelling errors and aiding in sentence auto-completion in search engines (such as Google search or Bing) to generating creative content for systems like chatbots, and translating languages.

A foundational approach in language modeling is the *word n-gram model*. This model operates on the principle that the likelihood of a subsequent word in a sequence is dependent only on a fixed number of preceding words. Take, for instance, the bigram model, an n-gram model reliant on just one preceding word. It calculates the probability of the next word based on the frequency of its occurrence with its predecessor. For example, the likelihood of the word *"world"* following *"hello"* is computed by comparing the frequency of the phrase *"hello world"* against the total occurrences of *"hello"* in a given text corpus. While n-grams are more basic compared to state-of-the-art models discussed in the following sections, they are fundamental in understanding language modeling principles.

1.1.2 Pretrained Language models (PLMs)

Pretraining in machine learning entails developing models using large datasets and generic tasks with the aim of learning general-purpose abilities and knowledge. This knowledge is then transferable to more specific downstream tasks. This approach became the standard in the computer vision community, particularly following the release of ImageNet [15], a large labeled image dataset.

In NLP, initial strides were made with the introduction of static (context-independent) word embeddings such as word2vec [16] and GloVe [17]. These early methods, which revolved around the concept of turning words into a vector of numbers so computers can understand them, offered simple, single-layer representations but necessitated training all remaining task-specific layers from scratch, as opposed to pretraining the entire model. Think of it as giving every word a unique code that shows its meaning and how it is related to other words. For instance, "dog" and "puppy" would have similar codes because they are related concepts. Despite the widespread adoption of word embeddings improving several NLP tasks, the need for substantial amounts of task-specific labeled data continued to be challenging. Moreover, these word embeddings struggle with words that have different meanings depending on the context. For instance, the word 'bank' will have the same representation, whether it refers to a financial institution or a river's edge.

The practice of pretraining entire models gained prominence in NLP with ELMo [18] and ULMFit [19], both utilizing the Long Short Term Memory (LSTM) architecture [20]. These models represented a significant leap, forming sophisticated, contextual representations that went a step further by giving words different codes based on their context in a sentence, and marking a considerable performance gain in many tasks in the field of NLP. However, a pivotal shift occurred in 2017 with the introduction of the Transformer architecture [21]. This model, employing an attention mechanism, processes all input data elements in parallel, eliminating the requirement for the computationally expensive recurrent computations found in LSTMs. This innovation facilitated the efficient management of long-range dependencies in data and significantly accelerated training processes, leading to more expressive and nuanced representations. Unlike the reliance on labeled datasets in computer vision, NLP pretraining depends on the availability of unlabeled data. The abundance of such data on the web allows pretraining to scale up (i.e., it is often possible to achieve better performance by training a larger model on a larger dataset).

In the following sections, we will first offer an overview of the Transformer architecture, subsequently discussing the various classes of PLMs that are grounded in this architecture. This discussion aims to illuminate the transformative impact of these advancements on the field of NLP.

1.1.2.1 The Transformer architecture

In 2017, Google introduced a new deep neural network architecture called the *Transformer* in their seminal paper *"Attention is all you need"* [21]. Initially designed for machine translation, it quickly became a cornerstone in tackling various NLP tasks. A transformer model is a type of encoder-decoder model that processes data in a unique way. The encoder could be thought of as part of the model that reads and understands the input information, like reading a sentence in a book, and the decoder as the part that takes what the encoder has understood and generates a response or output, similar to answering a question about the sentence read. More concretely, the transformer model uses a self-attention mechanism to assign importance weights to every part of the input sequence and how they relate to all other parts of the input. This self-attention mechanism maps input sequences to output sequences of the same length. Each output element contains information about all the other inputs in a way that reveals their relevance in the current context. Specifically, for every input word represented by its embedding vector $\mathbf{x}$, three smaller vectors called query ($\mathbf{q}$), key ($\mathbf{k}$) and value ($\mathbf{v}$) are created using weight matrices $\mathbf{W}^Q$, $\mathbf{W}^K$, $\mathbf{W}^V$ that are learned during the training process. The attention is then computed as

a weighted sum of the values, where the weight assigned to each value is computed by a compatibility function of the query with the corresponding key. In practical terms, word embeddings are grouped in a matrix $\mathbf{X}$, and the queries, keys and values are grouped into matrices $\mathbf{Q}$, $\mathbf{K}$ and $\mathbf{V}$. The self-attention is computed using equation 1.1.

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}}\right)\mathbf{V} \tag{1.1}$$

$$\text{where } \mathbf{Q} = \mathbf{W}^Q\mathbf{X},\ \mathbf{K} = \mathbf{W}^K\mathbf{X} \text{ and } \mathbf{V} = \mathbf{W}^V\mathbf{X} \tag{1.2}$$

Here, d_k represents the dimension of the key vectors. The Transformer enhances its self-attention mechanism through "multi-headed" attention. This involves creating multiple sets of query, key, value weight matrices, each initialized randomly. This design allows the model to represent multiple aspects of the input and focus on different positions simultaneously. It achieves this by concatenating the outputs from the h separate self-attention mechanisms, and projecting them via another learned weight matrix $\mathbf{W^O}$ using the following equations.

$$\text{MultiHead}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{Concat}\,(\text{head}_1, ..., \text{head}_h)\mathbf{W^O} \tag{1.3}$$

$$\text{where head}_i = \text{Attention}(\mathbf{Q}\mathbf{W}_i^Q, \mathbf{K}\mathbf{W}_i^Q, \mathbf{V}\mathbf{W}_i^Q) \tag{1.4}$$

Now that we have outlined the key innovation of the Transformer architecture — the multi-headed self-attention module — let's briefly explain how it integrates into the model's encoder-decoder architecture. The encoder is a stack of identical layers (i.e., 6 layers are used in the paper), each with two modules: a multi-headed self-attention module followed by point-wise fully connected feed-forward network module. Similarly, the decoder component is a stack of identical decoder layers but has additional multi-head cross-attention module that connects the encoder and decoder. Each module in both the encoder and decoder is surrounded by a residual connection, followed by a layer normalization that helps with gradient flow.

Unlike recurrent networks, the Transformer explicitly encodes the sequential nature of words using positional encodings in the input embeddings. Its architecture supports parallelization, allowing it to scale to large numbers of trainable parameters. These deep networks can be trained efficiently on parallel by leveraging Graphical Processing Unit (GPU) hardware, enabling learning from vast data amounts.

1.1.2.2 Encoder-only PLMs

Encoder-only models use the encoder part of the transformer architecture and are typically trained using the pretraining objective called *Masked Language Model (MLM)*. The MLM task requires masking a portion of the input text and then training a model to predict the masked tokens — in other words, to reconstruct the original non-masked input. When training an MLM, words are chosen at random to be masked using a special token `[MASK]`, or replaced by a random token. This forces the model to collect bidirectional information in making predictions. Such models stack several transformer layers to learn increasingly complex and meaningful representations. These types of models are typically used for producing embeddings. The most popular PLMs include Bidirectional Encoder Representations from Transformers (BERT) [22], RoBERTa [23], and XLM-R [24].

1.1.2.3 Decoder-only PLMs

The conventional *autoregressive* language task requires predicting the next word given all previous words in a sequence. Models trained using this training objective only utilized the decoder portion of the transformer architecture. Similar to the MLM models, autoregressive models also stack multiple transformer decoder layers with masked self-attention. This allows the models to attend to all previous words in the sequence when predicting the next tokens. Such models commonly called left-to-right models are well-suited to language generation, in particular in response to prompts as the continuation of a text. The most popular autoregressive models include the OpenAI's[1] Generative Pre-trained Transformer (GPT) family GPT-2 [25], GPT-3 [26], GPT-4 [27].

1.1.2.4 Encoder-Decoder PLMs

The encoder-decoder PLMs are class of models that learn to generate a sequence of words $y_1, y_2, ...y_n$ given an input sequence $x_1, x_2, ...x_m$. The objective is to maximize the output's log-likelihood: $log(P(y_1, ..., y_n|x_1, ..., x_m); \theta_T)$, in which θ_T are the parameter in the full encoder-decoder transformer model. The typical language task in these models is called the denoising task. In this task, different forms of corruption are applied to the input text, and the aim is to reconstruct the original sequence by denoising it. Forms of sequence corruption include sentence permutation, token masking similar to the MLM task, document rotation etc. Representative models

[1] https://openai.com/blog/introducing-gpts

include Text-to-Text Transfer Transformer (T5) [28] and Bidirectional Auto-Regressive Transformers (BART) [29]. The sequence-to-sequence nature of such PLMs makes them well-suited to perform tasks such as text summarization, question generation and distractor generation.

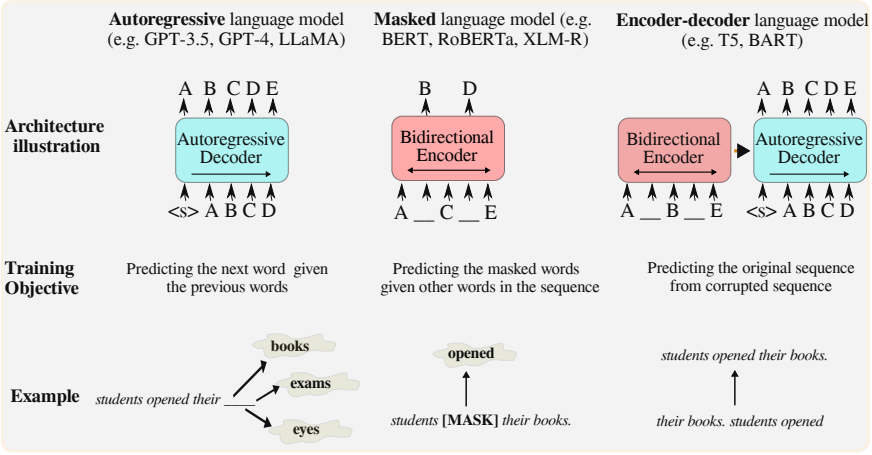


Figure 1.1: The three most common PLM types along with their architecture and training objective. Only the corruption strategy of document rotation (i.e., from BART) is shown for the encoder-decoder language model. Figure adapted from [29]

1.1.3 Fine-tuning PLMs

Up until the introduction of the Transformer structure in 2017 and subsequent pretrained models, the standard way of learning high-quality NLP models was the fully supervised paradigm. In the *fully-supervised paradigm*, a task-specific model is trained from scratch on a dataset of input-output examples for the target task. The standard shifted to *pretrain, fine-tune* paradigm, where a PLM is adapted to different downstream tasks by introducing additional parameters and fine-tuning them using task-specific objective functions. Fine-tuning typically utilizes less data, on the order of as few as several hundred to a thousand examples, in a supervised manner than the large amount of data, comprising several terabytes of text data with billions of words, used in pretraining language models. In the following paragraphs, we turn to describing strategies of how PLMs are adapted to perform accurately on disparate NLP downstream tasks.

PLMs as feature extractors: The most straightforward approach to using large PLMs for downstream tasks is to "freeze" the model weights and use its output vector representation as context-aware word embeddings for a subsequent architecture, which is trained from scratch for the specific task. While this still involves a forward pass through the PLM over the input text, the LM's weights are not fine-tuned, rendering this approach closer to a feature extraction family of approaches in classic NLP. Examples of such a strategy include cases with limited compute power or unsupervised tasks such as word sense disambiguation [30], where frozen embeddings enable a variety of operations such as cosine similarity, nearest-neighbour matching or clustering to perform these tasks.

Fully or partially fine-tuning PLMs: This strategy fine-tunes some (i.e., typically the top few layers) or all the layers of the PLM, and then adds one or two feed-forward output layers. The newly added layers are trained together with the PLM in an end-to-end fashion to adapt the PLM to the desired downstream task. Fine-tuning in this manner is most suitable for sequence classification tasks (e.g., finetuned sentiment analysis [12], natural language inference [23], semantic similarity [31]), sequence tagging tasks such as Named Entity Recognition (NER), and span extraction tasks (e.g., Question Answering [32]) in which the newly trained layers learn the start and end span of an answer. One of the baselines for our distractor generation task in Chapter 3 is an example of this approach.

Fine-tuning PLMs in Customized Models: Some tasks are specialized and may require significant additional architecture to adapt the PLM to their needs. With sufficient fine-tuning data, one may choose to train both a substantial task-specific architecture and also fine-tune the PLM at the same time. This is the preferred choice for structure prediction tasks such as dependency parsing [33], NER [34], and coreference resolution [32]. Our own proposed methods in Chapter 2 and Chapter 4 are examples of this approach. We built specialized architectures to adapt PLMs to the task of distractor ranking for creating Multiple-choice questions, and the task of gap-fill exercise generation for grammar learning, respectively. Moreover, in Chapter 5 we make use of existing PLM adapted coreference model that adds the substantial e2e-coref algorithm [32], which transforms ratings of pairs of spans produced by the language model into valid mention clusters.

Efficient fine-tuning: So far we have seen approaches where PLMs are used as feature extractors, top layers are fine-tuned, or significant new architecture is built on the top of the PLMs. However, another direction exists

for the efficient usage of these models that is not limited to the previously discussed approaches. The most common approach involves fine-tuning a small network that is tightly coupled with the PLM known as Adapter modules [35]. This method inserts a small set of newly initialized weights in the PLM. All weights in the PLM are set to 'freeze', and only the newly added weights of the adapters are updated during fine-tuning. Another technique in the efficient fine-tuning of PLMs is the Low-rank adaptation (LoRA) [36]. This technique, unlike adapters, approximates the existing weights of the PLM using two smaller matrices through low-rank matrix decomposition. During fine-tuning, LoRA updates only the low-rank matrices, which are small compared to the full-weight matrices. The original weights of the PLMs remain frozen.

1.1.4 Prompt and predict paradigm

The second significant paradigm shift in NLP revolves around the concept of using textual *prompts* instead of fine-tuning a separate PLM for each new downstream task, as discussed in the previous section. This approach involves reformulating downstream tasks to resemble those encountered during the PLM's pretraining phase. By carefully constructing appropriate prompts, one can direct the behavior of the PLM and leverage the knowledge it has encoded. This method enables the PLM to *predict* the desired output without necessitating additional task-specific fine-tuning. This exciting prospect of employing a single PLM across various tasks is gaining traction in the field.

With the emergence of large PLMs, the first signs of PLMs being multi-task learners emerged. For example, GPT-2 understands that if an instruction "TL;DR" ("too long; didn't read") is provided as a prompt, then it should generate a summarized form of the text supplied following the instruction. This ability of PLMs to perform tasks only by relying on the knowledge acquired during pretraining is called *zero-shot* learning. The performance of such models was even further improved with the introduction of instruction-tuned PLMs such as InstructGPT [37], which are explicitly trained to follow user instructions.

Another common strategy, *few-shot* setting, takes this concept a step further by providing the PLM with a handful of examples (so-called 'shots') that demonstrate the downstream task as part of the prompt construction. This process is called in-context learning and typically provides a few input-output exemplars that demonstrate the task. This has been successful for range of NLP tasks [26]. Another different prompting strategy, *chain-of-thought* [38], induces PLMs to generate intermediate steps before predicting

the final response.

In Chapter 3, we introduce a new variant of in-context few-shot learning wherein the example demonstrations presented to the PLM are determined dynamically, using a ranker model we built in Chapter 2.

1.2 AI for Education

The rapid advancement of artificial intelligence (AI) is profoundly transforming various aspects of human interaction, communication, lifestyle, learning, and professional practices. This transformation extends to the realm of education, where AI's impact is becoming increasingly significant. *AI in education* can be defined as the application of AI technologies to the four fundamental pillars of education: learning, teaching, assessment, and administration [39]. These pillars collectively support and enhance the educational process. In the following sections, we explore how AI has been seamlessly integrated into learning, teaching, and assessment. This exploration is particularly relevant to the thesis's emphasis on automatic educational question creation that supports these three crucial aspects of education. We aim to understand AI's role and transformative effect on these educational dimensions. The role of AI in enhancing school administration is noteworthy. It improves performance in various areas, including educational management platforms, planning, scheduling, and identifying systemic learning gaps to help educators in decision-making and enhance their system-wide efficiency. For an in-depth analysis of AI's impact on administration, see reference [39].

1.2.1 AI in Learning

In the context of learning, AI is transforming the acquisition of knowledge by students. The application of AI technologies in education primarily focuses on enhancing learning through personalized and adaptive systems. These systems utilize AI to tailor the curriculum and educational content to the unique needs, learning styles, and pace of each learner [40]. They can analyse a student's past learning experiences and draw comparisons with similar profiles among peers, thereby providing customized content and recommendations.

Moreover, AI-powered educational tools like intelligent tutoring systems (ITS) [41] provide real-time assistance and feedback. This approach enhances the efficiency and engagement of the learning process, adding an element of enjoyment without necessitating direct intervention from teachers. For instance, in a dialogue-based ITS, students can engage in

step-by-step instructional tasks through conversations in natural language with these systems. These intelligent systems dynamically adjust to the student's level of engagement, optimizing motivation and facilitating a more personalized learning experience. Another prominent example of AI-based learning is language learning applications like Duolingo [2], which support learning by offering access to language courses, dictionaries, and grammar, as well as providing real-time automated feedback on pronunciation, fluency, and other aspects.

1.2.2 AI in Teaching

The second role of AI in education is to support teachers. First, it helps teachers in creating more effective and engaging instructional materials. Several AI systems analyze vast amounts of educational data to identify the most effective teaching strategies for different groups of students [42–44]. For instance, AI systems that utilize multimodal sensor data to detect and analyze students' affective states were employed [45, 46]. Teachers used these systems to optimize their delivery of course material, refine pedagogical approaches, and enhance their communication strategies based on the identified emotional responses of students.

Second, AI systems have been used to enhance teachers' teaching ability. AI technologies have been applied to help teachers manage their classroom teaching efficiently [47–50]. AI systems can also automate administrative tasks (e.g., automatically generating variants of the same question using equally plausible but different distractor sets in a multiple-choice question [51]), freeing teachers to focus more on the pedagogical aspects of their roles. In addition to supporting teaching, AI has also been used to support teachers' professional development [42, 52]. In these studies, AI agents that analyzed real-time data in classrooms gave teachers comments and suggestions on their teaching ability, such as questioning skills and knowledge of pedagogical content of subject matter.

1.2.3 AI in Assessment

Integrating AI systems in educational assessment mainly focuses on automatic scoring and student performance prediction. AI systems have been used to grade student responses ranging from simple quizzes [53] to complex analytical essays [54]. This helps provide students with instant feedback that promotes learning and corrects inaccurate first responses [55], and also reduces teachers' workload. This automation not only makes

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

the grading process simpler but can also ensure a level of consistency and objectivity in evaluation that is challenging to achieve manually.

The second key area is using AI to predict student performance trajectory [56–58], particularly in online education. These AI systems identify trends and predict future academic outcomes by analyzing vast amounts of data on students' learning patterns, submission timelines, engagement in discussion forums, and past performance. This predictive capability is crucial for the early identification of students who might be at risk of underperforming, allowing educators and institutions to intervene with targeted support and resources. Furthermore, these insights can help in customizing teaching approaches and learning materials to suit the needs of individual students better.

1.2.4 Language models in AI in education

Pretrained language models are reshaping how educational content is created, delivered, and interacted with. The surge in using PLMs for developing educational applications has been evident in recent years. For instance, the number of research papers combining PLMs with educational applications in major academic venues has increased significantly. In the IEEE TLT journal[3], only 6 articles were published in 2020, the year I began working in this domain, compared to 17 in 2023. Similarly, at the NLP BEA workshop[4], the count rose from 4 papers in 2020 to 19 in 2023. In the subsequent paragraphs, we will briefly explore the application of PLMs across the educational roles outlined in the preceding sections, and how this thesis contributes to the field.

PLMs have been used to significantly contribute to personalized learning experiences [59]. By analyzing student-generated text data, such as essays or forum posts, these models can understand individual learning styles and linguistic capabilities. Additionally, these models have also enhanced the communication between teachers and students. For example, educational chatbots and virtual assistants [60, 61] that use PLMs have been developed to facilitate instant, on-demand interactions, answering student queries and providing explanations. This role can act as an additional support, extending the reach of teachers beyond the classroom.

In assessment, PLMs brought a new dimension to evaluating student performance. These models were employed to grade written assignments and automatically provide instant and constructive feedback [62–64]. Their ability to understand and analyze natural language allows for a more nu-

[3]https://ieee-edusociety.org/publication/ieee-tlt
[4]https://sig-edu.org/bea/2023

Table 1.1: Overview of contributions presented in this thesis.

Chapter	Task	Contribution
2	Distractor generation	Frame the task as a ranking problem and propose PLM-driven rankers
3	Distractor generation	New strategy to guide LLMs to generate distractors that outperform the rankers
4	Gap-filling exercise generation	New customized neural network that adapts a PLM for the task
5	Coreference resolution	Adapt coreference resolution to new languages using translation tools

anced assessment of students' written work, going beyond mere keyword matching to assess comprehension, argumentation, and creativity.

In this thesis, we focus on using PLMs to automate the creation of distractors for multiple-choice questions (MCQs) and develop gap-filling exercises in educational settings. The goal is to enhance personalized learning systems, as these automated questions can be customized to match the educational level and experiences of individual students. Additionally, this approach significantly benefits teachers by alleviating the burden of continuously generating tests and exercises.

Specifically, in distractor generation, teachers can create multiple versions of the same question, each with unique distractors. This diversity in question sets serves a dual purpose: it curtails the likelihood of students sharing answers and enhances the effectiveness of summative assessments. This feature is particularly crucial in high-stakes examinations, like certification tests, where question repetition is not viable due to security concerns. In contrast, for gap-filling exercises, this methods can be especially useful when introducing new grammar concepts. Teachers can effortlessly generate exercises based on their newly introduced grammar concepts to support the formative assessment of their students.

Furthermore, the automated nature of these questions simplifies the scoring process. This not only provides immediate feedback to students but also significantly reduces the workload for teachers, enhancing the overall efficiency of the assessment process.

1.3 Research contributions

In this section, we outline the main contribution of this thesis. We organize our technical contribution in chapters, each one tackling clearly defined

research questions. Table 1.1 gives an overview of the various contributions presented in this thesis. In Chapters 2 – 4, we propose neural network architectures and strategies for adapting PLMs for different educational tasks. While in Chapter 5, we deviate from educational applications to focus on adapting the task of coreference resolution to new languages. The contribution of each chapter is summarized as follows:

- In Chapter 2, we propose a neural network architecture that adapts multilingual PLM for the task of distractor ranking. We use this model to smartly reuse distractors from a large existing set of manually created answers and distractors for questions over a variety of domains, subjects, and languages to help teachers create new MCQs. We demonstrate that this model is able to generate higher quality distractors compared to baselines. This is evidenced through a user study with teachers as well as through automated metrics.

- In Chapter 3, as a direct extension of the proposed method in Chapter 2, we leverage large language models (LLM) to generate free-form distractors as compared to ranking existing distractors. We propose a novel strategy for guiding LLMs in generating plausible distractors by prompting them with dynamically retrieved example demonstrations using a question ranker proposed in Chapter 2. We show that combining local models with LLMs produces higher quality distractors.

- In Chapter 4, we present a method to adapt a language model for a gap-fill grammar exercise generation task. We propose and create a specialized neural network architecture to customize a language model to predict suitable gaps in texts (e.g., a paragraph, sentence) for language learning in French. Moreover, we publicly release the real-world dataset we created for the task.

- In Chapter 5, we extend our adaptability theme of the thesis to adapting coreference resolution, a key NLP task, to new languages. We examine the use of translation tools to facilitate coreference resolution in resource-limited languages. We analyze two approaches: translating training data from a high-resourced language to the target language for model training, and translating test data to a high-resource language (e.g., English) for model inference. We also analyze the main challenges of these methods, identifying the limitations of machine translation tools as the primary issue.

- In Chapter 6, we summarize our core findings and outline future research directions opened by this thesis.

Additionally, appendices contain additional published work that may not directly align with the main theme of this thesis. Thus, in Appendix A [9] we describe our contribution to CLPsych 2019 shared task where we achieve competitive results using linear models and ensemble models to predict the degree of suicide risk of people based on their posts on Reddit. Furthermore, I contributed as a co-author in [12, 65–67].[5] In [65], we provide the first multi-format educational dataset in which each question is phrased in two forms, cloze and open-ended, and linked to its grounding sentences. In [66], we propose a strategy for selecting content (e.g., paragraph or sentence) to support question generation systems in the context of education. In [67], we propose a simple solution to a specific failure case on conjoined mentions (e.g., 'Tom and Mary') of the SOTA word-level coreference resolution model. Finally, in [12], we examine how fine-tuned multilingual models generalize to out-of-distribution test data in zero-shot cross-lingual transfer scenarios for sentiment classification. This study includes an analysis of the effects of language and domain shifts between training and testing data.

[5]These four papers are not included in the appendices in accordance with the faculty policy.

1.4 Publications

The research[6] results obtained during this PhD research have been published in scientific journals and presented at a series of international conferences and workshops. The following list provides an overview of these publications.

1.4.1 Publications in international journals (listed in the Science Citation Index[7])

I **S.K. Bitew**, A. Hadifar, L. Sterckx, J. Deleu, C. Develder, and T. Demeester, *Learning to Reuse Distractors to support Multiple Choice Question Generation in Education*. IEEE Transactions on Learning Technologies. 2022.

1.4.2 Publications in international conferences

III **S.K. Bitew**, J. Deleu, A. Seza Doğruöz, C. Develder and T. Demeester, *Learning from Partially Annotated Data: Example-aware Creation of Gap-filling Exercises for Language Learning*. Proceedings of the 18th Workshop on Innovative Use of NLP for Building Educational Applications (BEA) at EACL, 2023

IV **S.K. Bitew**, J. Deleu, C. Develder and T. Demeester, *Lazy Low-Resource Coreference Resolution: a Study on Leveraging Black-Box Translation Tools*. Proceedings of the Fourth Workshop on Computational Models of Reference, Anaphora and Coreference (CRAC) at EMNLP 2021.

V **S.K. Bitew**, J. Deleu, C. Develder and T. Demeester, *Distractor Generation for Multiple-Choice Questions with Predictive Prompting and Large Language Models*. Proceedings of the First Workshop on Responsible Knowledge Discovery in Education (RKDE) at ECML-PKDD 2023.

VI **S.K. Bitew**, G. Bekoulis, J. Deleu, L. Sterckx, K. Zaporojets, T. Demeester, and C. Develder, *Predicting Suicide Risk from Online Postings*

[6]This research was funded by the Flanders Innovation and Entrepreneurship (VLAIO), Flanders, Belgium, through the imec-icon Project "AI-Driven e-Assessment" (AIDA); and in part by the Flemish Government through the "Onderzoeksprogramma ArtificiëleIntelligentie (AI) Vlaanderen" Program, Research Foundation Flanders and the Flemish Government under the Research Program Artificial Intelligence. This research would not have been possible without their support.

[7]The publications listed are recognized as 'A1 publications', according to the following definition used by Ghent University: *"A1 publications are articles listed in the Science Citation Index, the Social Science Citation Index or the Arts and Humanities Citation Index of the ISI Web of Science, restricted to contributions listed as article, review, letter, note or proceedings paper."*

in Reddit – The UGent-IDLab submission to the CLPysch 2019 Shared Task A. 6th Ann. Workshop on Computational Linguistics and Clinical Psychology (CLPsych 2019) at NAACL-HLT, 2019.

1.4.3 Other publications (not included in this thesis)

VII **S.K. Bitew**, V. Schelstraete, K. Zaporojets, K. Van Nieuwenhove, R. Meganck, and C. Develder, *Personality Style Recognition via Machine Learning: Identifying Anaclitic and Introjective Personality Styles from Patients' Speech*. Computational Linguistics in the Netherlands Journal, 2023.

VIII A. Hadifar, **S.K. Bitew** , J. Deleu, C. Develder, and T. Demeester, *EduQG: A Multi-Format Multiple-Choice Dataset for the Educational Domain*. IEEE-Access, 2023.

IX A. Hadifar, **S.K. Bitew**, J. Deleu, V. Hoste, C. Develder, and T. Demeester, *Diverse Content Selection for Educational Question Generation*. Proceedings of the Student Research Workshop at EACL (SRW), 2023.

X K. D'Oosterlinck, **S.K. Bitew**, B. Papineau, C. Potts, T. Demeester, and C. Develder, *CAW-coref: Conjunction-Aware Word-level Coreference Resolution*. Proceedings of the Sixth Workshop on Computational Models of Reference, Anaphora and Coreference (CRAC) at EMNLP 2023.

XI M. De Raedt, **S.K. Bitew**, Fréderic Godin, T. Demeester, and C. Develder, *Zero Shot Cross-Lingual Sentiment Classification under Distribution Shift: an Exploratory Study*. Proceedings of the Third Workshop on Multilingual Representation Learning (MRL) at EMNLP 2023.

References

[1] E. Commission, S. Directorate-General for Education, Youth, and Culture. *Ethical guidelines on the use of artificial intelligence (AI) and data in teaching and learning for educators*. Publications Office of the European Union, 2022.

[2] H. Wang, Z. Lu, H. Li, and E. Chen. *A dataset for research on short-text conversations*. In Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, pages 935–945, 2013.

[3] S. Xingjian, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-c. Woo. *Convolutional LSTM network: A machine learning approach for precipitation nowcasting*. In Advances in neural information processing systems, pages 802–810, 2015.

[4] Q. Diao, M. Qiu, C.-Y. Wu, A. J. Smola, J. Jiang, and C. Wang. *Jointly Modeling Aspects, Ratings and Sentiments for Movie Recommendation (JMARS)*. In Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, pages 193–202, New York, NY, USA, 2014. ACM. Available from: http://doi.acm.org/10.1145/2623330.2623758, doi:10.1145/2623330.2623758.

[5] M. Bojarski, D. Del Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, et al. *End to End Learning for Self-Driving Cars*. 2016.

[6] M. Klumpp. *Automation and artificial intelligence in business logistics systems: human reactions and collaboration requirements*. International Journal of Logistics Research and Applications, 21(3):224–242, 2018.

[7] A. L. Samuel. *Some studies in machine learning using the game of checkers*. IBM Journal of research and development, 3(3):210–229, 1959.

[8] *IBM Cloud Learning Hub*. Available from: https://www.ibm.com/cloud/learn.

[9] S. K. Bitew, G. Bekoulis, J. Deleu, L. Sterckx, K. Zaporojets, T. Demeester, and C. Develder. *Predicting suicide risk from online postings in Reddit the UGent-IDLab submission to the CLPysch 2019 shared task a*. In Proceedings of the sixth workshop on computational linguistics and clinical psychology, pages 158–161, 2019.

[10] S. K. Bitew, V. Schelstraete, K. Zaporojets, K. Van Nieuwenhove, R. Meganck, and C. Develder. *Personality Style Recognition via Machine Learning: Identifying Anaclitic and Introjective Personality Styles from Patients' Speech*. arXiv preprint arXiv:2311.04088, 2023.

[11] F. Stahlberg. *Neural machine translation: A review*. Journal of Artificial Intelligence Research, 69:343–418, 2020.

[12] M. De Raedt, S. K. Bitew, F. Godin, T. Demeester, and C. Develder. *Zero-Shot Cross-Lingual Sentiment Classification under Distribution Shift: an Exploratory Study*. In D. Ataman, editor, Proceedings of the 3rd Workshop on Multi-lingual Representation Learning (MRL), pages 50–66, Singapore, December 2023. Association for Computational Linguistics. Available from: https://aclanthology.org/2023.mrl-1.5, doi:10.18653/v1/2023.mrl-1.5.

[13] D. Jurafsky and J. H. Martin. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*.

[14] F. Jelinek. *Statistical methods for speech recognition*. MIT press, 1998.

[15] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. *Imagenet: A large-scale hierarchical image database*. In 2009 IEEE conference on computer vision and pattern recognition, pages 248–255. Ieee, 2009.

[16] T. Mikolov, K. Chen, G. S. Corrado, and J. Dean. *Efficient Estimation of Word Representations in Vector Space*. In International Conference on Learning Representations, 2013. Available from: https://api.semanticscholar.org/CorpusID:5959482.

[17] J. Pennington, R. Socher, and C. Manning. *GloVe: Global Vectors for Word Representation*. In A. Moschitti, B. Pang, and W. Daelemans, editors, Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics. Available from: https://aclanthology.org/D14-1162, doi:10.3115/v1/D14-1162.

[18] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer. *Deep Contextualized Word Representations*. In M. Walker, H. Ji, and A. Stent, editors, Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), pages 2227–2237, New Orleans, Louisiana, June 2018. Association

for Computational Linguistics. Available from: https://aclanthology.org/N18-1202, doi:10.18653/v1/N18-1202.

[19] J. Howard and S. Ruder. *Universal Language Model Fine-tuning for Text Classification*. In I. Gurevych and Y. Miyao, editors, Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 328–339, Melbourne, Australia, July 2018. Association for Computational Linguistics. Available from: https://aclanthology.org/P18-1031, doi:10.18653/v1/P18-1031.

[20] S. Hochreiter and J. Schmidhuber. *Long short-term memory*. Neural computation, 9(8):1735–1780, 1997.

[21] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. *Attention is all you need*. Advances in neural information processing systems, 30, 2017.

[22] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. In J. Burstein, C. Doran, and T. Solorio, editors, Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. Available from: https://aclanthology.org/N19-1423, doi:10.18653/v1/N19-1423.

[23] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. *Roberta: A robustly optimized bert pretraining approach*. arXiv preprint arXiv:1907.11692, 2019.

[24] A. Conneau, K. Khandelwal, N. Goyal, V. Chaudhary, G. Wenzek, F. Guzmán, E. Grave, M. Ott, L. Zettlemoyer, and V. Stoyanov. *Unsupervised Cross-lingual Representation Learning at Scale*. In D. Jurafsky, J. Chai, N. Schluter, and J. Tetreault, editors, Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 8440–8451, Online, July 2020. Association for Computational Linguistics. Available from: https://aclanthology.org/2020.acl-main.747, doi:10.18653/v1/2020.acl-main.747.

[25] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever, et al. *Improving language understanding by generative pre-training*. 2018.

[26] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. *Language models are*

few-shot learners. Advances in neural information processing systems, 33:1877–1901, 2020.

[27] OpenAI. *GPT-4 Technical Report*, 2023. arXiv:2303.08774.

[28] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. *Exploring the limits of transfer learning with a unified text-to-text transformer*. The Journal of Machine Learning Research, 21(1):5485–5551, 2020.

[29] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer. *BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension*. In D. Jurafsky, J. Chai, N. Schluter, and J. Tetreault, editors, Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 7871–7880, Online, July 2020. Association for Computational Linguistics. Available from: https://aclanthology.org/2020.acl-main.703, doi:10.18653/v1/2020.acl-main.703.

[30] G. Wiedemann, S. Remus, A. Chawla, and C. Biemann. *Does BERT Make Any Sense? Interpretable Word Sense Disambiguation with Contextualized Embeddings*. In Proceedings of the 15th Conference on Natural Language Processing (KONVENS 2019): Long Papers, pages 161–170, Erlangen, Germany, 2019. German Society for Computational Linguistics & Language Technology.

[31] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. R. Salakhutdinov, and Q. V. Le. *Xlnet: Generalized autoregressive pretraining for language understanding*. Advances in neural information processing systems, 32, 2019.

[32] M. Joshi, D. Chen, Y. Liu, D. S. Weld, L. Zettlemoyer, and O. Levy. *Spanbert: Improving pre-training by representing and predicting spans*. Transactions of the association for computational linguistics, 8:64–77, 2020.

[33] A. Üstün, A. Bisazza, G. Bouma, and G. van Noord. *UDapter: Language Adaptation for Truly Universal Dependency Parsing*. In B. Webber, T. Cohn, Y. He, and Y. Liu, editors, Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 2302–2315, Online, November 2020. Association for Computational Linguistics. Available from: https://aclanthology.org/2020.emnlp-main.180, doi:10.18653/v1/2020.emnlp-main.180.

[34] F. Souza, R. Nogueira, and R. Lotufo. *Portuguese named entity recognition using BERT-CRF*. arXiv preprint arXiv:1909.10649, 2019.

[35] A. Bapna and O. Firat. *Simple, Scalable Adaptation for Neural Machine Translation*. In K. Inui, J. Jiang, V. Ng, and X. Wan, editors, Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 1538–1548, Hong Kong, China, November 2019. Association for Computational Linguistics. Available from: https://aclanthology.org/D19-1165, doi:10.18653/v1/D19-1165.

[36] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen. *LoRA: Low-Rank Adaptation of Large Language Models*. In The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022. OpenReview.net, 2022. Available from: https://openreview.net/forum?id=nZeVKeeFYf9.

[37] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, et al. *Training language models to follow instructions with human feedback*. Advances in Neural Information Processing Systems, 35:27730–27744, 2022.

[38] J. Wei, X. Wang, D. Schuurmans, M. Bosma, brian ichter, F. Xia, E. H. Chi, Q. V. Le, and D. Zhou. *Chain of Thought Prompting Elicits Reasoning in Large Language Models*. In A. H. Oh, A. Agarwal, D. Belgrave, and K. Cho, editors, Advances in Neural Information Processing Systems, 2022. Available from: https://openreview.net/forum?id=_VjQlMeSB_J.

[39] Q. Xia, T. K. Chiu, X. Zhou, C. S. Chai, and M. Cheng. *Systematic literature review on opportunities, challenges, and future research recommendations of artificial intelligence in education*. Computers and Education: Artificial Intelligence, page 100118, 2022.

[40] T. A. Mikropoulos and A. Natsis. *Educational virtual environments: A ten-year review of empirical research (1999–2009)*. Computers & education, 56(3):769–780, 2011.

[41] H. S. Nwana. *Intelligent tutoring systems: an overview*. Artificial Intelligence Review, 4(4):251–277, 1990.

[42] V. Lampos, J. Mintz, and X. Qu. *An artificial intelligence approach for selecting effective teacher communication strategies in autism education*. npj Science of Learning, 6(1):25, 2021.

[43] N. L. S. Aldeman, K. M. de Sá Urtiga Aita, V. P. Machado, L. C. D. da Mata Sousa, A. G. B. Coelho, A. S. da Silva, A. P. da Silva Mendes,

F. J. de Oliveira Neres, and S. J. H. do Monte. *Smartpathk: a platform for teaching glomerulopathies using machine learning*. BMC medical education, 21(1):248, 2021.

[44] D. Crowe, M. LaPierre, and M. Kebritchi. *Knowledge based artificial augmentation intelligence technology: Next step in academic instructional tools for distance learning*. TechTrends, 61(5):494–506, 2017.

[45] P. J. Standen, D. J. Brown, M. Taheri, M. J. Galvez Trigo, H. Boulton, A. Burton, M. J. Hallewell, J. G. Lathe, N. Shopland, M. A. Blanco Gonzalez, et al. *An evaluation of an adaptive learning system based on multimodal affect recognition for learners with intellectual disabilities*. British Journal of Educational Technology, 51(5):1748–1765, 2020.

[46] D. Luo. *Guide Teaching System Based on Artificial Intelligence*. International Journal of Emerging Technologies in Learning (iJET), 13(08):pp. 90–102, Aug. 2018. Available from: https://online-journals.org/index.php/i-jet/article/view/9058, doi:10.3991/ijet.v13i08.9058.

[47] J. Zhang. *Computer assisted instruction system under artificial intelligence technology*. International Journal of Emerging Technologies in Learning (iJET), 16(5):4–16, 2021.

[48] D. Yang, E.-S. Oh, and Y. Wang. *Hybrid physical education teaching and curriculum design based on a voice interactive artificial intelligence educational robot*. Sustainability, 12(19):8000, 2020.

[49] Y. Wang and G. Zheng. *Application of artificial intelligence in college dance teaching and its performance analysis*. International Journal of Emerging Technologies in Learning (iJET), 15(16):178–190, 2020.

[50] J. Mahon, B. Bryant, B. Brown, and M. Kim. *Using second life to enhance classroom management practice in teacher education*. Educational Media International, 47(2):121–134, 2010.

[51] S. K. Bitew, A. Hadifar, L. Sterckx, J. Deleu, C. Develder, and T. Demeester. *Learning to reuse distractors to support multiple choice question generation in education*. IEEE Transactions on Learning Technologies, 2022.

[52] K. D. H. Gunawan, L. Liliasari, I. Kaniawati, and W. Setiawan. *Implementation of competency enhancement program for science teachers assisted by artificial intelligence in designing HOTS-based integrated science learning*. Jurnal Penelitian dan Pembelajaran IPA, 7(1):55–65, 2021.

[53] A. Fazal, F. K. Hussain, and T. S. Dillon. *An innovative approach for automatically grading spelling in essays using rubric-based scoring*. Journal of Computer and System Sciences, 79(7):1040–1056, 2013.

[54] D. Ramesh and S. K. Sanampudi. *An automated essay scoring systems: a systematic literature review*. Artificial Intelligence Review, 55(3):2495–2527, 2022.

[55] M. L. Epstein, A. D. Lazarus, T. B. Calvano, K. A. Matthews, R. A. Hendel, B. B. Epstein, and G. M. Brosvic. *Immediate feedback assessment technique promotes learning and corrects inaccurate first responses*. The Psychological Record, 52:187–201, 2002.

[56] Ö. F. Akmeşe, H. Kör, and H. Erbay. *Use of machine learning techniques for the forecast of student achievement in higher education*. Information Technologies and Learning Tools, 82(2):297–311, 2021.

[57] R. Costa-Mendes, T. Oliveira, M. Castelli, and F. Cruz-Jesus. *A machine learning approximation of the 2015 Portuguese high school student grades: A hybrid approach*. Education and Information Technologies, 26(2):1527–1547, 2021.

[58] J. Yu. *Academic Performance Prediction Method of Online Education using Random Forest Algorithm and Artificial Intelligence Methods.* International Journal of Emerging Technologies in Learning, 15(5), 2021.

[59] D. Kulshreshtha, M. Shayan, R. Belfer, S. Reddy, I. V. Serban, and E. Kochmar. *Few-shot question generation for personalized feedback in intelligent tutoring systems*. In 11th Conference on Prestigious Applications of Artificial Intelligence, 25 July 2022, Vienna, Austria (co-located with IJCAI-ECAI 2022), pages pp. 17–30, 2022. doi:10.3233/FAIA220062.

[60] P. Smutny and P. Schreiberova. *Chatbots for learning: A review of educational chatbots for the Facebook Messenger*. Computers & Education, 151:103862, 2020.

[61] M. A. Kuhail, N. Alturki, S. Alramlawi, and K. Alhejori. *Interacting with educational chatbots: A systematic review*. Education and Information Technologies, 28(1):973–1018, 2023.

[62] P. U. Rodriguez, A. Jafari, and C. M. Ormerod. *Language models and automated essay scoring*. arXiv preprint arXiv:1909.09482, 2019.

[63] R. Yang, J. Cao, Z. Wen, Y. Wu, and X. He. *Enhancing automated essay scoring performance via fine-tuning pre-trained language models with combination of regression and ranking*. In Findings of the Association for Computational Linguistics: EMNLP 2020, pages 1560–1569, 2020.

[64] C. M. Ormerod, A. Malhotra, and A. Jafari. *Automated essay scoring using efficient transformer-based language models*. arXiv preprint arXiv:2102.13136, 2021.

[65] A. Hadifar, S. K. Bitew, J. Deleu, C. Develder, and T. Demeester. *EduQG: A Multi-Format Multiple-Choice Dataset for the Educational Domain*. IEEE Access, 11:20885–20896, 2023.

[66] A. Hadifar, S. K. Bitew, J. Deleu, V. Hoste, C. Develder, and T. Demeester. *Diverse Content Selection for Educational Question Generation*. In Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics: Student Research Workshop, pages 123–133, 2023.

[67] K. D'Oosterlinck, S. K. Bitew, B. Papineau, C. Potts, T. Demeester, and C. Develder. *CAW-coref: Conjunction-Aware Word-level Coreference Resolution*. In M. Ogrodniczuk, V. Ng, S. Pradhan, and M. Poesio, editors, Proceedings of The Sixth Workshop on Computational Models of Reference, Anaphora and Coreference (CRAC 2023), pages 8–14, Singapore, December 2023. Association for Computational Linguistics. Available from: https://aclanthology.org/2023.crac-main.2, doi:10.18653/v1/2023.crac-main.2.

2

Adapting Language Models to Distractor Ranking for Educational Multiple-Choice Questions

This chapter highlights our contribution to a generic educational task of distractor generation. We introduce a new neural network architecture that adapts a multilingual pretrained language model to ranking existing distractors. Our innovative, data-driven method employs context-aware representations from language models for both questions and distractors. We use this model to smartly reuse distractors from a large existing set of manually created answers and distractors for questions over a variety of domains, subjects, and languages to help teachers create new MCQs. We show that our model is able to generate better quality distractors compared to baselines using automated metrics and a user study with teachers we conducted.

⋆⋆⋆

Learning to Reuse Distractors to support Multiple Choice Question Generation in Education

S.K. Bitew, A. Hadifar, L. Sterckx, J. Deleu, C. Develder and T. Demeester

In IEEE Transactions on Learning Technologies, 2022

Abstract Multiple choice questions (MCQs) are widely used in digital learning systems, as they allow for automating the assessment process. However, due to the increased digital literacy of students and the advent of social media platforms, MCQ tests are widely shared online, and teachers are continuously challenged to create new questions, which is an expensive and time-consuming task. A particularly sensitive aspect of MCQ creation is to devise relevant distractors, i.e., wrong answers that are not easily identifiable as being wrong. This paper studies how a large existing set of manually created answers and distractors for questions over a variety of domains, subjects, and languages can be leveraged to help teachers in creating new MCQs, by the smart reuse of existing distractors. We built several data-driven models based on context-aware question and distractor representations, and compared them with static feature-based models. The proposed models are evaluated with automated metrics and in a realistic user test with teachers. Both automatic and human evaluations indicate that context-aware models consistently outperform a static feature-based approach. For our best-performing context-aware model, on average 3 distractors out of the 10 shown to teachers were rated as high-quality distractors. We create a performance benchmark, and make it public, to enable comparison between different approaches and to introduce a more standardized evaluation of the task. The benchmark contains a test of 298 educational questions covering multiple subjects & languages and a 77k multilingual pool of distractor vocabulary for future research.

2.1 Introduction

Online learning has become an indispensable part of educational institutions. It has emerged as a necessary resource for students and schools all over the globe. The recent COVID-19 pandemic has made the transition to online learning even more pressing. One very important aspect of online learning is the need to generate homework, test, and exam exercises to aid and evaluate the learning progress of students [1]. Multiple choice questions (MCQs) are the most common form of exercises [2] in online education as they can easily be scored automatically. However, the construction of MCQs is time consuming [3] and there is a need to continuously generate new (variants of) questions, especially for testing, since students tend to share questions and correct answers from MCQs online (e.g., through social media).

The rapid digitization of educational resources opens up opportuni-

ties to adopt artificial intelligence (AI) to automate the process of MCQ construction. A substantial number of questions already exist in a digital format, thus providing the required data as a first step toward building AI systems. The automation of MCQ construction could support both teachers and learners. Teachers could benefit from an increased efficiency in creating questions, in their already high workload. Students' learning experience could improve due to increased practice opportunities based on automatically generated exercises, and if these systems are sufficiently accurate, they could power personalized learning [4].

A crucial step in MCQ creation is the generation of distractors [5]. Distractors are incorrect options that are related to the answer to some degree. The quality of an MCQ heavily depends on the quality of distractors [3]. If the distractors do not sufficiently challenge learners, picking the correct answer becomes easy, ultimately degrading the discriminative power of the question. The automatic suggestion of distractors will be the focus of this paper.

Several works have already proposed distractor generation techniques for automatic MCQ creation, mostly based on selecting distractors according to their similarity to the correct answer. In general, two approaches are used to measure the similarity between distractors and an answer: graph-based and corpus-based methods. *Graph-based* approaches use the semantic distance between concepts in the graph as a similarity measure. In language learning applications, typically WordNet [6, 7] is used to generate distractors, while for factoid questions domain-specific (ontologies) are used to generate distractors [8–11]. In *corpus-based methods*, similarity between distractors and answers has been defined as having similar frequency count [12], belonging to the same POS class [13], having a high co-occurrence likelihood [14], having similar phonetic and morphological features [7], and being nearby in embedding spaces [15–17]. Other works such as [5, 18–20] use machine learning models to generate distractors by using a combination of the previous features and other types of information such as Term-Frequency Inverse-Document-Frequency (TF-IDF) scores.

While the current state-of-the-art in MCQ creation is promising, we see a number of limitations. First of all, existing models are often *domain specific*. Indeed, the proposed techniques are tailored to the application and distractor types. In language learning, such as vocabulary, grammar or tense usage exercises, typically similarity based on basic syntactic and statistical information works well: frequency, POS information, etc. In other domains, such as science, health, history, geography, etc., distractors should be selected on deeper understanding of context and semantics, and the current methods fail to capture such information.

The second limitation, *language dependency*, is especially applicable to factoids. Models should be agnostic to language because facts do not change with languages. Moreover, building a new model for each language could be daunting task as it would require enough training data for each language.

In this work, we study how the automatic retrieval of distractors can facilitate the efficient construction of MCQs. We use a high-quality large dataset of question, answer, distractor triples that are diverse in terms of language, domain, and type of questions. Our dataset was made available by a commercial organization active in the field of e-assessment (see Section 2.3.2), and is therefore representative for the educational domain, with a total of 62k MCQ, none of them identical, encompassing only 92k different answers and distractors. Despite an average of 2.4 distractors per question, there is a large reuse of distractors over different questions. This motivates our premise to retrieve and *reuse* distractors for new questions. We make use of the latest data-driven Natural Language Processing (NLP) techniques to retrieve candidate distractors. We propose *context-aware multilingual models* that are based on deep neural network models that select distractors by taking into account the context of the question. They are also able to handle variety of distractors in terms of length and type. We compare our proposed models to a competitive *feature-based* baseline that is based on classical machine learning methods trained on several handcrafted features.

The methods are evaluated for distractor quality using automated metrics and a real-world user test with teachers. Both the automatic evaluation and the user study with teachers indicate that the proposed context-aware methods outperform the feature-based baseline. Our contribution can be summarized as follows:

- We built three multilingual Transformer-based distractor retrieval models that suggest distractors to teachers for multiple subjects in different languages. The first model (Section 2.3.4.3) requires similar distractors to have similar semantic representations, while the second (Section 2.3.4.2) learns similar representations for similar questions, and the last combines the complementary advantages of these two models (Section 2.3.4.3).

- We performed a user study with teachers to evaluate the quality of distractors proposed by the models, based on a four-level annotation scheme designed for that purpose.

- The evaluation of our best model on in-distribution held-out data reveals an average increase of 20.4% in terms of recall at 10, compared to our baseline model adapted from [19]. The teacher-based annotations on language learning exercises show an increase by 4.3% in the

fraction of good distractors among the top 10 results, compared to teacher annotations for the same baseline. For factoid questions, the fraction of quality distractors more than doubles w.r.t. the baseline, with an improvement of 15.3%.

- We released[1] a test-set of educational questions of 6 subjects with 50 MCQs per subject and annotated distractors, and 77k size distractor vocabulary as benchmark to stimulate further research. The dataset, which is made by experts, contains multilingual and multi-domain distractors.

The remainder of the paper is organized as follows: Section 2.2 describes the relevant work in MCQs in general and distractor generation in particular. Section 2.3 introduces the dataset, explains the details of the proposed methods and the evaluation setup of the user study with teachers. In Section 2.5, the results of both the user study and automated evaluations is reported. And finally, in Section 2.6, we present the conclusion, lines for future work, and limitations of our proposed models.

2.2 Related work

2.2.1 MCQs in Education

Multiple choice questions (MCQs) are widely used forms of exercises that require students to select the best possible answer from a set of given options. They are used in the context of learning, and assessing learners' knowledge and skills. MCQs are categorized as objective types of questions because they primarily deal with the facts or knowledge embedded in a text rather than subjective opinions [21]. It has been shown that recalling information in response to a multiple-choice test question bolsters memorizing capability, which leads to better retention of that information over time. It can also change the way information is represented in memory, potentially resulting in deeper understanding [22] of concepts.

An MCQ item consists of three elements:

- *stem*: is the question, statement, or lead-in to the question.
- *key*: the correct answer.
- *distractors*: alternative answers meant to challenge students' understanding of the topic.

[1] https://dx.doi.org/10.21227/gnpy-d910 or https://github.com/semerekiros/dist-retrieval

For example, consider the MCQ in the first row of Table 2.3: the stem of the MCQ is *"Which inhabitants are not happy with Ethiopia's plans of the Nile?"*. Four potential answers are given with the question. Among these, the correct answer is *"Egyptians"*, which is the key. The alternatives are the distractors.

MCQs are used in several teaching domains such as information technology [23], health [24, 25], historical knowledge [26], etc. They are also commonly used in standardized tests such as GRE and TOEFL. MCQs are preferred to other question formats because they are easy to score, and students can also answer them relatively quickly since typing responses is not required. Moreover, MCQs enable a high level of test validity if they are drawn from a representative sample of the content areas that make up the pre-determined learning outcomes [25]. The most time-consuming and non-trivial task in constructing MCQ is distractor generation [3, 19]. Distractors should be plausible enough to force learners to put some thought before selecting the correct answer. Preparing good multiple-choice questions is a skill that requires formal training [27, 28]. Moreover, several MCQ item writing guidelines are used by content specialists when they prepare educational tests. These guidelines also include recommendations for developing and using distractors [29–31]. Despite these guidelines, inexperienced teachers may still construct poor MCQs due to lack of training and limited time [32].

Besides reducing teachers' workloads, the automation of the distractor generation could potentially correct some minor mistakes made by teachers. For example, one of the rules suggested by [29] says: "the length of distractors and the key should be about the same". Such property could be easily integrated in the automation process.

MCQs also have drawbacks; they are typically used to measure lower-order levels of knowledge, and guesswork can be a factor in answering a question with a limited number of alternatives. Furthermore, because of a few missing details, learners' partial understanding of a topic may not be sufficient to correctly answer a question, resulting in partial knowledge not being credited by MCQs [22]. Nonetheless, MCQs are still extensively utilized in large-scale tests since they are efficient to administer and easy to score objectively [2].

2.2.2 Distractor Generation

Many strategies have been developed for generating distractors for a given question. The most common approach is to select a distractor based on its similarity to the key for a given question. Many researchers approximate

the similarity between distractor and key according to WordNet [33–35]. WordNet [36] is a lexical database that groups words into sets of synonyms, and concepts semantically close to the key are used as distractors. The usage of such lexical databases is sound for language or vocabulary learning but not for factoid type questions. We instead provide a more general approach that could be used for both tasks, and instead of only using the key as the source of information while suggesting distractors, we also make use of the stem.

For learning factual knowledge, several works rely on the use of specific domain ontology as a proxy for similarity. Papasalouros *et al.* [8] employ several ontology-based strategies to generate distractors for MCQ questionnaires. For example, they generate “Brussels is a mountain" as a good distractor for an answer “Everest is a mountain" because both concept *City* and concept *Mountain* share the parent concept *Location*. Another very similar work by Lopetegui *et al.* [37] selects distractors that are declared siblings of the answer in a domain-specific ontology. The work by Leo *et al.* [10] improves upon the previous works by generating multi-word distractors from an ontology in the medical domain. Other works that rely on knowledge bases apply query relaxation methods, where the queries used to generate the keys were slightly relaxed to generate distractors that share similar features with the key [9, 38, 39]. While the methods in these works are dependent on their respective ontologies, we provide an approach that is ontology-agnostic and instead uses contextual similarity between distractors and questions.

Another line of works for distractor generation uses machine-learning models. Liu *et al.* [5] use a regression model based on characteristics such as character glyph, phonological, and semantic similarity for generating distractors in Chinese. Liang *et al.* [19] use two methods to rank distractors in the domain of school sciences. The first method adopts machine learning classifiers on manually engineered features (i.e., edit distance, POS similarity, etc.) to rank distractors. The second uses generative adversarial networks to rank distractors. Our baseline method is inspired by their first approach but was made to account for the multilingual nature of our dataset by extending the feature set.

There have also been a number of works on generating distractors in the context of machine comprehension [40]. Distractor generation strategies that fall in this category assume access to a contextual resource such as a book chapter, an article or a wikipedia page where the MCQ was produced from. The aim is then to generate a distractor that takes into account the reading comprehension text, and a pair composed of the question and its correct answer that originated from the text [41–43]. This line of work is

incomparable to our work because we do not have access to an external contextual resource the questions were prepared from.

In this paper, we focus on building one model that is able to suggest candidate distractors for teachers both in the context of language and factual knowledge learning. Unlike previous methods, we tackle distractor generation with a multilingual dataset. Our distractors are diverse both in terms of domain and language. Moreover, the distractors are not limited to single words only.

2.3 Methodology

In this section, we formally define distractor generation as a ranking problem; describe our datasets; describe in detail the feature-based baseline and proposed context-aware models including their training strategies & prediction mechanisms.

2.3.1 Task Definition: Distractor Retrieval

We assume access to a distractor candidate set $\mathcal{D}$ and a training MCQ dataset $\mathcal{M}$. Note that $\mathcal{D}$ can be obtained by pooling all answers (keys and distractors) from $\mathcal{M}$ (as in our experimental setting), but could also be augmented, for example, with keywords extracted from particular source texts. We formally write $\mathcal{M} = \{(s_i, k_i, \mathcal{D}_i) | i = 1, \ldots, N\}$. where for each item i among all N available MCQs, s_i refers to the question stem, k_i is the correct answer key, and $\mathcal{D}_i = \left\{d_i^{(1)}, ..., d_i^{(m_i)}\right\} \subseteq \mathcal{D}$ are the distractors in the MCQ linked to s_i and k_i. The aim of the distractor retrieval task is to learn a point-wise ranking score $r_i(d) : (s_i, k_i, d) \rightarrow [0, 1]$ for all $d \in \mathcal{D}$, such that distractors in $\mathcal{D}_i$ are ranked higher than those in $\mathcal{D} \setminus \mathcal{D}_i$, when sorted according to the decreasing score $r_i(d)$.
This task definition resembles the metric learning [44] problem in information retrieval. To learn the ranking function, we propose two types of models: feature-based models and context-aware neural networks.

2.3.2 Data

In this section, we describe our datasets, namely: (i) *Televic dataset*, a big dataset that we used to train our models. (ii) *Wezooz dataset*, a small-scale external test set used for evaluation.

Table 2.1: The statistics of our dataset

	Train	**Validation**	**Test**
# Questions	61758	600	500
# Distractors per question	2.4	2.3	2.3
Avg question length	27.8 tokens	28.1 tokens	27.6 tokens
Avg distractor length	2.2 tokens	2.3 tokens	2.1 tokens
Avg answer length	2.2 tokens	2.3 tokens	2.2 tokens
Total # distractors	94,205	-	-
Total # distractors $\leq$ 6 tokens	77,505	-	-

2.3.2.1 Televic dataset

This data is gathered through Televic Education's platform assessmentQ.[2] The tool is a comprehensive online platform for interactive workforce learning and high-stakes exams. It allows teachers to compose their questions and answers for practice and assessment. As a result, the dataset is made up of a large and high-quality set of questions, answers and distractors, manually created by experts in their respective fields. It encompasses a wide range of domains, subjects, and languages, without however any metadata on the particular course subjects that apply to the individual items.

We randomly divide our dataset into train/validation/test splits. We discard distractors with more than 6 tokens as they are very rare and unlikely to be reused in different contexts. We keep questions with at least one distractor. Table 2.1 summarizes the statistics of our dataset. The dataset contains around 62k MCQs in total. The size of the dataset is relatively large when compared to previously reported educational MCQ datasets such as SCiQ [45], and MCQL [19] which contain 13.7K and 7.1K MCQs respectively. On average, a question has more than 2 distractors, and contains exactly one answer. We use all the answer keys and distractors in the preprocessed dataset as the pool of candidate distractors (i.e., list of 77,505 filtered distractors) for proposing distractors for any new question.

The distractors in the dataset are not limited to single word distractors. More than 65% of the distractors contain two or more words as can be seen in Fig. 2.1a.

The data stems from multiple languages. Figure 2.1b shows the language distribution as detected by an off-the-shelf language classifier.[4] Given that Televic is a Belgian company, more than 50% of the questions are in Dutch,

[2]https://www.televic-education.com/en/assessmentq
[3]We used ISO 639-1:2002 standard for names of languages.
[4]We used the *langid* language classifier: https://github.com/saffsd/langid.py

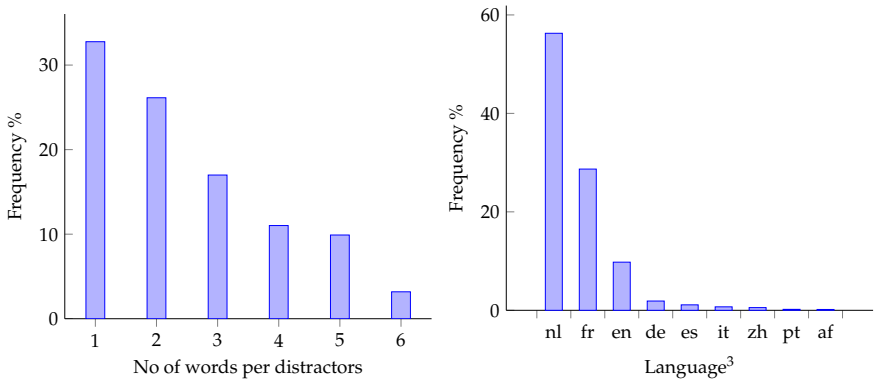


Figure 2.1: (a) distractor length in number of tokens and (b) language distribution for the Televic dataset.

while French and English are the next most common languages in the dataset.

Another dimension of the dataset is its domain diversity. It comprises questions about language/vocabulary learning (e.g., French and English) and factoids covering subjects such as Math, Health, History, Geography, and Sciences. Besides material from secondary school education, it covers materials from assessment tasks for professionals such as training in hospitals or manufacturing firms. The data is anonymized and contains no customer information.

Depending on the question type we observe different types of distractors. (1) Factoid distractors: names of people, locations, organizations, concepts, dates. (2) Distractors with numerical elements, such as multiples, factors, rounding errors, etc. (3) Language distractors: spelling, grammatical, tense, etc. However, the proposed models are agnostic of the type and origin of the data, and the automated evaluation on the Televic test set contains a random sample covering the different question types and origins (see Section 2.5.1). Note that although our dataset is a real-world commercial dataset, it only contains single-answer MCQs. However, the models we will put forward, could be readily extended towards multiple-answer MCQs, if such data were available.

2.3.2.2 WeZooz dataset

This data is a small-scale test set of questions gathered from WeZooz Academy,[5] which is a Flanders-based company providing an online platform with digital teaching materials for secondary school students and teachers. We selected four subjects; Natural sciences, Geography, Biology and History. Each subject was made to contain a fixed list of 50 questions that were randomly selected, and augmented with distractor annotations by teachers for these respective subjects (see Section 2.4). Note that this is an *external* test set, in the sense that the data distribution in the training set is not necessarily representative for this test set. This serves as a proof-of-concept for the general validity of our proposed method and models to specific use cases.

2.3.3 Feature-based Distractor Scoring

We built a strong feature-based model as our baseline. Feature-based models are a class of machine learning models that require a pre-specified set of handcrafted features as input. We designed 20 types of features capturing similarity between questions, answers, and the collection of candidate distractors. Formally, given a triplet (s, k, d) of question stem, key and distractor, our feature-based model first maps the input into a 20-dimensional feature vector $\phi(s, k, d) \in \mathbb{R}^{20}$, after which a classifier is trained to score the triplets according to compatibility of the question-answer-distractor combination. Our set of features can be segmented into four categories which are described below. A more detailed explanation of each feature can be found in Appendix 2.B.

(i) *Morphological Features*: this category contains features that are related to the form and shape of words that occur in our (s, k, d) triplets. This includes features such as edit distance, difference in token length, longest common suffix between k & d, etc.

(ii) *Static embedding based features*: We trained a Word2Vec model [46] on our dataset to learn static embeddings for the distractors. We treat distractors and answers attached to the same question as chunks sharing similar context. The objective is to learn a vector space in which their representations will also be closer. We leverage the embedding representations to extract several numerical features. For example, we calculate the cosine similarity and word mover's distance [47] between the embeddings of d & k.

[5] https://www.wezoozacademy.be/

(iii) *Language Prior*: since our data is multilingual we also calculate the prior probability of the candidate distractor matching with the language of the question, and attach it to each feature vector.

(iv) *Corpus-based Features*: this category contains features that are derived from the statistics of words in the corpus. It includes features such as the frequency of a distractor in the dataset and the inverse document frequency of distractors.

As classifier, we apply a *Logistic Regression (LR)* model to distinguish feature representations of actual question-answer-distractor triplets, present in the training, from triplets for which the distractor components belong to different question-answer combinations, sampled randomly. During training, the model's parameters are set to output high scores for actual triplets while the model is penalized for predicting high scores for others.

2.3.4 Context-aware Neural Distractor Scoring

Advanced context-aware neural models, unlike traditional feature-based models, do not require manual feature engineering. They have the ability to represent words depending on their semantic role and context in the considered text. In this work, we primarily focus on such context-aware models called *transformers* [48], which provide rich representations, and proved to achieve state-of-the-art results for many tasks in NLP such as question answering [49], machine translation [50], and text summarization [51]. A transformer is a deep neural network that uses a self-attention mechanism to assign importance weights to every part of the input sequence in how they are related to all other parts of the input. Transformers can scale to very large numbers of trainable parameters, stacked into very deep networks, which can still be trained very efficiently on parallel GPU hardware and thus learn from very large amounts of data. In NLP, such models are often trained on large unlabeled corpora to learn the inherent word and sentence level correlations (i.e., to model language structure) between varying contexts. This process is called *pretraining*, and downstream NLP tasks usually rely on such a pretrained generic model to be *finetuned* to their more specific needs instead of training a new model from scratch. Leveraging the knowledge gained during a generic pretraining process to improve prediction effectiveness for a specific supervised learning task, is a form of *transfer learning* [52, 53]. A common language task often used for pretraining transformer models called *masked language modelling* (MLM) requires masking a portion of the input text and then training a model to predict the masked tokens — in other words, to reconstruct the original non-masked input.

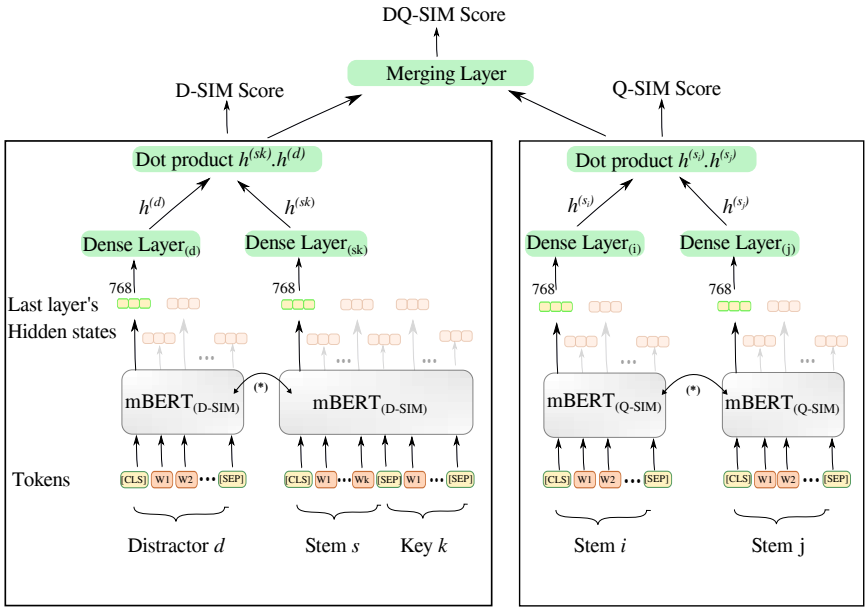


Figure 2.2: Our proposed context-aware distractor retrieval systems. For the D-SIM model (i.e., left), distractor d and concatenation of the stem s & key k separated by [SEP] are fed into the same mBERT$_{\text{(D-SIM)}}$ encoder, and then their respective vector representations at [CLS] are used as inputs to two different dense layers that do not share parameters. The outputs of these dense layers, $h^{(d)}$ & $h^{(sk)}$ are used to calculate the similarity between d & $s[sep]k$ using the dot product. Similarly for Q-SIM (i.e., right), two question stems i & j are encoded separately using the same mBERT$_{\text{(Q-SIM)}}$, and their respective [CLS] output vectors are fed into two different dense layers (i.e., dense layer$_{\text{(i)}}$ & dense layer$_{\text{(j)}}$) to produce their corresponding representations $h^{(s_i)}$ & $h^{(s_j)}$. These are used to calculate their similarity between the two stems using dot product. The DQ-SIM model (i.e., top) linearly combines the two models using a merging layer with an α parameter. $(\star)$ denotes parameter reuse by the encoders.

BERT (Bidirectional Encoder Representations from transformers) [54] is the most popular pretrained masked language model and has been widely used in many downstream tasks such as question answering & generation, machine reading comprehension, and machine translation, by fine-tuning it using a labelled dataset that provides supervision signal.

In this work, we present models to rank and retrieve distractors, based on such a pretrained transformer text encoder, which we finetuned by requiring similar distractors to have similar representations, and similar questions also to have similar representation. In the following paragraphs, we provide

a detailed description of these models, visualized in Fig. 2.2, followed by a description of the training procedure and the inference mechanism.

2.3.4.1 Distractor similarity based model (D-SIM)

We hypothesize that distractors co-occurring within the same MCQ item are semantically related through their link with the corresponding question stem and answer key. Following that hypothesis, the D-SIM model is designed (and trained) to yield a similar vector representation for a given (stem, key) pair (s_i, k_i), as each of the corresponding distractors d_i. Following the same logic, all candidate distractors $d \in \mathcal{D}$ can then be scored in terms of their similarity (in representation space) with a given new (stem, key) pair, after which the top candidates are returned by the model as likely valid distractors. We use the pretrained multilingual BERT (mBERT) encoder [54], followed by a fully connected linear layer (i.e., dense layer) to obtain initial representations for a (stem, key) pair, as well as for the distractors. We designed our model in a bi-encoder setting inspired by [55], and schematically shown on the left-hand side of Fig. 2.2. The distractor d is fed into the mBERT encoder, and the output representation of the $[CLS]$ token[6] is used as an input to the dense layer. The output from the dense layer is taken as the corresponding representation h_d. The considered stem and key are concatenated into a single sequence of tokens[7] as "s_i $[SEP]$ k_i", which is fed into the *same* mBERT encoder (i.e., with parameter reuse, as indicated by the double arrow in Fig. 2.2). Similar to the distractor embedding, we take the [CLS] token representation and feed it to the dense layer (i.e., *different* dense layer with no parameter sharing), and take its output as the vector representation of the key-aware stem h_{sk}. Finally, the similarity score between (s_i, k_i) and d is obtained as the dot product between their respective representations:

$$r_i^{\text{D-SIM}}(d) = h_i^{(sk)} \cdot h^{(d)}$$

During training, the encoder is fine-tuned to achieve higher scores for compatible stem/key and distractor combinations, and lower scores for incompatible ones (as described in Section 2.3.5 in more detail).

2.3.4.2 Question similarity based model (Q-SIM)

This model is based on the assumption that different questions that share one or more distractors or answer keys are likely semantically related, such

[6] [CLS] is a special token that is prepended to the input, and its corresponding output representation is pretrained to represent the entire sequence that is used for classification tasks.

[7] The often used [SEP] token is a special token known by the model, that separates input sentences.

Table 2.2: Q-SIM training data examples.

Distractor/Answer	Associated Questions	Description
koolhydraten en vetten	1. Welke groepen voedingsstoffen leveren vooral energie? 2. Welke voedselcomponenten kunnen stoffen leveren die zowel bij assimilatie als bij dissimilatie in cellen worden gebruikt?	Factoid questions with multi-word distractor in Dutch.
surrounded	1. The guest house is … on the countryside. 2. The valley was … by forests.	A fill in the gap question for English language learning.
Marokko	1. Welk land is in 2011 gesplitst door het langdurig conflict in Darfur ? 2. Rabat is de hoofdstad van …	A combination of fill-in the gap and normal questions

that their associated distractors could be used as good candidate distractors for one another. To accomplish this, we first rearrange the training data in such a way that these questions, sharing at least one distractor or key, are clustered together (see Table 2.2 for an example). Then, we train our Q-SIM model to produce similar representation for question stem pairs (s_i, s_j) that are in the same cluster. The right-hand side of Fig. 2.2 depicts the Q-SIM model, again based on a bi-encoder architecture. The stem representation $h_i^{(s)}$ for a question MCQ_i is again obtained through an mBERT encoder, followed by a fully connected linear layer, similarly to $h_i^{(sk)}$ but ignoring the question key. The Q-SIM scoring function is defined as

$$r_i^{\text{Q-SIM}}(d_j) = h_i^{(s)} \cdot h_j^{(s)}$$

and can be interpreted as follows. For a given question MCQ_i, its stem representation $h_i^{(s)}$ is compared through dot product similarity with the representation of any candidate distractor d_j originating from a question MCQ_j. The particular representation of d_j assumed in Q-SIM is in fact MCQ_j's stem representation $h_j^{(s)}$. Note that Q-SIM does not allow making a distinction in terms of score between different distractors from the same MCQ. Candidate distractors with the same score are considered equally likely according to Q-SIM, and ranked in an arbitrary order. Based on the intuition outlined above, more complex formulations for Q-SIM can be designed, for example with a feature characterizing the nature of the pairwise comparison (i.e., the actual answers of the considered questions, two of their respective distractors, or the answer for the one and a distractor for the other). However, given the already significant improvement of the presented basic Q-SIM formulation (see Section 2.5.1), we chose to include only that model in our evaluation. In fact, its simple intuitive formulation

makes it straightforward to explain to teachers, which is an important aspect in their trust in the model [56].

2.3.4.3 Distractor and Question similarity model (DQ-SIM)

This model combines the previous two models using a merging layer (visualized on top of Fig. 2.2), based on the intuition that a well-chosen combined model may benefit from the complementary advantages of both individual models. This merging layer combines the outputs from D-SIM and Q-SIM using a merging parameter α, to control the contribution of the individual models. We investigated empirical score-based and rank-based merging strategies. The score-based model assumes a linear combination of both respective *scores* $r_i^{\text{D-SIM}}$ and $r_i^{\text{Q-SIM}}$ from D-SIM and Q-SIM, in which their individual contribution is controlled by the hyperparameter α:

$$r_i^{\text{DQ-SIM-score}}(d) = \alpha\, r_i^{\text{D-SIM}}(d) + (1-\alpha)\, r_i^{\text{Q-SIM}}(d)$$

The rank-based model combines the distractor *ranks* $\rho_i^{\text{D-SIM}}$ and $\rho_i^{\text{Q-SIM}} \in \{1, 2, 3, ..., N\}$ from D-SIM and Q-SIM into the score

$$r_i^{\text{DQ-SIM-rank}}(d) = \frac{\alpha}{\log\left(\rho_i^{\text{D-SIM}}(d)+1\right)} + \frac{1-\alpha}{\log\left(\rho_i^{\text{Q-SIM}}(d)+1\right)}$$

This scoring function is based on weighted combination of inverse distractor rankings, such that high-ranked distractors have more weight. We use logarithmic smoothing to avoid the potential contribution of low-ranked distractors from vanishing too rapidly.

2.3.5 Training

We use *contrastive learning* as our training strategy [57]. Contrastive learning [46, 58, 59] is a machine learning technique that aims to learn representations of data by contrasting similar and dissimilar examples. It aims to bring similar instances closer together in the representation space by maximizing the similarity between their embeddings, while pushing dissimilar samples further apart by minimizing their similarity.

In a contrastive learning setting, it is often the case that similar example pairs (i.e., also called positive examples) are available explicitly in training datasets, whereas dissimilar or negative examples need to be sampled from an extremely large pool of instances. For the Q-SIM model, a positive pair consists of two questions sharing at least one distractor, whereas for the D-SIM model, we require similar representations for a given (stem, key) item and a distractor corresponding to the same MCQ.

As a negative sampling strategy, we use in-batch negatives [60] while training our models. For D-SIM, the in-batch negatives are gold-standard positive distractors for the other instances in the same batch. While for Q-SIM, the in-batch negatives are the positive questions that come from the other instances in the same batch. Reusing gold standard distractors or questions from the same batch as negatives makes training more efficient, compared to randomly sampling negatives for each positive pair in the batch.

With the notation $r_i(d)$ (common in both D-SIM and Q-SIM) for scoring MCQ_i against distractor d, and by introducing the sigmoid function $\sigma(r) = 1/(1+e^{-r})$, we can write the contrastive loss [61] $\mathcal{L}_i$ to be minimized for MCQ_i with matching distractors d^+ as follows:

$$\mathcal{L}_i = -\sum_{d^+} \log \sigma\big(r_i(d^+)\big) - \sum_{d^-} \log \sigma\big(-r_i(d^-)\big)$$

in which $r_i(d^+)$ denotes the score of a positive distractor for the considered question, and $r_i(d^-)$ the scores for the in-batch negatives (summed over the considered batch of training instances). If the quantity $\sigma\big(r_i(d)\big)$ is interpreted as the probability that distractor d is compatible with MCQ_i (in the sense of model D-SIM or Q-SIM), then minimizing the above loss term can be understood as maximizing the joint estimated probability of d^+ being compatible distractors for MCQ_i, and the in-batch negatives d^- to be incompatible ones.

2.3.6 Using the models for predictions

This section describes the inference mechanism for our models. Inference refers to using a trained model to make predictions about new data. For each of the models, the goal is inducing an ordering of all candidate distractors in response to a given question stem and answer key, such that the top ranked ones can be proposed as fitting distractors.

For the D-SIM model, since the considered (stem, key) pair and the distractor to be scored against it are independently fed to the network, the embeddings of the pool of distractors can be computed offline. The vector representation $h^{(sk)}$ of a given stem and its answer key is calculated, compared through the dot product with each of the pre-calculated distractor representations $h^{(d)}$, and these are then ordered according to decreasing score.

Similarly, for the Q-SIM model, the pool of questions' embeddings is calculated offline and stored. At run time, for a given question stem s, we compute its embedding $h^{(s)}$, score it against all pre-calculated stem representations for the MCQs in the corpus, and rank the candidate distractors

according to the decreasing score of their corresponding question stem. Note that we assign that same score to each of the distractors of a given stem (for use in DQ-SIM-score). We then rank all distractors according to decreasing scores (randomly ordering those with identical scores).

Finally, once the scores for D-SIM and Q-SIM are calculated for each candidate distractor, the DQ-SIM model can be evaluated directly, by ranking them according to the decreasing score $r^{\text{DQ-SIM-score}}$ or $r^{\text{DQ-SIM-rank}}$.

2.4 Experimental Design

This section describes the evaluation methodology and the metrics we used to measure the quality of the generated distractors using the different methods described in Section 2.3. Section 2.4.1 introduces our hypotheses and the experiments we designed to test them. The automatic evaluation metrics we used are explained in Section 2.4.2.

2.4.1 Evaluation Setup

In order to validate our models' theoretical effectiveness and practical applicability, we formulate the following three key hypotheses, which we will test through experiments based on both automatic and human annotator evaluation:

- **Hypothesis 1**: *Context-aware models, based on generic pre-trained language models, lead to more effective distractor selection models than shallow prediction models based on manually engineered features.*

- **Hypothesis 2**: *Manual distractor quality scores are correlated with machine-generated distractor candidate rankings.*

- **Hypothesis 3**: *Top-ranked machine-proposed distractor candidates are comparable in quality to expert-generated distractors, for a given question stem and answer key.*

For Hypothesis 1, we first of all set up a large-scale automatic evaluation experiment with the Televic dataset (see Table 2.1). In addition, a focused small-scale automatic evaluation of context-aware and feature-based models was carried out on the WeZooz external data (see Section 2.3.2.2 for details) that contains several subjects.

We complemented that automatic evaluation with human evaluation, since hard comparison of ground-truth distractors with machine-generated distractors may not give the whole picture of accuracy. Indeed, both for

language learning and factual knowledge learning, MCQs can have a potentially large set of viable distractors that are not included by the gold standard distractor set. Thus, automated metrics could flag a correctly proposed candidate distractor as wrong because of the scarcity of the gold standard dataset. To avert this problem, many previous works asked human experts to judge the quality of the distractors that were generated by their systems [62, 63]. Hence, we also invited teachers to provide their expert opinion, each focusing solely on a set of questions on their own subject of expertise. In the following paragraphs, we explain the procedure we followed to set up that expert evaluation, which we will use in assessing all aforementioned Hypotheses 1–3.

First, we prepared a small sample of test questions for language and factual knowledge learning. For language learning, we used French and English. These questions were randomly drawn from the held-out test split of the Televic dataset introduced in Section 2.3.2.1. For the factoid type questions, we use the WeZooz dataset introduced in Section 2.3.2.2. Each of the subjects contains a fixed list of 50 questions. Second, we applied the different trained models to rank distractors according to their relevance for each question in the test set. We then kept the top-10 ranked candidate distractors for each of the models. Finally, teachers were shown distractor predictions unified over all models (i.e., duplicates were removed) as well as the provided gold-truth distractors for each test question (see the illustration provided in Fig. 2.4 in Appendix 2.C). Note that the order of the unified list of distractors was randomized, to avoid introducing order bias.

The teacher participants were explicitly instructed to rate each candidate distractor based on how much they thought it would help them if they were given the task of preparing distractors for that specific question. Specifically, we asked them to annotate each distractor independently of the other distractors in the list, based on a four-level annotation scheme that we designed to measure the quality of distractors. Our scale is closely related to the three-point evaluation scale proposed by [63] (Table 2.3 shows examples of each category):

- *True Answer*: specifies that the distractor partially or completely overlaps with the answer key.
- *Good distractor*: specifies that the distractor is viable and could be used in an MCQ as is.
- *Poor distractor*: specifies that the distractor is on topic but could easily be ruled out by students. This could happen due to one or both of the following reasons.

  - *Poor meaning*: the distractor has poor meaning. For example, it is too general, although not completely off-topic.
  - *Poor format*: the distractor's format is different from the format of the answer key and does not fit with the stem.

- *Nonsense distractor*: specifies that the proposed distractor is completely out of context.

Although the third category (i.e., poor distractor) implies that the proposed distractor is ineffective as is, a minor tweak may result in a useful distractor. Furthermore, even if a significant change is required, it may inspire teachers to create new effective distractors.

Using the annotations we gathered from the teachers, we tested Hypotheses 2 and 3. For *Hypothesis 2*, we evaluated whether the higher ranked distractors also have a higher perceived usefulness. This was done by comparing the human scoring of distractor candidates in the top-5 to that of those ranked 5–10: for a good distractor generation model, the top-5 should on average contain significantly more 'good' ones. We designed a statistical analysis to test the null hypothesis that the rating distribution is not related to whether candidate distractors were ranked top-5 or 5–10. We used Fisher's exact test[8] to test this hypothesis.

For *Hypothesis 3*, we evaluated the extent to which the teachers perceived the system-generated distractor candidates as the ground-truth distractors. Again, we use Fisher's exact test to test the null hypothesis that the distribution of quality of distractors is not related to whether the distractors are human-generated or system-generated.

2.4.2 Automated Metrics

We use two groups of information retrieval metrics to automatically evaluate our systems: (1) Order-unaware metrics: Recall@k and Precision@k, which measure the fraction of gold-standard distractors that are in the top-k distractors and the fraction of relevant distractors in the top-k retrieved distractors, respectively. (2) Order aware metrics: mean reciprocal rank (MRR) and mean average precision (MAP), which respectively reflect how high the most relevant item is ranked in the list, and how high all relevant ones are ranked on average.

[8] We also conducted a chi-square test and reached the same conclusions.

Table 2.3: Annotation scheme examples

Question	Answer	Distractors	Category	Moderation
Which inhabitants are not happy with Ethiopia's plans of the Nile?	Egyptians	1. Itali	Poor format	because of wrong spelling.
		2. Kenyans	Good	-
		3. gypsies	Poor meaning	because
My mum brought the washing in .... it was raining.	because	1. until	Good	-
		2. since	True Answer	
		3. investigate	Nonsense	out of context
How old was Beethoven when he died?	56 years	1. 1.5v	Nonsense	out of context
		2. 60 years	Good	-
		3. 180 years	Poor meaning	humans cannot live 180 years.

2.5 Results and Discussion

In this section, we provide evidence of the effectiveness of our context-aware models by reporting the experimental results and discussing the insights gained. Section 2.5.1 compares the baseline with our proposed context-aware models using reproducible automated metrics (Hypothesis 1). Section 2.5.2 discusses the user study results with experts (Hypotheses 1–3). Note that all the numerical results reported in this section are in percentage points.

2.5.1 Automatic Evaluation

When considering the results of our automated evaluation based on the recovery of ground-truth distractors, it is essential to note that information about ground-truth distractors for a given item was never used during the model's training. Table 2.4 shows the large-scale evaluation of the systems on the Televic test set. We report our results as the mean and standard deviation of five different runs of our models using five random seeds as shown in Table 2.4. All three context-aware models consistently outperform our feature based model (denoted 'baseline') on all metrics. DQ-SIM performs the best according to most metrics, confirming that Q-SIM and D-SIM have their own (complementary) merits. Q-SIM is better than D-SIM at recovering ground truth distractors (i.e., Recall@10 of 82.3 compared to 76.0), but inferior at ranking the best relevant distractor at the top in the list, which we conclude from the lower Precision@1 (40.4 vs. 44.9) and MRR (55.6 vs. 60.7) scores. This is related to the nature of the Q-SIM model. The candidate distractors belonging to its best matching

question would be put at the top of the returned distractors in a random order. Our results show that D-SIM is better at estimating the most likely distractor than Q-SIM is in finding a relevant question *and* arriving with the relevant distractor on top after random ordering. However, the Precision@4 results show that Q-SIM has more success in identifying a question with good distractors, than D-SIM has in detecting good distractors among its top 4 results. The other reported metrics (Recall@10, Precision@4, MAP) indicate the overall higher effectiveness of Q-SIM when looking further than only the top result. In our MCQ generation setting, recall within the top 10 results is the more important metric, since the presence of high quality distractors in the automatically generated list is more important than their correct ranking.

Table 2.4: Automatic ranking evaluation Full-ranking

Models	**R@10**	**P@1**	**P@4**	**MAP**	**MRR**
Baseline	71.3±1.2	21.1±1.8	23.7±0.5	33.5±1.0	43.9±1.9
D-SIM	76.0±0.7	**44.9**±0.5	24.4±0.8	44.9±0.6	60.7±1.3
Q-SIM	82.3±0.5	40.4±1.5	35.9±0.9	54.9±0.9	55.6±1.1
DQ-SIM	**91.7**±0.6	41.9±0.8	**38.2**±0.7	**57.3**±0.5	**62.8**±0.4

R:recall, P: precision, MAP: mean avg. precision,
MRR: mean reciprocal rank; evaluation on Televic test set.

Figure 2.3 depicts the performance of DQ-SIM for the two merging strategies, in terms of Recall@10 on the validation set described in Section 2.3.4.3. The linear combination of the scores outperforms the rank-based merging strategy. The score-based strategy achieves the best performance at $\alpha = 0.8$, giving more weight to the Q-SIM model. This is reasonable given that the Q-SIM model outperforms the D-SIM model on the recall metric.

Table 2.5 compares the baseline with DQ-SIM (i.e., the best context-aware model according to the evaluation on the Televic dataset) in a small-scale setting for all four Wezooz dataset subjects as well as English and French from the Televic test set. Since we want to compare models in terms of their ability to rank relevant distractors higher in the list, we added the ground-truth distractors from all the subjects to the existing distractors pool. Ideally, the best model would rank all the ground-truth distractors high in the list. Similar to the large-scale evaluation, DQ-SIM consistently outperforms the baseline for all subjects on both metrics. Recall@10 and MAP are higher for the language category than for factoid questions because the test questions for the former come from the same distribution (i.e., Televic test questions) as the data the models were trained on (i.e., Televic train set). On the

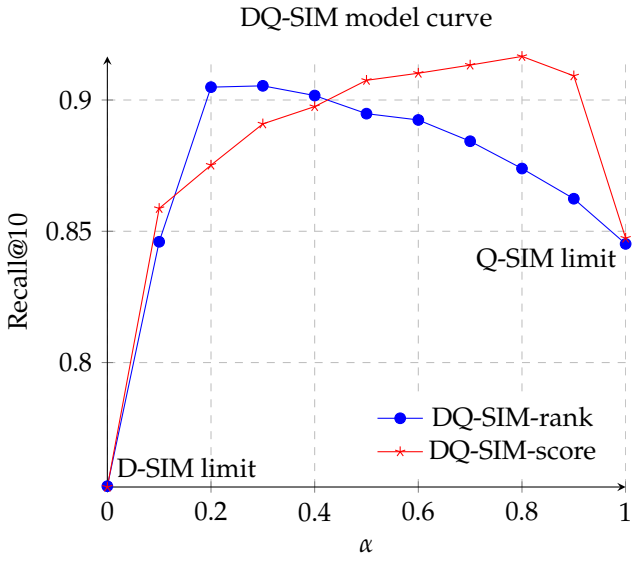


Figure 2.3: Different α values for combining Q-SIM and D-SIM models using rank and raw scores on the validation set

other hand, the test data for the factoids come from a different distribution (i.e., WeZooz dataset) than the training data such that the evaluation for these subjects additionally measures the robustness of the model to a data distribution shift (i.e., its domain transfer abilities). The DQ-SIM model is far more robust than the baseline.

Table 2.5: Small scale Automatic ranking evaluation

Models	**Baseline**		DQ-SIM	
	R@10	**MAP**	**R@10**	**MAP**
English*	60.1	33.6	**98.3**	**85.8**
French*	46.6	17.7	**81.1**	**61.1**
Nat. Sciences	24.3	7.7	**74.3**	**37.3**
History	14.3	3.4	**62.2**	**35.7**
Biology	30.6	7.6	**72.0**	**41.8**
Geography	32.3	12.1	**61.5**	**34.4**

R: recall, MAP: mean avg. precision; * denotes subject
is drawn from the Televic test set, while the rest are from WeZooz.

Table 2.6: Inter-annotation agreement of ground-truth distractors (%)

	True Ans.	Good	Poor	Nonsense
Languages	5	70	14	11
Factoids	2	83	9	6
Overall	3	79	11	7

2.5.2 Expert Evaluation

Following the procedure introduced in Section 2.4, a total of 12,723 ratings for distractor quality were gathered from the annotations by teachers (see Table 2.9 for details of rating statistics). These ratings come from the top-10 ranked distractors for each of the four models, and the ground-truth distractors (i.e., all simultaneously presented and randomly shuffled). We retained the gold standard distractors in the lists to be annotated, because we wanted to investigate the agreement among teachers in creating distractors. In the following subsections, we study teachers' (dis)agreement on the quality of distractors, compare the various models using the evaluation from experts, and revisit Hypotheses 1–3 in light of these results.

2.5.2.1 Inter-annotator agreement

We adopt 2 strategies to assess inter-annotator agreement. First, we analyze how teachers rated the ground-truth distractors, which were made by other teachers who prepared the questions. As can be seen from Table 2.6, in general, we find that 79% of the actual distractors were deemed good, 11% poor, followed by 7% nonsense and 3% true answers. There is greater agreement between teachers in what is considered a good distractor on the factoids than for language learning exercises (83% vs. 70%).

Second, we study the agreement of teachers by asking them to rate the same set of distractors using our four-level scale annotation scheme. We selected the subjects English, from the languages category, and History, from factoids, for annotations by at least two teachers. Table 2.7 shows the inter-annotator agreement of teachers using the Jaccard similarity coefficient. The Jaccard similarity measures similarity between two sets of data by calculating what fraction of the union of those datasets is covered by their intersection. In our case, it is calculated as the number of times the teachers agreed on a distractor category label (i.e., one of the four quality labels) divided by the total number of distractors that were annotated (by either

Table 2.7: Inter-annotation agreement of experts in terms of Jaccard similarity coefficient (%)

Subjects	True	Good	Poor	Nonsense	Overall
English	25.8	42.9	12.8	40.0	47.9
History	0.0	43.6	24.3	59.7	57.7

annotator) with that label. In general, we note a higher agreement on what is considered a good distractor and a nonsense distractor. Particularly, the overall agreement between the History teachers is higher than the English teachers. This is in line with the higher agreement for factoid type questions discussed in the previous paragraph. The Jaccard similarity is sensitive to small sample sizes. For example, a total of only two distractors were rated 'true answer' by the history teachers which yielded no similarity (i.e., a '0' in the first column in Table 2.7).

Calculating the inter-annotator agreement with the commonly used Cohen's kappa [64] value, we confirm aforementioned higher agreement for factoid questions than for language: Cohen's kappa is 29.3 among English teachers, which represents "fair agreement", and 40.5 among History teachers, indicating "moderate agreement".

As a final metric to assess potential ambiguity in scoring distractors, we calculate conditional probabilities $P(X|Y)$ of having a second annotator assigning label X given that a first one said Y. For example, unsurprisingly, the probability of rating a distractor 'good' given that it was rated 'nonsense' by another teacher and vice-versa was 6% for English and 5% for History. This implies that the confusion in differentiating good distractors from nonsense distractors was minimal. Details are presented in Table 2.12 in Appendix 2.D.

2.5.2.2 Evaluation of models by experts

Table 2.8 shows the expert evaluation of distractors in terms of *good distractor rate* (GDR@10) and *nonsense distractor rate* (NDR@10). GDR@10 is calculated as the percentage of distractors that were rated 'good' among the top 10 ranked distractors for each model. Similarly, NDR@10 is calculated as the percentage of distractors that were rated 'nonsense' among the top 10 ranked distractors for each model. We are interested in reporting the NDR metric because (i) it could be used to distinguish between good and bad systems, and (ii) in a real-world scenario discarding a system with

Table 2.8: Expert evaluation of distractors (%)

Models	Language learning		Factoid learning	
	GDR@10 ↑	NDR@10 ↓	GDR@10 ↑	NDR@10 ↓
Baseline	23.6	45.4	13.6	66.0
D-SIM	25.9	45.2	15.0	64.8
Q-SIM	26.3	45.3	19.0	61.6
DQ-SIM	**27.9**	**44.6**	**28.9**	**50.1**

GDR: good distractor rate, NDR: nonsense distractor rate;
↑: higher is better, ↓: lower is better; evaluation on WeZooz test set

high NDR score could be helpful since the frequent occurrence of nonsense distractors may scare away users by eroding their trust in the model. The reported metrics are averages of all the subjects in each category. ↑ indicates larger values are better and ↓ indicates smaller values are better. In general, context-aware models were rated better in proposing plausible distractors than the baseline model. They also produced fewer nonsense distractors. The DQ-SIM outperformed all the other models. On average, 3 out of its top 10 proposed distractors were rated good distractors. Moreover, on average 5.5 distractors for languages and 5 for factoids were generally found on-topic (i.e., distractors rated as either good or poor distractors) for DQ-SIM.

The NDR@10 is lower for all models for language subjects than for factoid questions. We hypothesize this is because the test data for the language category comes from the same distribution the models were trained on.

2.5.2.3 Discussion of key hypotheses

We now discuss to what extent our experimental results confirm our aforementioned key Hypotheses 1–3.

Hypothesis 1 states that the context-aware models generate better quality distractors than the feature-based models. As discussed in Section 2.5.1, the automated evaluation shows that the context-aware models consistently outperform the feature-based model on the Televic and WeZooz datasets. The human evaluation in Section 2.5.2.2 further confirms this by demonstrating that distractors generated by context-aware models were rated higher in quality than those generated by feature-based models.

Hypothesis 2 states that human distractor quality ratings are correlated

with the automated candidate distractor rankings. To test this hypothesis, we collapsed the four ratings into two categories: *plausible* (i.e., rated as good distractors) and *less plausible* (i.e., rated as true answer, bad and nonsense distractors). Table 2.10 in Appendix 2.D shows the contingency table for Fisher's exact test for our best model, i.e., DQ-SIM. The fraction of top-5 ranked distractors that received 'good distractor' ratings (i.e., 30.3%) is higher than that for the ones ranked 5–10 (i.e., 21.6%). We found that this difference is statistically significant. Indeed, the null hypothesis that the automatic ranking of distractors is unrelated to how teachers rated them is strongly rejected (p =1.7e-8).

Hypothesis 3 asserts that the quality of top-ranked machine-generated distractors is comparable with human-made distractors. To test this, we compare the distribution of ratings of the ground-truth distractors (i.e., expert-generated distractors) with the distribution of ratings for the DQ-SIM model (i.e., system-generated distractors). As for Hypothesis 2, we collapse the ratings into *plausible* and *less plausible* classes. Table 2.11 in Appendix 2.D shows the contingency table for Fisher's exact test, to compare the quality between system-generated and human-generated distractors. The null hypothesis that the source of the distractor (i.e., human-generated or system-generated) is unrelated to the quality label assigned by the teachers, is strongly rejected (p <1.e-10). Indeed, the quality of the human-generated distractors was found to be better than the system-proposed distractors. Still, we believe system-generated distractors have value: given that they can be generated quickly and automatically, presenting them as suggestions — rather than relying on a fully automated system — seems a practically meaningful way of working, which could save teachers a significant amount of time (compared to purely creating a list of distractors without any assistance).

2.6 Conclusion and Future Work

This paper introduced and evaluated multilingual context-aware distractor retrieval models for reusing distractor candidates that can facilitate the task of MCQ creation. Particularly, we proposed three models: (1) The D-SIM model that learns similar contextual representations for similar distractors, (2) The Q-SIM model that requires similar questions to have similar representations, and (3) The DQ-SIM model that linearly combines the previous two models benefiting from their respective strengths. Importantly, the DQ-SIM model showed a considerably reduced nonsense distractor rate, which we consider a useful asset in terms of trust in the model by teachers. We also asked teachers to evaluate the quality of distractors using a

four-level annotation scheme that we introduced. As the result, teachers considered 3 out of 10 suggested distractors as high-quality, to be readily used. Additionally, they found two more distractors to be within topic, albeit of lower quality, and useful as inspiration for teachers to come up with their own good distractors. Finally, we released a test consisting of 298 educational MCQs with annotated distractors covering six subjects and a 77K distractor vocabulary to promote further research.

In future work, we foresee three directions. First, it is worth reiterating that the current work assumes access to a substantial pool of distractors. Even though with such large item pools, it is expected that many options are available for an incoming newly written question, the current work is unable to generate a brand new distractor. A possible solution could be to employ pure generative models that can freely generate distractors. Moreover, generative models could correct the 'poor format' errors. However, it has to be noted that such models require access to a context where the distractors and questions come from, such as a chapter of a book, Wikipedia article, etc. A second research direction is to extend the current work to a multimodal system that considers other sources of information, e.g., images that accompany MCQs in digital learning tools. Finally, an area that we are currently investigating is how to make sure the complete list of distractors in a single MCQ is sufficiently diverse: note that in the present study, we were only interested in retrieving a list of plausible distractors independent of each other. However, typical MCQ distractors should not only be plausible but also sufficiently diverse.

Acknowledgments

This work was funded by VLAIO ('Flanders Innovation & Entrepreneurship') in Flanders, Belgium, through the *imec-icon* project AIDA ('AI-Driven e-Assessment'). This research also received funding from the Flemish Government under the "Onderzoeksprogramma Artificiële Intelligentie (AI) Vlaanderen" programme. We would like to thank the AIDA partners Televic Education and WeZooz Academy for contributing data and use cases, as well as the ZAVO ('Zorgzaam Authentiek Vooruitstrevend Onderwijs') secondary school teachers for participating in the study.

2.7 Appendix

2.A Training and Implementation details

- **Feature-based models**: for feature extraction and model training we use components from the scikit-learn package for python [65]. As negative training examples, we sample a total of 100 non-distractors for each MCQ.

- **Context-aware models**: our transformer based model is implemented using Pytorch [66] and Huggingface [67]. We initialize our encoder with `bert-base-multilingual-uncased`. We fine-tune the last two layers and leave the other layers frozen. The most important hyperparameters are the learning rate, batch size, the duration of training, and the output width of our dense layer. To avoid extensive hyperparameter tuning, we made the following choices. First, we choose the output dimension of the dense layer to be $d_{out} = 700$ because we empirically found that it yielded good results. For the learning rate, we kept the choice of 10^{-5} from Karpukhin *et al.* [60] in combination with the robust Adam optimizer [68]. Also in line with [60], we know increasing batch size may lead to slightly improved results, and thus decided the batch size to be 64, the highest value that would fit our V100 memory. We train each model for 25 epochs at which point performance on the development begins to plateau due to overfitting.

2.B Feature Vector Description

We describe each feature we used to build our feature-based classifiers in below.

1. `tfidf_word_match_share` : a word overlap metric between both k & d and s & d which weighs overlapping words according to their inverse document frequency value.

2. `word_match_share`: fraction of word tokens that are shared between both k & d and s & d.
3. `equal_num` : boolean feature that checks whether k & d have equal numbers of digits.
4. `longest_substring` : fraction of longest matching sub-string between k and d.
5. `token_len_sim` : boolean feature that checks if the amount of tokens in k is equal with d.
6. `token_len_diff` : difference in amount of tokens in k and d.
7. `char_len_sim` : boolean feature that checks if the amount of characters in k is equal with d.
8. `char_len_diff` : difference in amount of tokens in k and d.
9. `is_caps` : boolean feature that checks if both k and d are capitalized.
10. `count_caps` : boolean feature that checks if both k and d have the same number of upper cased characters.
11. `has_num` : boolean feature that checks if the strings k and d have numbers.
12. `get_count` : absolute number of occurrences of d in our dataset.
13. `first_char_match` : boolean feature that checks if both k and d start with the same 5-gram characters.
14. `last_char_match` : boolean feature that checks if both k and d end with the same 5-gram characters.
15. `w2v_ad_sim` : a numeric feature that calculates the cosine similarity between the answer key and distractor using their word2vec representations.
16. `wmd_w2v_qd` : word mover's distance between the question and distractor using their word2vec vector representations.
17. `wmd_w2v_ad` : word mover's distance between the answer and distractor using their word2vec vector representations.
18. `glove_ad_sim` : the cosine similarity between the answer and distractor using their averaged glove embeddings.
19. `wmd_glove_ad` : the word mover's distance between the answer and distractor using their averaged glove embeddings.
20. `lang_prior` : the prior distribution of the source language of the question.

2.C Annotation Platform

Figure 2.4 shows the annotation tool that we built for the quality annotation task by teachers. Each page presents a question, its actual answers, and a randomly shuffled list of candidate distractors that are proposed by the different models. Teachers assign quality labels to each of these proposed distractors by selecting one of the four radio-button options. If the teacher selects *poor distractor* as a label for a distractor, then a drop-down menu with

two more options (i.e., *poor format* and *poor meaning*) is shown. Finally, the annotator/teacher can go to the following question by pressing the 'Next' button displayed at the left bottom of the screenshot.

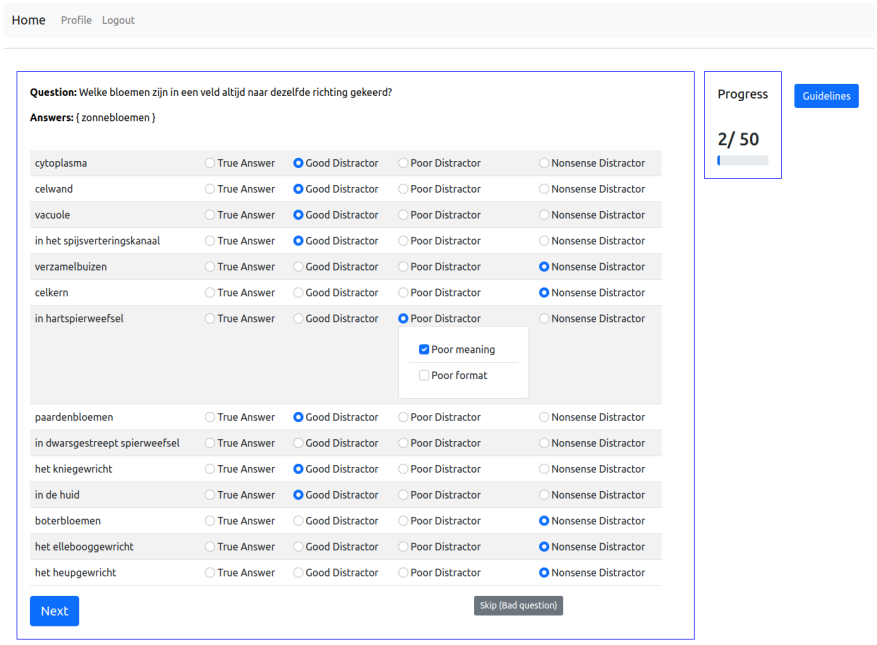


Figure 2.4: Screenshot of the distractor annotation tool. The teacher is shown a question, an answer, and a shuffled list of ground-truth distractors & candidate distractor suggestions by all the models.

2.D User study details

This section contains the user study details. Table 2.9 describes the data gathered from the annotations provided by the teachers. Every subject has 50 questions except English, which had two duplicates that were later removed, leaving only 48 questions. On average, there are 2 distractors for each question item. We collected 1090 annotations for the original ground-truth distractor, and 11,633 annotations for the proposed candidate distractors (i.e., top ten ranked distractors by each of the four models). A total of 8 teachers participated in the study. English (i.e., from languages) and History (i.e., from factoids) were annotated twice by two different teachers for the purposes of calculating interannotator agreement.

Table 2.10 shows the contingency table for hypothesis 2 that tests the correlation between automated distractor rankings (i.e., using our best

Table 2.9: Ratings Data Description

Subjects	Item count	Dist. count		Ratings count		No of Raters
		Original dist	Proposed dist	Gold dist	Proposed dist	
English	48	130	723	260	1882	2
French	50	92	1148	92	1650	1
Geography	50	145	966	290	1960	1
History	50	130	1335	260	2420	2
Biology	50	88	1266	88	1761	1
Nat. Sciences	50	100	1407	100	1960	1
Total	298	685	6845	1090	11633	8

Table 2.10: Contingency table for automatic ranking & human rating correlation using DQ-SIM

	Plausible	**Less plausible**
Ranked top 5	425	977
Ranked 5–10	303	1097

model DQ-SIM) and human ratings using Fisher's exact test. The *plausible* column contains the count of distractors that were rated 'good' and the *less plausible* column the count of all distractors that were rated otherwise (i.e., 'poor', 'true answer' and 'nonsense' distractors). The rows indicate the count of top-5 ranked distractors and the 5–10 ranked distractors.

Table 2.11: Contingency table for comparing human & system generated distractors

	Plausible	**Less plausible**
Human-generated	511	156
System-generated	255	412

Table 2.11 shows the contingency table for testing hypothesis 3 that compares the quality of human-generated with system-generated distractors. We use Fisher's exact test to test the hypothesis. The table shows counts of ratings in each category. For the human-generated row, we keep track of how each ground-truth distractor was rated, and update the counts depending on whether the distractors were rated 'good' (i.e., *plausible*) or the other labels (i.e., *less plausible*). Similarly, for the system-generated row, we count the ratings of top-*k* proposed distractors and update the counts in each column accordingly, where k is determined by the number of ground-truth distractors for that specific question.

Table 2.12: Conditional probabilities between raters (average of both directions)

Sub.	$gd \mid tr$	$gd \mid pf$	$gd \mid pm$	$gd \mid ns$	$pf \mid ns$	$pm \mid ns$
Eng.	35%	19%	44%	6%	11%	14%
His.	50%	22%	34%	5%	12%	6%

Table 2.12 illustrates the confusion observed between teachers in choosing which label to assign to a distractor. We show the confusion using conditional probabilities computed over both directions of the raters, where gd, tr, pf, pm, and ns denote good, true answer, poor format, poor meaning and nonsense distractors respectively. For example, the first column (i.e., $P(gd \mid tr)$) shows the probability of rating a distractor 'good' given that it was rated 'true answer' by the other rater.

References

[1] J. Dunlosky, K. A. Rawson, E. J. Marsh, M. J. Nathan, and D. T. Willingham. *Improving students' learning with effective learning techniques: Promising directions from cognitive and educational psychology*. Psychological Science in the Public interest, 14(1):4–58, 2013.

[2] M. J. Gierl, O. Bulut, Q. Guo, and X. Zhang. *Developing, analyzing, and using distractors for multiple-choice tests in education: A comprehensive review*. Review of Educational Research, 87(6):1082–1116, 2017.

[3] B. G. Davis. *Tools for teaching*. John Wiley & Sons, 2009.

[4] W. Ma, O. O. Adesope, J. C. Nesbit, and Q. Liu. *Intelligent tutoring systems and learning outcomes: A meta-analysis.* Journal of educational psychology, 106(4):901, 2014.

[5] M. Liu, V. Rus, and L. Liu. *Automatic chinese multiple choice question generation using mixed similarity strategy*. IEEE Transactions on Learning Technologies, 11(2):193–202, 2017.

[6] R. Mitkov, A. Varga, L. Rello, et al. *Semantic similarity of distractors in multiple-choice tests: extrinsic evaluation*. In Proceedings of the workshop on geometrical models of natural language semantics, pages 49–56, 2009.

[7] J. Pino, M. Heilman, and M. Eskenazi. *A selection strategy to improve cloze question quality*. In Proceedings of the Workshop on Intelligent

Tutoring Systems for Ill-Defined Domains. 9th International Conference on Intelligent Tutoring Systems, Montreal, Canada, pages 22–32. Citeseer, 2008.

[8] A. Papasalouros, K. Kanaris, and K. Kotis. *Automatic Generation Of Multiple Choice Questions From Domain Ontologies.* e-Learning, 1:427–434, 2008.

[9] A. Faizan and S. Lohmann. *Automatic generation of multiple choice questions from slide content using linked data*. In Proceedings of the 8th International Conference on Web Intelligence, Mining and Semantics, pages 1–8, 2018.

[10] J. Leo, G. Kurdi, N. Matentzoglu, B. Parsia, U. Sattler, S. Forge, G. Donato, and W. Dowling. *Ontology-based generation of medical, multi-term MCQs*. International Journal of Artificial Intelligence in Education, 29(2):145–188, 2019.

[11] T. Alsubait, B. Parsia, and U. Sattler. *Generating Multiple Questions From Ontologies: How Far Can We Go?* In Proceedings from the First International Workshop on Educational Knowledge Management (EKM 2014), Linköping, November 24, 2014, number 104, pages 19–30. Linköping University Electronic Press, 2014.

[12] D. Coniam. *A preliminary inquiry into using corpus word frequency data in the automatic generation of English language cloze tests*. Calico Journal, pages 15–33, 1997.

[13] T. Goto, T. Kojiri, T. Watanabe, T. Iwata, and T. Yamada. *Automatic generation system of multiple-choice cloze questions and its evaluation*. Knowledge Management & E-Learning: An International Journal, 2(3):210–224, 2010.

[14] J. Hill and R. Simha. *Automatic generation of context-based fill-in-the-blank exercises using co-occurrence likelihoods and Google n-grams*. In Proceedings of the 11th Workshop on Innovative Use of NLP for Building Educational Applications, pages 23–30, 2016.

[15] G. Kumar, R. E. Banchs, and L. F. D'Haro. *Automatic fill-the-blank question generator for student self-assessment*. In 2015 IEEE Frontiers in Education Conference (FIE), pages 1–3. IEEE, 2015.

[16] Q. Guo, C. Kulkarni, A. Kittur, J. P. Bigham, and E. Brunskill. *Questimator: Generating knowledge assessments for arbitrary topics*. In IJCAI-16: Proceedings of the AAAI Twenty-Fifth International Joint Conference on Artificial Intelligence, 2016.

[17] S. Jiang and J. S. Lee. *Distractor generation for chinese fill-in-the-blank items*. In Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications, pages 143–148, 2017.

[18] C. Liang, X. Yang, D. Wham, B. Pursel, R. Passonneaur, and C. L. Giles. *Distractor generation with generative adversarial nets for automatically creating fill-in-the-blank questions*. In Proceedings of the Knowledge Capture Conference, pages 1–4, 2017.

[19] C. Liang, X. Yang, N. Dave, D. Wham, B. Pursel, and C. L. Giles. *Distractor generation for multiple choice questions using learning to rank*. In Proceedings of the thirteenth workshop on innovative use of NLP for building educational applications, pages 284–290, 2018.

[20] M. Liu, V. Rus, and L. Liu. *Automatic chinese factual question generation*. IEEE Transactions on Learning Technologies, 10(2):194–204, 2016.

[21] D. R. Ch and S. K. Saha. *Automatic multiple choice question generation from text: A survey*. IEEE Transactions on Learning Technologies, 13(1):14–25, 2018.

[22] A. C. Butler. *Multiple-choice testing in education: Are the best practices for assessment also good for learning?* Journal of Applied Research in Memory and Cognition, 7(3):323–331, 2018.

[23] K. Woodford and P. Bancroft. *Using multiple choice questions effectively in information technology education*. In Ascilite, volume 4, pages 948–955, 2004.

[24] A.-M. Brady. *Assessment of learning with multiple-choice questions*. Nurse Education in Practice, 5(4):238–242, 2005.

[25] J. Collins. *Education techniques for lifelong learning: writing multiple-choice questions for continuing medical education activities and self-assessment modules.* Radiographics: a review publication of the Radiological Society of North America, Inc, 26(2):543–551, 2006.

[26] R. Blackey. *So Many Choices, so Little Time: Strategies for Understanding and Taking Multiple-Choice Exams in History*. The History Teacher, 43(1):53–66, 2009. Available from: http://www.jstor.org/stable/40543353.

[27] H. M. Abdulghani, F. Ahmad, M. Irshad, M. S. Khalil, G. K. Al-Shaikh, S. Syed, A. A. Aldrees, N. Alrowais, and S. Haque. *Faculty development programs improve the quality of Multiple Choice Questions items' writing*. Scientific reports, 5(1):1–7, 2015.

[28] N. Naeem, C. van der Vleuten, and E. A. Alfaris. *Faculty development on item writing substantially improves item quality*. Advances in health sciences education, 17(3):369–376, 2012.

[29] T. M. Haladyna and S. M. Downing. *A taxonomy of multiple-choice item-writing rules*. Applied measurement in education, 2(1):37–50, 1989.

[30] T. M. Haladyna and M. C. Rodriguez. *Developing and validating test items*. Routledge, 2013.

[31] R. Moreno, R. J. Martínez, and J. Muñiz. *Guidelines based on validity criteria for the development of multiple choice items*. Psicothema, 27(4):388–394, 2015.

[32] R. Vyas and A. Supe. *Multiple choice questions: a literature review on the optimal number of options*. Natl Med J India, 21(3):130–3, 2008.

[33] R. Mitkov, H. Le An, and N. Karamanis. *A computer-aided environment for generating multiple-choice test items*. Natural language engineering, 12(2):177–194, 2006.

[34] R. Mitkov et al. *Computer-aided generation of multiple-choice tests*. In Proceedings of the HLT-NAACL 03 workshop on Building educational applications using natural language processing, pages 17–22, 2003.

[35] Y.-C. Lin, L.-C. Sung, and M. C. Chen. *An automatic multiple-choice question generation scheme for english adjective understanding*. In Workshop on Modeling, Management and Generation of Problems/Questions in eLearning, the 15th International Conference on Computers in Education (ICCE 2007), pages 137–142, 2007.

[36] G. A. Miller, R. Beckwith, C. Fellbaum, D. Gross, and K. J. Miller. *Introduction to WordNet: An on-line lexical database*. International journal of lexicography, 3(4):235–244, 1990.

[37] M. A. Lopetegui, B. A. Lara, P.-Y. Yen, Ü. V. Çatalyürek, and P. R. Payne. *A novel multiple choice question generation strategy: alternative uses for controlled vocabulary thesauri in biomedical-sciences education*. In AMIA Annual Symposium Proceedings, volume 2015, page 861. American Medical Informatics Association, 2015.

[38] D. Seyler, M. Yahya, and K. Berberich. *Knowledge questions from knowledge graphs*. In Proceedings of the ACM SIGIR International Conference on Theory of Information Retrieval, pages 11–18, 2017.

[39] K. Stasaski and M. A. Hearst. *Multiple choice question generation utilizing an ontology*. In Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications, pages 303–312, 2017.

[40] G. Lai, Q. Xie, H. Liu, Y. Yang, and E. Hovy. *RACE: Large-scale ReAding Comprehension Dataset From Examinations*. In M. Palmer, R. Hwa, and S. Riedel, editors, Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pages 785–794, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. Available from: https://aclanthology.org/D17-1082, doi:10.18653/v1/D17-1082.

[41] Y. Gao, L. Bing, P. Li, I. King, and M. R. Lyu. *Generating distractors for reading comprehension questions from real examinations*. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 33, pages 6423–6430, 2019.

[42] H. Zhu, F. Wei, B. Qin, and T. Liu. *Hierarchical attention flow for multiple-choice reading comprehension*. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 32, 2018.

[43] H.-L. Chung, Y.-H. Chan, and Y.-C. Fan. *A BERT-based Distractor Generation Scheme with Multi-tasking and Negative Answer Training Strategies.* In T. Cohn, Y. He, and Y. Liu, editors, Findings of the Association for Computational Linguistics: EMNLP 2020, pages 4390–4400, Online, November 2020. Association for Computational Linguistics. Available from: https://aclanthology.org/2020.findings-emnlp.393, doi:10.18653/v1/2020.findings-emnlp.393.

[44] B. Kulis et al. *Metric learning: A survey*. Foundations and Trends® in Machine Learning, 5(4):287–364, 2013.

[45] J. Welbl, N. F. Liu, and M. Gardner. *Crowdsourcing Multiple Choice Science Questions*. In L. Derczynski, W. Xu, A. Ritter, and T. Baldwin, editors, Proceedings of the 3rd Workshop on Noisy User-generated Text, pages 94–106, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. Available from: https://aclanthology.org/W17-4413, doi:10.18653/v1/W17-4413.

[46] T. Mikolov, K. Chen, G. S. Corrado, and J. Dean. *Efficient Estimation of Word Representations in Vector Space*. In International Conference on Learning Representations, 2013. Available from: https://api.semanticscholar.org/CorpusID:5959482.

[47] M. Kusner, Y. Sun, N. Kolkin, and K. Weinberger. *From word embeddings to document distances*. In International conference on machine learning, pages 957–966. PMLR, 2015.

[48] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. *Attention is all you need*. Advances in neural information processing systems, 30, 2017.

[49] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever, et al. *Improving language understanding by generative pre-training*. 2018.

[50] S. Edunov, M. Ott, M. Auli, and D. Grangier. *Understanding Back-Translation at Scale*. In E. Riloff, D. Chiang, J. Hockenmaier, and J. Tsujii, editors, Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 489–500, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. Available from: https://aclanthology.org/D18-1045, doi:10.18653/v1/D18-1045.

[51] M. Zhong, P. Liu, Y. Chen, D. Wang, X. Qiu, and X. Huang. *Extractive Summarization as Text Matching*. In D. Jurafsky, J. Chai, N. Schluter, and J. Tetreault, editors, Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 6197–6208, Online, July 2020. Association for Computational Linguistics. Available from: https://aclanthology.org/2020.acl-main.552, doi:10.18653/v1/2020.acl-main.552.

[52] S. J. Pan and Q. Yang. *A survey on transfer learning*. IEEE Transactions on knowledge and data engineering, 22(10):1345–1359, 2009.

[53] I. Goodfellow, Y. Bengio, and A. Courville. *Deep learning*. MIT press, 2016.

[54] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. *Bert: Pre-training of deep bidirectional transformers for language understanding*. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), 2019.

[55] M. Guo, Y. Yang, D. Cer, Q. Shen, and N. Constant. *MultiReQA: A Cross-Domain Evaluation for Retrieval Question Answering Models*. arXiv preprint arXiv:2005.02507, 2020.

[56] H. Khosravi, S. B. Shum, G. Chen, C. Conati, Y.-S. Tsai, J. Kay, S. Knight, R. Martinez-Maldonado, S. Sadiq, and D. Gašević. *Explainable artificial intelligence in education*. Computers and Education: Artificial Intelligence, 3:100074, 2022.

[57] K. Sun, T. Yao, S. Chen, S. Ding, J. Li, and R. Ji. *Dual contrastive learning for general face forgery detection*. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 36, pages 2316–2324, 2022.

[58] S. Chopra, R. Hadsell, and Y. LeCun. *Learning a similarity metric discriminatively, with application to face verification*. In 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), volume 1, pages 539–546. IEEE, 2005.

[59] N. A. Smith and J. Eisner. *Contrastive estimation: Training log-linear models on unlabeled data*. In Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05), pages 354–362, 2005.

[60] V. Karpukhin, B. Oguz, S. Min, P. Lewis, L. Wu, S. Edunov, D. Chen, and W.-t. Yih. *Dense Passage Retrieval for Open-Domain Question Answering*. In B. Webber, T. Cohn, Y. He, and Y. Liu, editors, Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 6769–6781, Online, November 2020. Association for Computational Linguistics. Available from: https://aclanthology.org/2020.emnlp-main.550, doi:10.18653/v1/2020.emnlp-main.550.

[61] K. Sohn. *Improved deep metric learning with multi-class n-pair loss objective*. Advances in neural information processing systems, 29, 2016.

[62] A. Singh Bhatia, M. Kirti, and S. K. Saha. *Automatic generation of multiple choice questions using wikipedia*. In International conference on pattern recognition and machine intelligence, pages 733–738. Springer, 2013.

[63] J. Araki, D. Rajagopal, S. Sankaranarayanan, S. Holm, Y. Yamakawa, and T. Mitamura. *Generating questions and multiple-choice answers using semantic analysis of texts*. In Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, pages 1125–1136, 2016.

[64] M. L. McHugh. *Interrater reliability: the kappa statistic*. Biochemia medica, 22(3):276–282, 2012.

[65] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al.

Scikit-learn: Machine learning in Python. Journal of machine learning research, 12(Oct):2825–2830, 2011.

[66] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. *PyTorch: An Imperative Style, High-Performance Deep Learning Library*. In Advances in Neural Information Processing Systems 32, pages 8024–8035. Curran Associates, Inc., 2019. Available from: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[67] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, et al. *Huggingface's transformers: State-of-the-art natural language processing*. arXiv preprint arXiv:1910.03771, 2019.

[68] D. P. Kingma and J. Ba. *Adam: A method for stochastic optimization*. arXiv preprint arXiv:1412.6980, 2014.

3

Leveraging Large Language Models for Distractor Generation

In this chapter, we study how large language models could be leveraged for the task of distractor generation in educational multiple-choice questions. We propose a novel strategy for guiding LLMs in generating plausible distractors by prompting them with question items automatically retrieved using local models introduced in Chapter 2 from question banks. These retrieved question items serve as well-chosen in-context examples. We combine the original question (i.e., to generate distractor for) with these examples into a prompt to the LLM. We show that our strategy leads to performance gains in terms of generating high quality distractors.

⋆⋆⋆

Distractor generation for multiple-choice questions with predictive prompting and large language models

S.K. Bitew, J. Deleu, C. Develder and T. Demeester

In Proceedings of the First Workshop on Responsible Knowledge Discovery in Education (RKDE) at ECML-PKDD 2023.

Abstract Large Language Models (LLMs) such as ChatGPT have demonstrated remarkable performance across various tasks and have garnered

significant attention from both researchers and practitioners. However, in an educational context, we still observe a performance gap in generating distractors — i.e., plausible yet incorrect answers — with LLMs for multiple-choice questions (MCQs). In this study, we propose a strategy for guiding LLMs such as ChatGPT, in generating relevant distractors by prompting them with question items automatically retrieved from a question bank as well-chosen in-context examples. We evaluate our LLM-based solutions using a quantitative assessment on an existing test set, as well as through quality annotations by human experts, i.e., teachers. We found that on average 53% of the generated distractors presented to the teachers were rated as high-quality, i.e., suitable for immediate use as is, outperforming the state-of-the-art model. We also show the gains of our approach[1] in generating high-quality distractors by comparing it with a zero-shot ChatGPT and a few-shot ChatGPT prompted with static examples.

Keywords Distractor generation, natural language processing , large language models , predictive prompting , language learning , neural networks.

3.1 Introduction

The rapid advancement in artificial intelligence (AI) and large language models (LLMs) have paved the way for transformative applications across various domains, including the education domain. Since several LLMs (e.g., GPT-3 [1], InstructGPT [2], GPT-4 [3]) have been pretrained on massive amounts of data across multiple domains and languages, they are capable of solving natural language processing (NLP) tasks with little training examples (i.e., few-shot learning) or no additional training (i.e., zero-shot learning). This opens up new opportunities for adopting LLMs in the development of many educational technological solutions that aim to automate time-consuming and laborious educational tasks such as generating questions [4] and exercises [5], essay scoring [6], and automated feedback [7].

In particular, the recent release of ChatGPT, an LLMs-based generative AI model that requires only natural language prompts without additional model training or fine-tuning, has demonstrated diverse potential in automating various educational tasks. For example, ChatGPT has achieved the equivalent of a passing score for a third-year medical student (above 60%) in the United States Medical Licence Examination (USMLE) Step 1 exam, and provided logical justification and informational context across the majority of answers [8]. Likewise, ChatGPT's performance on four real

[1] https://github.com/semerekiros/distractGPT/

exams (containing 95 MCQs and 12 essay writing questions), at the University of Minnesota Law School was equivalent to C+ students implying a pass in the course [9]. Li *et al.* [10] show the capability of ChatGPT in generating high-quality reflective responses in writing assignments administered for pharmacy courses.

One important educational task is the generation of multiple-choice questions (MCQs). MCQs have long been a popular form of formative and summative assessment in education due to their automatic scoring capability and the potential they hold for delivering timely and targeted feedback, which is crucial for facilitating effective learning [11]. However, the process of crafting high-quality MCQs with effective distractors (i.e., plausible yet incorrect answers) has traditionally been both a challenging and time-consuming task for educators (e.g., teachers, content creators etc.) as poorly prepared distractors undermine the quality of MCQs [12]. This is where LLMs offer substantial benefits as they can be leveraged to automate the MCQ construction process, thus saving educators' time and effort while maintaining the quality and validity of the assessment items. For instance, teachers could employ LLMs to not only create different variants of the same MCQ questions but also develop different MCQs of comparable difficulty levels, facilitating targeted assessment for students with similar proficiency levels. Furthermore, students can benefit from the availability of several MCQs, enabling them to engage in regular practice, which is a well-established and highly effective learning strategy [13]. Additionally, such models could be used for large-scale testing contexts (e.g., licensure and certification testing) in which it is necessary to have multiple forms of a test and to introduce new question items regularly to minimize security concerns related to item exposure.

In a recent study [14] conducted around the same time as the release of ChatGPT, researchers used local language models to automatically retrieve and reuse distractors to create new MCQs for education by leveraging existing pools of question items. In a user study they conducted with teachers, 3 out of 10 distractors proposed by their system were found to be high-quality, which is generally sufficient for creating an MCQ, as an average MCQ typically contains 3 distractors. However, they also report a staggering 50% production of distractors that were entirely out of context given a question (so-called "nonsense distractors"). With the emergence of ChatGPT, the question arises: *does this previous approach become obsolete?* In our current study, we aim to address this question by examining the ability of out-of-the-box ChatGPT to generate effective distractors to be measured on the same scale as the previous study and evaluated by experts. Moreover, we study how both approaches could be combined into an even more effective

approach. We also delve into the reliability issue, specifically in decreasing the production of nonsense distractors, which has implications for teachers' trust in the distractor generation tools. To guide our investigation, we formulate the following research questions (RQs):

1. **RQ1**: In comparison to ranking-based models, does ChatGPT generate high-quality distractors for educational MCQs?
2. **RQ2:** To what extent can we rely on ChatGPT-generated distractors, and how can we measure their trustworthiness?
3. **RQ3:** Is it possible to enhance the capability of distractor generation by combining ranking-based models with LLMs?

To answer the RQs, we designed ChatGPT prompting strategies and we solicited feedback from human experts, i.e., teachers, to evaluate the quality of generated distractors. We also compared the different strategies in terms of the reliability of generating less nonsensical distractors. In general, we found ChatGPT-driven solutions produced high-quality distractors compared to ranking-based models. They are also more reliable than the ranking-based model as they produce significantly less number of nonsense distractors. We also combined the rank-based approach with ChatGPT, through the automatic composition of an example-based prompt from the output of the rank-based model. We found that this leads to a more reliable and effective generation of distractors. The contribution of this paper can be summarized as follows:

- We proposed a strategy to guide LLMs, specifically ChatGPT, to generate effective distractors for MCQs across various subjects by prompting the model with question items automatically retrieved from existing question banks.
- We performed a user study with teachers to evaluate the quality of distractors proposed by our strategy.
- The evaluation of our approach unveils its dual capability to generate valuable distractors while simultaneously minimizing the occurrence of nonsensical options.

The remainder of the paper is organized as follows: Section 3.2 describes the relevant work in distractor generation and LLM prompting strategies. Section 3.3 explains the details of the baselines and the proposed method, while Section 3.4 introduces the test dataset and the evaluation setup of the user study with teachers. In Section 3.5.2, we report the results and provide some insights. Finally, in Section 3.6, we present the conclusion by summarizing the key findings and implications of our study.

3.2 Related Work

Since we briefly covered the broad application of LLMs in education in the introduction, in this section we only focus on describing prior works on distractor generation (Section 3.2.1) and discussing LLMs' prompting strategies (Section 3.2.2) and their relevance to our work.

3.2.1 Distractor Generation

We focus on generating incorrect options (i.e., distractors) for multiple-choice questions (MCQs), which is a time-consuming task that impacts MCQ quality and has been extensively researched. Broadly speaking, the main methods for generating distractors can be categorized into retrieval-based and generation-based techniques.

Retrieval-based methods generate distractors by selecting the most similar alternative answers in existing knowledge bases or question item corpora. To approximate the similarity between distractors and the answer key (and question stem), several approaches are used based on (i) embedding space proximity [14–16], (ii) similarity in lexical databases such as WordNet [17], which is of particular importance in language and vocabulary learning [18, 19], and (iii) the semantic distance within domain-specific ontologies, which is critical in factoid-type questions [20–23]. This ultimately leads to the selection of candidate distractors based on a ranking strategy [24].

Generation-based methods make use of deep learning models to directly generate distractors. Pioneering research [25–27] demonstrated the feasibility of using sequence-to-sequence models to generate distractors, while more recently, solutions based on BERT [28, 29] or T5 [30] have been explored. Rather than directly (auto-regressively) generate a distractor, the technique of back translation has shown to be relatively effective (beating a BERT-based baseline) for fill-in-the-blank language assessment tests [31].

In this work, we investigate the potential of ChatGPT[2], a large and autoregressive language model, in creating distractors. We aim to combine retrieval-based and generative-based approaches by (i) automatically retrieving similar question items from pre-existing question banks to compose an example prompt and (ii) using this example prompt to guide ChatGPT to generate relevant distractors.

3.2.2 Prompting strategies

Recent instruction-based large language models (LLMs) have been a game-changer for various tasks, showing remarkable performance without any

[2] https://chat.openai.com/

task-specific training (e.g., through finetuning) of the LLM [1, 32]. A specific task is solved through phrasing an instruction (zero-shot), possibly including a few input/output examples (few-shot) for the task at hand, as the so-called prompt that serves as input to the LLM. The few-shot setting, including some examples, is commonly referred to as in-context learning (ICL). Another prompting strategy, chain-of-thought, induces language models to generate intermediate steps before predicting the final response [33].

In this paper, we introduce a variant of ICL wherein the examples presented to the LLM are determined dynamically, based on the test example (i.e., the question to generate distractors for, in our case).

3.3 Methods

We now describe our finetuned T5-based model (Section 3.3.1), and out-of-the-box ChatGPT-based solutions in a zero-shot setting (Section 3.3.2), as well as using in-context learning (Section 3.3.3).

3.3.1 T5-based Distractor Generation

We fine-tuned a multilingual T5 (mT5) model [34] to generate distractors. To this end, we use a private dataset (i.e., the Televic dataset from [14]) of 62K multiple-choice question items in the form of triplets comprising a question, answer and distractors. These question items are diverse in terms of language, domain, subject and question type. On average, a question item has more than 2 distractors and contains exactly one answer. Additionally, the distractors in the dataset are not limited to single-word distractors.

Following the unsupervised pre-training objective used in the mT5 model, we rearranged our fine-tuning data into input and output sequences as illustrated in Fig. 3.1. Our mT5 model's input sequence is constructed by copying the question stem and answer from the original question item and inserting the sentence *"Which of the following are incorrect answers"* (or its translation depending on the language of the question item) between them. Furthermore, we masked each distractor (i.e., distractors could be multi-word spans) in the question item using a sentinel token[3] and separated them by increasing item numbers. The target sequence corresponds to all the dropped-out distractors and the objective is to predict the distractors.

The fine-tuning configuration that we have devised is intended to simplify the generation of multiple distractors. Specifically, all the necessary

[3]Each sentinel token is assigned a token ID that is unique to the sequence. The sentinel IDs are special tokens added to the model's vocabulary and do not correspond to any wordpiece

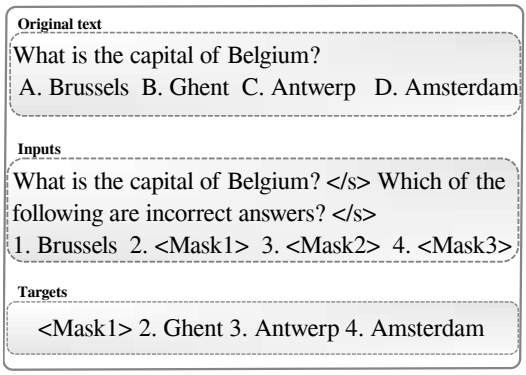


Figure 3.1: Schematic of our fine-tuning procedure. The input sequence is constructed by copying the question and the answer from the original text and adding the template sentence *"Which of the following are incorrect answers"*. Each distractor is masked with a unique sentinel token (shown as ⟨Mask$x$⟩). The output sequence then consists of the dropped-out distractors. Note that a single sentinel token replaces all consecutive spans of dropped-out tokens, and the template sentence is translated into the language of the question item (i.e., Dutch or French).

distractors for each question are generated as a list separated by numbers in a single decoding step.

3.3.2 Zero-shot ChatGPT

To use ChatGPT in a zero-shot setting (Zero-ChatGPT), we construct a prompt that concatenates a fixed instruction sentence and the test example, as shown in Fig. 3.2. Note that each time a new query is made to ChatGPT, we clear conversations to avoid the influence of previous samples through independent API calls. We use a Python ChatGPT wrapper[4] to call the ChatGPT API automatically.

3.3.3 Demonstration-based ChatGPT

Finally, we evaluate ChatGPT in a few-shot setting by probing it with smartly chosen demonstrations (Dynamic-Demo-ChatGPT). We propose to retrieve the most relevant question items from the Televic dataset (see Section 3.3.1) and use them as demonstrations for a given test instance. We accomplish this by leveraging the question similarity (Q-SIM) model proposed by [14] to automatically select the top similar question items for

[4]Note that all the calls to the API were made between 06/04/2023 and 11/04/2023. Link to wrapper: https://github.com/mmabrouk/chatgpt-wrapper

Original Question and Correct Answer:

What is the capital of Belgium?
Brussels

Zero-shot prompt input:

Generate 10 plausible but incorrect answers for the following question.
question: What is the capital of Belgium?
answer: Brussels

Figure 3.2: Example of a question with its correct answer and how we turn that into a zero-shot prompt. Note that we translate the fixed template parts for questions in languages other than English.

the given test instance. The Q-SIM model is a BERT-based ranking model that returns a ranked list of question items according to their similarity to a given test question. Figure 3.3 illustrates how we combine the original question (to generate distractors for) with the retrieved examples into a prompt to ChatGPT.

3.4 Experiments

3.4.1 Test Dataset

To quantitatively evaluate our distractor generating models introduced in Section 3.3, we use the Wezooz test data introduced by [14], which comprises 300 multiple-choice questions (MCQs) designed for language and factual knowledge learning and is aimed at secondary school students and teachers. It includes French and English questions for language learning purposes, while Natural sciences, Geography, History and Biology constitute the factoid questions. Each subject has 50 MCQs. Note that the data distribution of the factoid questions is different from the Televic dataset (see Section 3.3.1 for details), which we use to (i) fine-tune our mT5 model, and (ii) retrieve similar examples in our demonstration-based ChatGPT model. However, the language learning questions are drawn from the same distribution, in a similar design setup as [14].

Original Question and Correct Answer:

What is the capital of Belgium?
Brussels

Few-shot prompt input:

Generate 10 plausible but incorrect answers for the following question.

question: What is the capital of Germany?
answer: Berlin
incorrect answers: 1. Frankfurt 2. Paris 3. Hamburg 4. Madrid
⋮
question: What is the capital of France?
answer: Paris
incorrect answers: 1. Brussels 2. Marseille 3. Rome 4. Nice

question: What is the capital of Belgium?
answer: Brussels
incorrect answers:

Figure 3.3: Schematic of our demonstration-based prompt construction. The top-k example demonstrations are automatically retrieved from the Televic question pool, and concatenated with the instruction and test instance. This prompt is used as a query to ChatGPT for generating distractors. Note that the fixed template parts are translated into the language of the test question item (i.e., Dutch or French).

3.4.2 Human Expert Quality Assessment

We also investigated our models' output quality using human assessors, by collecting feedback from teachers. For each of the 300 questions in the aforementioned WeZooz test set, we generated 10 distractors with each of our 3 models. The teachers were then presented with a randomized list of all 30 generated distractors for each question. They were explicitly instructed to rate each distractor independent of the other distractors in the list, based on how much they thought it would help them if they were given the task of preparing distractors for that specific question. We used the four-level annotation scheme proposed by [14] to assign quality labels to each distractor: (1) **True Answer**: the distractor partially or completely overlaps with the answer key. (2) **Good distractor**: the distractor is viable and could be used in an MCQ as is. (3) **Poor distractor**: the distractor is on

Table 3.1: Inter-annotation agreement of experts, measured by the Jaccard similarity coefficient.

Subjects	True	Good	Poor	Nonsense	Overall
English	50.0	37.6	8.9	40.0	49.4
Geography	33.3	75.7	34.1	35.0	74.0

topic but could easily be ruled out by students. (4) **Nonsense distractor**: distractor is completely out of context.

3.5 Results and Discussion

In this section, we provide evidence of the effectiveness and reliability of our approach by reporting the experimental results and discussing the insights obtained. In Section 3.5.1, we explain the annotation agreement among the teachers, followed by the evaluation results in Section 3.5.2.

3.5.1 Inter-annotator agreement

Following the annotation scheme introduced in Section 3.4.2, a total of 12,860 ratings for distractor quality were collected from the annotation by teachers (see Table 3.4 in Section 3.7 for details of rating statistics). These ratings come from 10 distractors generated by each of the models (i.e., all presented simultaneously to teachers as randomly shuffled list). In total, 10 teachers participated in our quality assessment study.

We adopt two strategies to determine the level of agreement between annotators. First, we ask teachers to rate the same set of distractors using the four-level annotation scale. We selected the subjects English, from language category, and Geography, from factoids, for annotations by at least two teachers. Table 3.1 shows the inter-annotator agreement of teachers using the Jaccard similarity coefficient. The Jaccard similarity measures the similarity between two sets of data by calculating what fraction of the union of those datasets is covered by their intersection. In our case, it is calculated as the number of times the teachers agreed on a distractor quality label (i.e., one of the four labels), divided by the total number of distractors that were annotated (by either annotator) with that label. In general, we note a higher agreement on what is considered a good distractor compared to the other distractor categories. Moreover, the overall agreement between the Geography teachers is higher than the English teachers.

Table 3.2: Expert evaluation of distractors (%). GDR: good distractor rate, NDR: nonsense distractor rate; ↑: higher is better, ↓: lower is better; evaluation on WeZooz test set. The markers ⋆ and ‡ respectively denote the one-tailed significance levels of the bootstrap-based $p-$value, i.e., $p < 0.1$ and $p < 0.01$ with respect to the best model Dynamic-Demo-ChatGPT in each column.

Models	**Language learning**		**Factoid learning**	
	GDR@10 ↑	**NDR@10** ↓	**GDR@10** ↑	**NDR@10** ↓
DQ-SIM [14]	27.9‡	44.6‡	28.9‡	50.1‡
mT5	24.5‡	42.3‡	27.8‡	36.6‡
Zero-ChatGPT	30.2‡	34.6‡	57.6⋆	17.5⋆
Dynamic-Demo-ChatGPT	**46.7**	**15.5**	**58.8**	**16.4**

Second, we employed the widely utilized Cohen's kappa coefficient [35]. Our analysis substantiates the previously mentioned observation that annotators have a greater consensus when evaluating factoid questions compared to language-related queries as [14]. Specifically, among English teachers, the calculated Cohen's kappa value stands at 28.9, signifying a "fair agreement" level. Similarly, Geography teachers exhibit a higher level of agreement with a Cohen's kappa value of 52, indicating a level of agreement categorized as "moderate."

3.5.2 Evaluation of models

Table 3.2 shows the expert evaluation of distractors in terms of *good distractor rate* (GDR@10), and *nonsense distractor rate* (NDR@10). GDR@10 is calculated as the percentage of distractors that were rated 'good' among the proposed 10 distractor for each model. Similarly, NDR@10 is calculated as the percentage of distractors that were rated 'nonsense' among the 10 candidate distractors proposed by each model. We are interested in reporting the NDR metric because it could be used as a measure of the reliability of educational models, as a high occurrence of nonsense distractors may undermine users' trust in the model. The reported metrics are averages of all the subjects in each category (i.e., French and English for language learning, and Biology, Natural Sciences, History and Geography for factoids). In the table, the upward arrow (↑) indicates larger values are desired, while the downward arrow↓ indicates smaller values are preferred.

In general, the ChatGPT-based solutions (i.e., Zero-ChatGPT, and Dynamic-Demo-ChatGPT) were rated better in proposing plausible distractors than the baselines. They also produced fewer nonsense distractors. Particularly, the Dynamic-Demo-ChatGPT outperformed all the other models. On average,

approximately 5 of its 10 proposed distractors were rated high-quality distractors and only 1.5 distractors were rated nonsense. Moreover, on average 8.5 distractors were generally found to be on-topic (i.e., distractors rated as either good or poor distractors) for our best model Dynamic-Demo-ChatGPT.

All the models are better at generating effective distractors for factoids than for language questions as shown by the higher GDR@10 results for factoids than languages. We hypothesize this is because, for factoid questions, our models are mainly tasked with generating accurately composed distractors that are contextually incorrect. In contrast, when faced with language questions, the intended distractors may possess ungrammatical attributes, posing a challenge for our models to generate text that is intentionally ungrammatical.

Our purely generative local mT5 model does not improve the DQ-SIM model (i.e., previous state-of-the-art model on the test set) at proposing good distractors (i.e., GDR@10 of 24.5 vs. 27.9 and 28.9 vs. 27.8). However, it is a more reliable model as it produces fewer nonsense distractors as illustrated by its lower NDR@10 values of 42.3 and 36.6 for languages and factoids, respectively, in contrast to the corresponding values of 44.6 and 50.1 for the DQ-SIM model. The relatively high number of nonsense distractors in DQ-SIM is partly attributed to its inherent limitation of only ranking pre-existing distractors according to their relevance to a given question, thereby lacking the ability to generate brand-new distractors.

In addition, in order to ensure the validity of the differences between the models, we carry out a bootstrap significance analysis [36] by sampling with replacement the annotation results DQ-SIM, mT5, Zero-ChatGPT, and Dynamic-Demo-ChatGPT models 1000 times. The resulting one-tailed significance levels (p values) are indicated in Table 3.2 by markers $\star$ and ‡ which respectively denote $p < 0.1$ and $p < 0.01$ with respect to our best model Dynamic-Demo-ChatGPT in each column.

Effect of dynamically retrieved in-context examples We replace the dynamically retrieved examples with randomly selected language in-context examples from the Televic question bank, and we keep this selection constant (i.e., Static-Demo-ChatGPT) to generate distractors. Similar to the other models, we generated 10 distractors using the Static-Demo-ChatGPT model and asked teachers to annotate the quality of the distractors. We focused on the language learning category as it showed a huge performance improvement when transitioning from Zero-ChatGPT to Dynamic-Demo-ChatGPT.

We observe that the Dynamic-Demo-ChatGPT model significantly outperforms the Static-Demo-ChatGPT model in generating high-quality distractors as indicated by the GDR@10 metric in Table 3.3. However, the

Table 3.3: Effect of using dynamically retrieved in-context examples: Dynamic-Demo-ChatGPT vs. Static-Demo-ChatGPT that uses static in-context examples for language learning. The markers ‡ denotes the one-tailed significance level of the bootstrap-based $p-$value, i.e., $p < 0.01$ with respect to Dynamic-Demo-ChatGPT

Models	**GDR@10** ↑	**NDR@10** ↓
Static-Demo-ChatGPT	43.3‡	16.2
Dynamic-Demo-ChatGPT	**46.7**	15.5

difference in generating less nonsense distractor (i.e., NDR@10) is not significant. See Table 3.5 in Section 3.7 for an example of generated distractors using the approaches.

3.5.3 Discussion of Research questions

To answer **RQ1**, we compare the ChatGPT-based solutions (i.e., Zero-ChatGPT, Static-Demo-ChatGPT and Dynamic-Demo-ChatGPT) with the previous state-of-the-art ranking-based model, DQ-SIM in generating distractors. All the ChatGPT-based distractor generation strategies significantly outperform the DQ-SIM.

To address **RQ2**, we employ the NDR@10 metric as a proxy to measure the trustworthiness of models. Our best model produces an average of only 16% nonsense distractors, which is a remarkable improvement compared to the previously reported state-of-the-art performance of 50% NDR@10. This significant reduction of nonsense distractors can be expected to inspire more trust in the approach by teachers.

To answer **RQ3**, we compare Dynamic-Demo-ChatGPT, which combines a local ranking model with ChatGPT, against Zero-ChatGPT and Static-Demo-ChatGPT. As shown in Table 3.2 and Table 3.3, combining local models with ChatGPT leads to a better quality distractor generation, highlighting the effectiveness of this combined approach.

3.6 Conclusion

This research paper introduced and evaluated a novel strategy designed to guide LLMs, such as ChatGPT, in generating reliable and effective distractors for the creation of MCQs in educational contexts. Our proposed approach, Dynamic-Demo-ChatGPT model combines a rank-based approach with ChatGPT. This involves the dynamic retrieval of relevant question

items through the ranker that are then presented as in-context examples to ChatGPT for generating distractors. Our results indicated that the Dynamic-Demo-ChatGPT showed a considerably reduced production of nonsense distractors (i.e., only 16% rated as nonsense) compared to Zero-ChatGPT (i.e., out-of-the-box ChatGPT), which we consider a useful asset in terms of trust in the model by teachers. Moreover, on average, 5 out of the 10 distractors suggested by our approach were rated as high-quality by teachers, to be readily used.

For future work, we aim to investigate designing a fine-grained evaluation setup for distractors that takes into account various factors such as the level of the student, the difficulty of the questions etc. There is also a potential to explore alternative prompting strategies for LLMS, when generating distractors. For example, the utilization of self-correcting mechanism [37], which involves revising the initial output of an LLM by evaluating certain aspects of the text, could be explored in the context of distractor generation.

3.7 Appendix

3.A User Study Details

This section contains the user study details. Table 3.4 describes the data gathered from the annotations provided by teachers. Every subject contains 50 questions, except English which has 48 questions. We collected 12,860 annotations for the proposed candidate distractors (i.e., 10 distractors by each of the three models). A total of 10 teachers participated in the study. English (i.e., from languages) and Geography (i.e., from factoids) were annotated twice by two different teachers to calculate inter-annotator agreement. Additionally, to study the effect of dynamic retrieval of in-context examples, we asked 1 English and 1 French teacher to annotate the distractor predictions from the Static-Demo-ChatGPT model. The second column (i.e., *Item count*), shows the number of question items for each subject in the Wezooz dataset. Alongside, the *distractors count* column provides two distinct values: the gold truth distractors count within the dataset, and the count of unique distractors generated by our models. It is important to note that different models may produce identical distractors for a given question, resulting in varying numbers of newly generated distractors across the different subjects.

3.B Example Generated Distractors

Table 3.5 shows a cherry-picked example of our ChatGPT-based models.

Table 3.4: Ratings Data Description

Subjects	**Item count**	**Distractors count**		**Ratings count**	**No of Raters**
		Gold	**Generated**		
English	48	130	1324	3360	3
French	50	92	1396	2000	2
Geography	50	145	1248	3000	2
History	50	130	1354	1500	1
Biology	50	88	1391	1500	1
Nat. Sciences	50	100	1449	1500	1
Total	298	685	8162	12860	10

Table 3.5: Some generated examples from Zero-ChatGPT, Dynamic-Demo-ChatGPT, Static-Demo-ChatGPT models for English. High-quality distractors are shown in **boldface**, while on-topic and nonsense distractors are *italicized* and underlined, respectively. We only show 2 in-context examples for the Static-Demo-ChatGPTand Dynamic-Demo-ChatGPTmodels as part of the prompt but in practice, we use 5 of such examples.

Model	Example
Zero-ChatGPT	**Prompt**: Generate 10 plausible but incorrect answers for the following question. question: They do n't speak English and can not make ... understood. answer: themselves **Ground**: [them, us,you] **Prediction**: [1. *each other*, 2. any sense, 3. a sentence, 4. appropriate gestures, 5. complete thoughts, 6. proper communication, 7. accurate translations, 8. meaningful conversation, 9. clear statements, 10. intelligible sounds]
Static-Demo-ChatGPT	**Prompt**: Generate 10 plausible but incorrect answers for the following question. question: Why do n't we ask Jane to come over and talk ? She 's an ... answer: alumna incorrect answers: 1. alumnus 2. alumni 3. alumn question: Choose the opposite of the word in bold . This company is loss-making . answer: profit-making incorrect answers: 1. loose 2. lucrative 3. unprofitable . . . question: They do n't speak English and can not make ... understood. answer: themselves **Ground**: [them, us,you] **Prediction**: [1. *themselvess*, 2. **themselves** 3. **themselfs**, 4. **themselfes**, 5. *themselfis*, 6. *themselfies*, 7. **themselfves**, 8. *themselfselves*, 9. themselfiness, 10. themselfulated]
Dynamic-Demo-ChatGPT	**Prompt**: Generate 10 plausible but incorrect answers for the following question. question: They should choose ... would be best suited for the job answer: which incorrect answers: 1. whom 2. that 3. him who question: 36 . He does not speak English and can not make ... understood . answer: himself incorrect answers: 1. you 2. us 3. herself question: The new e-ticketing system , ... was installed last month , is working well . answer: which incorrect answers: 1. what 2. him 3. where ... question: They do n't speak English and can not make ... understood. answer: themselves **Ground**: [them, us,you] **Prediction**: [1.**theirself**, 2. **theirselves**, 3. **themself**, 4. **themselfs**, 5. **theirs**, 6. **us**, 7. they, 8. them is, 9. **theyself**, 10. **themselfes**]

References

[1] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. *Language models are few-shot learners*. Advances in neural information processing systems, 33:1877–1901, 2020.

[2] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, et al. *Training language models to follow instructions with human feedback*. Advances in Neural Information Processing Systems, 35:27730–27744, 2022.

[3] OpenAI. *GPT-4 Technical Report*, 2023. arXiv:2303.08774.

[4] G. Kurdi, J. Leo, B. Parsia, U. Sattler, and S. Al-Emari. *A systematic review of automatic question generation for educational purposes*. International Journal of Artificial Intelligence in Education, 30(1):121–204, 2020.

[5] S. K. Bitew, J. Deleu, A. S. Dogruöz, C. Develder, and T. Demeester. *Learning from Partially Annotated Data: Example-aware Creation of Gap-filling Exercises for Language Learning*. In Proceedings of the 18th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2023), pages 598–609, Toronto, Canada, July 2023. Association for Computational Linguistics. Available from: https://aclanthology.org/2023.bea-1.51.

[6] D. Ramesh and S. K. Sanampudi. *An automated essay scoring systems: a systematic literature review*. Artificial Intelligence Review, 55(3):2495–2527, 2022.

[7] A. P. Cavalcanti, A. Barbosa, R. Carvalho, F. Freitas, Y.-S. Tsai, D. Gašević, and R. F. Mello. *Automatic feedback in online learning environments: A systematic literature review*. Computers and Education: Artificial Intelligence, 2:100027, 2021.

[8] A. Gilson, C. W. Safranek, T. Huang, V. Socrates, L. Chi, R. A. Taylor, D. Chartash, et al. *How does CHATGPT perform on the United States Medical Licensing Examination? the implications of large language models for medical education and knowledge assessment*. JMIR Medical Education, 9(1):e45312, 2023.

[9] J. H. Choi, K. E. Hickman, A. Monahan, and D. Schwarcz. *Chatgpt goes to law school*. Available at SSRN, 2023.

[10] Y. Li, L. Sha, L. Yan, J. Lin, M. Raković, K. Galbraith, K. Lyons, D. Gašević, and G. Chen. *Can large language models write reflectively*. Computers and Education: Artificial Intelligence, 4:100140, 2023.

[11] P. Ramsden. *Learning to teach in higher education*. Routledge, 2003.

[12] M. J. Gierl, O. Bulut, Q. Guo, and X. Zhang. *Developing, analyzing, and using distractors for multiple-choice tests in education: A comprehensive review*. Review of Educational Research, 87(6):1082–1116, 2017.

[13] H. L. Roediger III and J. D. Karpicke. *Test-enhanced learning: Taking memory tests improves long-term retention*. Psychological science, 17(3):249–255, 2006.

[14] S. K. Bitew, A. Hadifar, L. Sterckx, J. Deleu, C. Develder, and T. Demeester. *Learning to Reuse Distractors to Support Multiple Choice Question Generation in Education*. IEEE Transactions on Learning Technologies, 2022. doi:10.1109/TLT.2022.3226523.

[15] S. Jiang and J. S. Lee. *Distractor generation for chinese fill-in-the-blank items*. In Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications, pages 143–148, 2017.

[16] Q. Guo, C. Kulkarni, A. Kittur, J. P. Bigham, and E. Brunskill. *Questimator: Generating knowledge assessments for arbitrary topics*. In IJCAI-16: Proceedings of the AAAI Twenty-Fifth International Joint Conference on Artificial Intelligence, 2016.

[17] G. A. Miller. *WordNet: a lexical database for English*. Communications of the ACM, 38(11):39–41, 1995.

[18] R. Mitkov, A. Varga, L. Rello, et al. *Semantic similarity of distractors in multiple-choice tests: extrinsic evaluation*. In Proceedings of the workshop on geometrical models of natural language semantics, pages 49–56, 2009.

[19] J. Pino, M. Heilman, and M. Eskenazi. *A selection strategy to improve cloze question quality*. In Proceedings of the Workshop on Intelligent Tutoring Systems for Ill-Defined Domains. 9th International Conference on Intelligent Tutoring Systems, Montreal, Canada, pages 22–32. Citeseer, 2008.

[20] J. Leo, G. Kurdi, N. Matentzoglu, B. Parsia, U. Sattler, S. Forge, G. Donato, and W. Dowling. *Ontology-based generation of medical, multi-term MCQs*. International Journal of Artificial Intelligence in Education, 29(2):145–188, 2019.

[21] A. Faizan and S. Lohmann. *Automatic generation of multiple choice questions from slide content using linked data*. In Proceedings of the 8th International Conference on Web Intelligence, Mining and Semantics, pages 1–8, 2018.

[22] T. Alsubait, B. Parsia, and U. Sattler. *Generating Multiple Questions From Ontologies: How Far Can We Go?* In Proceedings from the First International Workshop on Educational Knowledge Management (EKM 2014), Linköping, November 24, 2014, number 104, pages 19–30. Linköping University Electronic Press, 2014.

[23] A. Papasalouros, K. Kanaris, and K. Kotis. *Automatic Generation Of Multiple Choice Questions From Domain Ontologies.* e-Learning, 1:427–434, 2008.

[24] C. Liang, X. Yang, N. Dave, D. Wham, B. Pursel, and C. L. Giles. *Distractor generation for multiple choice questions using learning to rank*. In Proceedings of the thirteenth workshop on innovative use of NLP for building educational applications, pages 284–290, 2018.

[25] Y. Gao, L. Bing, P. Li, I. King, and M. R. Lyu. *Generating distractors for reading comprehension questions from real examinations*. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 33, pages 6423–6430, 2019.

[26] C. Y. Yeung, J. S. Lee, and B. K. Tsou. *Difficulty-aware distractor generation for gap-fill items*. In Proceedings of the The 17th Annual Workshop of the Australasian Language Technology Association, pages 159–164, 2019.

[27] X. Zhou, S. Luo, and Y. Wu. *Co-attention hierarchical network: Generating coherent long distractors for reading comprehension*. In Proceedings of the AAAI Conference on Artificial Intelligence, volume 34, pages 9725–9732, 2020.

[28] H.-L. Chung, Y.-H. Chan, and Y.-C. Fan. *A BERT-based Distractor Generation Scheme with Multi-tasking and Negative Answer Training Strategies.* In Findings of the Association for Computational Linguistics: EMNLP 2020, pages 4390–4400, Online, November 2020. Association for Computational Linguistics. doi:10.18653/v1/2020.findings-emnlp.393.

[29] D. Kalpakchi and J. Boye. *BERT-based distractor generation for Swedish reading comprehension questions using a small-scale dataset*. In Proceedings of the 14th International Conference on Natural Language Generation, pages 387–403, Aberdeen, Scotland, UK, August 2021. Association

for Computational Linguistics. Available from: https://aclanthology.org/2021.inlg-1.43.

[30] R. Rodriguez-Torrealba, E. Garcia-Lopez, and A. Garcia-Cabot. *End-to-End generation of Multiple-Choice questions using Text-to-Text transfer Transformer models*. Expert Systems with Applications, 208:118258, 2022.

[31] S. Panda, F. Palma Gomez, M. Flor, and A. Rozovskaya. *Automatic Generation of Distractors for Fill-in-the-Blank Exercises with Round-Trip Neural Machine Translation*. In Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop, pages 391–401, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi:10.18653/v1/2022.acl-srw.31.

[32] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, et al. *Language models are unsupervised multitask learners*. OpenAI blog, 1(8):9, 2019.

[33] J. Wei, X. Wang, D. Schuurmans, M. Bosma, brian ichter, F. Xia, E. H. Chi, Q. V. Le, and D. Zhou. *Chain of Thought Prompting Elicits Reasoning in Large Language Models*. In Advances in Neural Information Processing Systems, 2022. Available from: https://openreview.net/forum?id=_VjQlMeSB_J.

[34] L. Xue, N. Constant, A. Roberts, M. Kale, R. Al-Rfou, A. Siddhant, A. Barua, and C. Raffel. *mT5: A Massively Multilingual Pre-trained Text-to-Text Transformer*. In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 483–498, Online, June 2021. Association for Computational Linguistics. doi:10.18653/v1/2021.naacl-main.41.

[35] M. L. McHugh. *Interrater reliability: the kappa statistic*. Biochemia medica, 22(3):276–282, 2012.

[36] T. Sakai. *Evaluating information retrieval metrics based on bootstrap hypothesis tests*. IPSJ Digital Courier, 3:625–642, 2007.

[37] R. Wang, H. Wang, F. Mi, Y. Chen, R. Xu, and K.-F. Wong. *Self-Critique Prompting with Large Language Models for Inductive Instructions*. arXiv preprint arXiv:2305.13733, 2023.

4

Adapting Language Models to Gap-filling Exercise Generation for Language Learning

In this chapter, we adapt a language model to yet another educational task but a very specialized task of gap-fill grammar exercise generation for language learning in French. We create a real-world dataset of French gap-filling exercises covering an unknown combination of grammatical aspects. We introduce the task as an example-aware prediction of suitable gaps in texts based on partially annotated data. We propose and train a novel neural network architecture for the newly defined task based on a language model. We showcase that conditioning the model's output for a given input text on an example exercise of the envisioned exercise type, leads to an increased effectivenesss, compared to an example-independent baseline model. Additionally we analyse the model's ability to disentangle elementary exercise types, without being explicitly trained to do so, and we observe that it can recognize types to some extent, especially for the most commonly occurring types in the test set.

⋆⋆⋆

Learning from Partially Annotated Data: Example-aware Creation of Gap-filling Exercises for Language Learning

S.K. Bitew, J. Deleu, A. Seza Doğruöz, C. Develder and T. Demeester

In Proceedings of the 18th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2023)

Abstract Since performing exercises (including, e.g., practice tests) forms a crucial component of learning, and creating such exercises requires non-trivial effort from the teacher, there is a great value in automatic exercise generation in digital tools in education. In this paper, we particularly focus on automatic creation of gap-filling exercises for language learning, specifically grammar exercises. Since providing any annotation in this domain requires human expert effort, we aim to avoid it entirely and explore the task of converting existing texts into new gap-filling exercises, purely based on an example exercise, *without explicit instruction or detailed annotation* of the intended grammar topics. We contribute (i) a novel neural network architecture specifically designed for aforementioned gap-filling exercise generation task, and (ii) a real-world benchmark dataset for French grammar. We show that our model for this French grammar gap-filling exercise generation outperforms a competitive baseline classifier by 8% in F1 percentage points, achieving an average F1 score of 82%. Our model implementation and the dataset are made publicly available[1] to foster future research, thus offering a standardized evaluation and baseline solution of the proposed partially annotated data prediction task in grammar exercise creation.

4.1 Introduction

While digital education tools have been increasingly developed and deployed for over a decade, the e-learning sector has definitely boomed in the wake of COVID-19, even leading to a new Digital Education Action Plan from the European Commission.[2] As one application in e-learning, we particularly focus on language education, and specifically on the automatic generation of gap-filling grammar exercises. This type of exercises has been shown to be very effective in language learning, with a noticeable effect of such practice tests on students progress and is generally considered as a global measure of language proficiency [1]. Furthermore, automatic generation of exercises has been shown to produce relatively high quality exercises, for example, for multiple choice questions [2], demonstrating the potential effectiveness of reducing human effort and offering cost-effective

[1] https://github.com/semerekiros/GF2/
[2] https://education.ec.europa.eu/focus-topics/digital-education/action-plan

solutions towards personalized exercise generation. In terms of technology, recent developments in natural language processing, e.g., BERT [3], GPT-3 [4], InstructGPT [5], open up new opportunities for further upscaling and improving automatic generation of such tests/exercises.

In this paper we specifically propose to generate grammar exercises from existing texts, by inducing well-chosen gaps in a given input sentence, following a set of given example exercise sentences. Further, we aim to create models that can be trained on the exercises themselves, without further annotations. The latter implies that we want to forgo a fully supervised learning setting, because such models would require each gap in the available exercises to be manually annotated with additional metadata, such as the particular exercise type, e.g., for gap-filling exercises, a suitable category such as a verb tense. Thus, we focus on converting given input texts into gap-filling exercises, by mimicking the implicit rules underlying a given example exercise, rather than by following explicit instructions such as a prescribed exercise type.

Application scenario: Consider a language teacher, who just introduced a particular grammatical topic (e.g., a new verb tense), and needs the students to practice. The grammar topic of interest may need to be practiced in combination with particular other topics (e.g., related tenses already studied by the students). Given that gap-filling questions can be completed online and automatically assessed [6], the teacher creates a new gap-filling exercise, covering these combined grammar topics. The goal of our model is then to support the automatic creation of new exercises, based on that example exercise, by transforming other texts provided by the teacher into additional gap-filling exercises that target the same linguistic topics to be practiced, without explicit instructions by the teacher of which topics the model should include. This would allow the teacher to rapidly create new training material for the students, potentially more diverse, for example, in terms of topics of the texts, their temporal relevance, or the inherent linguistic difficulty.

Learning from partially annotated data: The scenario outlined above represents a learning task in between one-shot learning (i.e., learning from one example [7] and full supervision (i.e., based on the full annotation of all examples). On the one hand, the one-shot setting considers the example exercise as a single training instance defining the nature of the prediction task by the way it was constructed by the teacher (in this case, the included grammar topics). On the other hand, the fully supervised setting would require at least explicit knowledge of all exercise instructions (i.e., gap types

per exercise). Although we assume the availability of an entire corpus of such exercises, on overlapping grammar topics, we will not rely on explicit annotation of the nature of the gaps (i.e., gap type that defines the type/scope of the grammar exercise, or even just identifying the word category). Thus, we do want to learn from partially annotated examples, where the annotation is limited to just the indication of the gap and the text span that constitutes the expected answer. This basically amounts to the type of information that would be available in a one-/few-shot setting, but we aim to leverage the complete corpus to train our models.

Note that, while creating exercises, teachers are aware of the envisioned exercise type and the gap types, and such exercise type would also be communicated (e.g., as a free-text instruction) to students. Still, to keep our experiments and the gained insights transparent, we left out any exercise level instructions for our experiments.

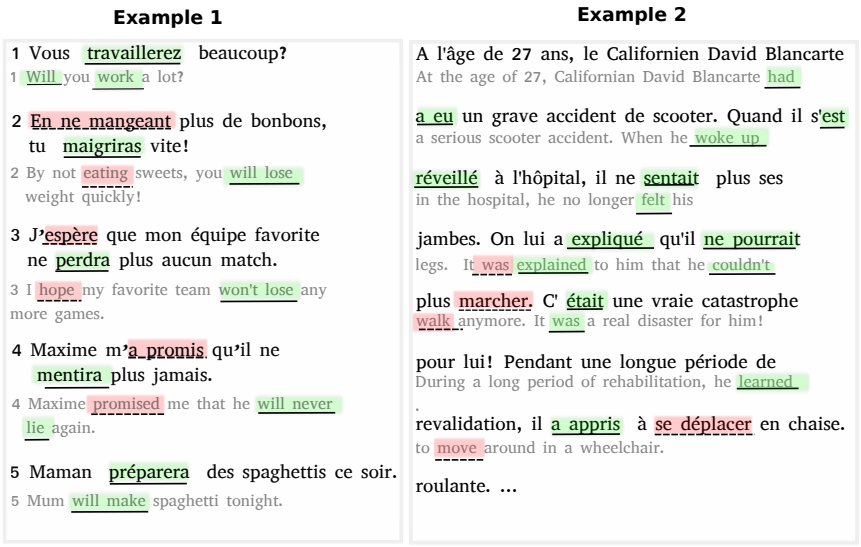


Figure 4.1: French grammar exercise from the GF2 corpus, with English translations for convenience shown in light grey. Green spans (with solid underline) are actual gaps as selected by teachers in the dataset, red spans represent potential gaps on other grammar topics but were not marked as gaps. (Left) Isolated sentence exercise with focus on a single tense (*futur simple*); (right) full text exercise combining two tense types (*imparfait* and *passé composé*).

Link with related research: In broad terms, the proposed work fits within the area of Automatic Question Generation (AQG) for the educational

domain. In the field of education, creating questions manually is an arduous task that demands considerable time, training, experience, and resources from educators [8]. As a solution to this challenge, researchers have turned towards AQG approaches to automatically generate homework, test, and exam exercises from readily available plain text that requires little to no human calibration. In particular, educational AQG systems have been developed for generating *factoid questions* covering several subjects such as history [9, 10], general sciences [11–13], health and biomedical sciences [14, 15], etc., as well as for *language learning* such as vocabulary or grammar exercises [16–18]. There has been some more generic recent work, however, on finding distractors for multiple choice questions across subjects and languages [19]. It is line with recent work on training deep neural networks for general-purpose question generation [20], based on large training sets. There is a clear preference for two question types that allow for automated assessment, i.e., multiple-choice questions (e.g., in [10, 12, 14, 15]) or gap-filling questions (as in [17, 18, 21, 22]).

Our work is focused on gap-filling questions, which typically require test-takers to fill in blank spaces in a text with missing word(s) omitted by test developers. The missing words can either be chosen from a set of possible answers (i.e., closed cloze questions), or generated from scratch using hints provided in the text (i.e., open cloze questions). To generate such questions, various strategies were employed, such as deleting every nth word from a text [23], or rationally deleting words according to specific purpose, e.g., usage of prepositions [24], verbs [25] etc. Previous studies have relied on selecting informative sentences [26, 27] from existing corpora, such as textbooks [28], WordNet [27], and then using techniques such as POS tagging [28] or term frequency analysis [2] to determine gap positions. More recently, [29], have developed sequence labeling model to automate the process of generating gap-filling exercises.

Another very relevant work by [30] devised a method to adapt an ELECTRA [31] model for the purpose of generating open cloze grammar exercises in English. Their approach involved classifying each individual token as either a gap or non-gap. However, there exist several notable distinctions between their approach and our own. Firstly, unlike their method that solely focused on individual tokens, we make gap decisions based on spans. This distinction is essential as our gaps can encompass multiple words, allowing for more comprehensive and contextually accurate grammar exercises. Secondly, our objective and experimental setup differ significantly. Our ultimate goal is to generate multiple versions of the same text, with each version targeting a distinct grammar aspect (e.g., future tense, prepositions of time or combinations of different types). In contrast, their approach

consistently produces exercises of the same type for a given input text (i.e., similar to our baseline model), lacking the versatility and adaptability our model offers.

We observed a tendency in generation of gap-filling questions aiming at well-defined tasks. To the best of our knowledge, none of the prior works have proposed strategies to capture common underlying structures in terms of task definition, while training on a heterogeneous set of real-world examples (e.g., covering various grammatical topics).

Key research contributions:

- We introduce the task of the example-aware prediction of suitable linguistic gaps in texts based on partially annotated data. This task is of paramount importance in the development of new gap-filling exercises.
- We present our real-world dataset of French gap-filling exercises covering unknown combinations of grammatical aspects. Our dataset called GF2 (*'Gap-Filling for Grammar in French'*) is released as a research benchmark for the introduced task.
- We propose and train a suitable neural network architecture for the task, and show that conditioning the model's output for a given input text on an example exercise of the envisioned exercise type, leads to an increased effectiveness, compared to an example-independent baseline model. Additionally we analyse the model's ability to disentangle elementary exercise types, without being explicitly trained to do so, and we observe that it can recognize types to some extent, especially for the most commonly occurring types in the test set.

4.2 Gap-filling Exercise Creation as a Span Detection Task

This section describes the particular prediction task this paper focuses on. We cast the creation of a French gap-filling exercise from an input text as a *binary span detection task*: the goal is detecting each span (i.e., consecutive sequence of tokens) that represents a correct gap. For clarity, we left out creating the 'hint' (e.g., the infinitive for verbs) which would make it a finalized gap-filling exercise, as it is considered less challenging and may deviate attention from the core problem of identifying the correct spans.

Figure 4.1 shows two example gap-filling exercises, with indication of the ground truth spans in green (and with solid underline). We denote

the distinguishing feature of each gap as its *gap type* (e.g., the tense *futur simple* for each of the valid tags in Example 1). An exercise typically covers multiple gap types, and the particular combination that characterizes a given exercise is called its *exercise type*. As such, many different exercise types can be constructed, and some may be unseen in the training data. For example, Example 2 (again in Fig. 4.1) combines three tenses (*imparfait*, *passé composé*, and *conditionnel présent*), which constitutes its exercise type. However, the same text could have been enriched with different gaps, corresponding to a different exercise type. In fact, our test set of one hundred exercises, for which we annotated gap types in terms of 12 elementary verb tenses, covers a total of 35 such composite exercise types.

Considering the lack of information regarding the exercise types for the training exercises, we further define the task we are examining more precisely. The objective is to detect the valid spans (i.e., spans that will be designated as gaps) of a given flat *input* text that mimics the same underlying exercise type as an example gap-filling exercise, which we denote as the *exemplar*. This exemplar serves as an indirect reference for the model to understand the desired exercise type. By utilizing this approach, we can better inform the model about the desired exercise type while accounting for the lack of exercise information available.

Note that our goal is working with real-world data. Our training data contains gap-filling examples following particular unknown exercise types. Moreover, teachers appear to not always select every possible span that satisfies the exercise type. We saw cases in our dataset (cf. Section 4.4.1), where the same verb occurring twice in the same form would be selected as a valid gap only once. Such real-world 'inconsistencies' contribute to the challenging nature of learning from such data without additional annotations.

4.3 Example-aware span detection model

This section describes our baseline model and proposed example-aware gap detection model. Figure 4.2 provides a schematic overview. We first detail the part indicated as *Baseline model*, inside the smaller dashed box, followed by the part that encodes the exemplar, which leads to the full model.

Baseline model: An input text $\mathbf{t}$, consisting of N tokens $\mathbf{t} = [t_0, t_1, \ldots, t_{N-1}]$ is encoded by a transformer based masked language model (MLM), in our experiments the multilingual XLM-RoBERTa [32]. From the corresponding transformer outputs $[\mathbf{h}_0, \mathbf{h}_1, \ldots, \mathbf{h}_{N-1}]$ (with $\mathbf{h}_i \in \mathbb{R}^k$, i=0...N-1), vector representations are constructed for all possible spans inside the input sequence,

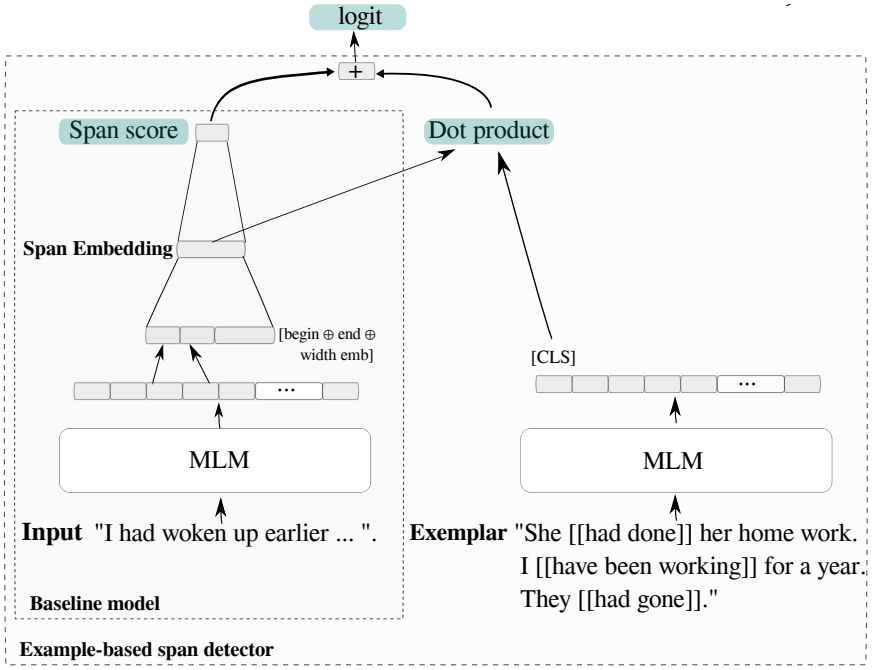


Figure 4.2: Example-aware gap detection model architecture. $\oplus$ denotes concatenation. In general, the model considers all possible spans up to a maximum width, but we depict here only one span from the input for brevity.

up to a certain length (in our experiments 12 tokens). The goal is then to make a binary prediction in terms of valid gaps, for each of these spans. In particular, for a span $\varsigma = [t_{\text{start}}, \ldots, t_{\text{end}}]$ with endpoint tokens t_{start}, t_{end} and width $|\varsigma| = (\text{end} - \text{start} + 1)$ in the input text, the corresponding span representation $\mathbf{h}_\varsigma$ is constructed as

$$\mathbf{h}_\varsigma = \text{FFNN}\big(\mathbf{h}_{\text{start}} \oplus \mathbf{h}_{\text{end}} \oplus \mathbf{h}_{|\varsigma|}\big)$$

in which $\oplus$ represents vector concatenation, $\mathbf{h}_{|\varsigma|}$ corresponds to a span width embedding, jointly learned with the model, and FFNN is a fully connected feed-forward model with a single hidden layer, ReLU activation, and output dimension k. The XLM-RoBERTa output representations $\mathbf{h}_{\text{start}}$ and $\mathbf{h}_{\text{end}}$ of the start and end token of ς are concatenated with the span width embedding $\mathbf{h}_{|\varsigma|}$, and transformed through FFNN into the k-dimensional span representation $\mathbf{h}_\varsigma$. The probability of span ς representing a valid gap is modeled as

$$p_{\text{base}}(\varsigma) = \sigma\big(\mathbf{w} \cdot \mathbf{h}_\varsigma + b\big)$$

in which the trainable parameters $\mathbf{w}$ and b are a k-length coefficient vector and bias, respectively, σ is the sigmoid function, and $\cdot$ represents the dot product. The baseline model is trained by minimizing the cross entropy loss between each span's score $p_{\text{base}}(\varsigma)$ and its label (1 for valid gaps, 0 otherwise). At inference, spans are predicted as gaps as soon as $p_\varsigma \geq 0.5$.

Example-aware gap detection model: As shown in Fig. 4.2, our example-aware model is a direct extension of the baseline model which by construction makes example-unaware predictions. The same MLM that encodes the input, is now used to also encode the exemplar, which contains the example exercise text as well as the correct gap information. The latter is added by surrounding each gap with the special tokens '[[' and ']]' (as seen in the figure). Details on how the examples are chosen, are provided in Section 4.4.2. The exemplar representation $\mathbf{h}_{exemplar}$ is obtained as the MLM's [CLS] representation[3].

We then quantify the compatibility of each span ς in the input text with the exemplar, through the dot product $\mathbf{h}_{\text{exemplar}} \cdot \mathbf{h}_\varsigma$ of their respective representations. In a direct extension of the baseline model, it leads to the proposed model for the probability $p_{\text{example-aware}}(\varsigma)$ that ς represents a valid gap:

$$p_{\text{example-aware}}(\varsigma) = \sigma\big(\mathbf{h}_\varsigma \cdot \mathbf{w} + \mathbf{h}_\varsigma \cdot \mathbf{h}_{\text{exemplar}} + b\big)$$

4.4 Empirical validation on real-world data

In this section, we first introduce the dataset that we will publicly release. Then, we explain how we train our models and use them for inference. Finally, we describe the strategies we adopted to evaluate the effectiveness of our models.

4.4.1 GF2 dataset: Gap-Fill for Grammar in French

We denote our new dataset as "Gap-Filling for Grammar in French" (GF2). It was contributed by Televic Education[4], and gathered through its education platform assessmentQ[5]. AssessmentQ is a comprehensive online platform for interactive workforce learning and high-stakes exams. It allows teachers

[3] [CLS] is a special token that is prepended to the input, and its corresponding output representation is pretrained to represent the entire sequence that is used for classification tasks

[4] https://www.televic.com/en/education

[5] https://www.televic-education.com/en/assessmentq

to compose their questions and answers for practice and assessment. As a result, the dataset is made up of a real-world set of gap-filling grammar exercise questions for French, manually created by experts. We cleaned and preprocessed the data before we could use it to train our models. First, organizational metadata information was removed. Other elements that we removed are the hints within the body of the text that could easily give away the gap positions, as well as inline instructions (if present) about the exercise type. Second, we automatically stripped off HTML tags from the documents. Our final dataset contains a total of 768 exercise documents, in which a total of 5,530 spans are tagged as gaps. The exercises were randomly split into 618 train documents, and 50 and 100 for validation and test, respectively. Table 4.1 summarizes GF2's descriptive statistics.

For the validation and test exercises, we made an extra manual effort to enrich each of the existing gaps with their gap type. Our annotations reflect the fact that the data contains a mix of verb and non-verb gaps. Every gap has an associated word type attribute (e.g. adverb, adjective, verb) and in case of verbs a tense attribute. In what follows we zoom in on the verb gaps and consider the tense as the main gap type. The bottom half of Table 4.1 shows the frequency of occurrence for the main verb types in the development and test documents. We use these annotations to get insights into the dataset and to evaluate the properties of our models (see Section 4.5). Note that the examples shown in Fig. 4.1 are actual entries from the GF2 dataset.

4.4.2 Training and inference

Our baseline model is relatively straightforward to train. We designate all spans indicated as gaps in our training data as valid gaps, which are considered positive examples. Conversely, any spans that are not indicated as gaps are labeled as negatives. We train our model by minimizing the cross entropy loss between each span's predicted score and its label as described in Section 4.3. However, training our example-aware model poses a challenge due to the lack of knowledge regarding the exercise types of the training exercises. Using one exercise as an example and another exercise of the same type as the input, along with the corresponding targets, is not therefore feasible. Instead, we make the assumption that exercises are generated by teachers who consistently follow the underlying exercise type throughout the entire exercise. As a result, we divide the training exercises into two parts: one part is used as an exemplar, and the other part serves as the actual input, for which the gaps are assumed to follow the same exercise type.

Table 4.1: Statistics of the GF2 dataset and breakdown into key verb tenses (gap types) in the validation and test split. For the train split we only know gap spans, not their types, since they are not labelled.

	Train	**Dev**	**Test**
# Documents	618	50	100
# Sentences	4786	378	707
# Gaps	4518	365	647
Subjonctif Présent (SPR)	UNK	1	28
Passé Composé (participe passé) (PCP)	UNK	31	8
Passé Composé (PC)	UNK	84	108
Imparfait (IM)	UNK	8	46
Conditionnel Présent (CPR)	UNK	23	92
Passé Récent (PR)	UNK	0	12
Futur Proche (FP)	UNK	1	9
Futur Simple (FS)	UNK	8	49
Indicatif Présent (IP)	UNK	126	144
Conditionnel Passé (CPA)	UNK	0	3
Impératif (IMP)	UNK	12	26
Plus-que-parfait (PQ)	UNK	0	1

To this end, we first segment each document in the training set into a list of sentences, along with their corresponding target gap positions. We create a new (exemplar, input) training pair by sampling one sentence to be used as the input, and uniformly sampling one up to m sentences from the remaining sentences within the same document to be used as the exemplar. The exemplar is constructed by concatenating these sampled sentences, with the addition of special symbols denoting the gap locations. (See Appendix 4.A for details.) These are the positive training examples that encourage the model to correctly learn predicting example-aware gaps. However, to facilitate efficient learning, it is crucial to also provide negative examples on which the model should not predict gaps. To create such negative training instances, a sentence is sampled as input from the considered document, but its span targets are set to zero (no gaps), and the negative exemplar is composed as before (including indicating the gaps), but by sampling sentences from a randomly selected *other* training exercise. There is risk of incidentally creating false negative training examples, if the exemplar gaps correspond with left-out gaps in the input. However, negative exemplars appeared important for obtaining a suitable model.

We determine the optimal proportion of negative to positive instances for training our models by employing a fine-tuning approach utilizing the

macro F1 score as the evaluation metric on the validation set. This increases the impact of the rarer gap types in the metric, and therefore in the final model, which we considered important for practical use. Other choices could have been made, however. Ultimately, the final model is trained on the union of the training and validation splits, using the optimal proportion determined via the fine-tuning process.

During inference, we use our trained model to predict the gap positions for an input text that is implicitly conditioned on the target exercise type through the exemplar.

Implementation and training details: We implement our models using pytorch and Huggingface. We initialize our MLM encoders with `xlm-roberta-base`. To avoid extensive hyper-parameter tuning, we made the following choices; a learning rate of 2e$-$5 in combination with the robust Adam optimizer. We use a batch size of 16 and train our models for 30 epochs. We consider all spans up to a maximum length 12 and we set k, the number of sentences per exemplar to 3.

Table 4.2: Tense disentangling ability in terms of precision, recall, and F1 (in %) on the test set, as reported for each key verb tenses (with on the right their support, i.e., number of occurrences). We also show the macro F1 score for the static baseline (*baseline*) and our proposed example-aware gap prediction (*ours*).

	Baseline			*Ours*			
Tenses	P	R	F1	P	R	F1	Support
SPR	5.0±0.3	78.6±8.9	9.4±0.6	7.5±0.2	81.0±12.5	13.7±0.4	28
PCP	0.1±0.1	4.2±6.3	0.2±0.3	12.6±4.1	62.5±12.5	20.7±6.2	8
PC	21.3±1.2	86.4±3.7	34.2±1.8	64 ±9.4	86.1±1.9	73.1±5.5	108
IM	9.3±0.4	88.4±3.7	16.2±0.8	12.0±2.5	78.3±10.9	20.9±3.9	46
CPR	19.9±0.5	94.5±2.9	32.8±0.8	28.3±2.9	92.4±4.7	43.2±3.1	92
PR	2.7±0.1	100.0±0.0	5.3±0.1	9.7±2.0	100.0±0.0	17.7±3.3	12
FP	1.6±0.0	77.7±0.0	3.1±0.1	6.0±0.9	77.8±0.0	11.1±1.5	9
FS	9.9±0.3	88.5±1.7	17.8±0.5	13.6±1.1	84.4±10	23.3±1.7	49
IP	24.6±1.2	75.0±4.3	37.1±1.9	32.0±1.4	66.2±11.9	42.9±2.4	144
CPA	0.1±0.1	11.1±16	0.2±0.3	0	0	0	3
IMP	5.2±0.3	88.5±2.2	9.9±0.5	16.8±1.7	84.6±3.9	25.3±2.1	26
PQ	0.2±0.0	100.0±0.0	0.5±0.0	0.6±0.1	100±0.0	1.2±0.2	1
Macro F1		13.9			**24.4**		

4.4.3 Evaluation setup

In order to assess and analyze the performance of the baseline and the example-aware model, we design two evaluation strategies that look at different effectiveness aspects.

Binary gap prediction evaluation: the primary objective of our model is to mimic the real-world setting where gap labels are not given. We measure how well our models predict gap positions (i.e., gap or no-gap decisions for all input spans). To do this, we split up each of the exercise documents in our test into two parts that are roughly the same size, given that by assumption they then represent the same exercise type. We calculate the automated metrics by using one half as the exemplar and the second as the input text to our model. We repeat this process by exchanging the roles of the parts. It is worth noting that we excluded one-sentence test documents (i.e., because they can not be chunked into two parts), which amount to 16% of the total test documents. However, since most of the excluded sentences (i.e., one-line documents) only had one gap, we only removed 2.7% of the total gaps in the test set.

Gap type disentangling evaluation: The goal of the second evaluation setting is to analyze how well the model has learned to disentangle individual gap types, despite not being explicitly trained to do so. This analysis is based on the assumption that a model that scores high on that aspect, would be stronger in dealing with new or rare exercise types. Potentially even at creating new combinations of existing exercises. This is an aspect we plan to study further when designing more advanced models in future research. To this end, we construct a small set of 12 exemplars, one for each of the key verb tenses, by randomly selecting them from the original data and subsequently removing them from the train/validation/test splits. Each exemplar comprises multiple sentences, all of which are homogeneously annotated with the same intended verb type, which will serve as the desired homogeneous exercise type. We evaluate our model on every sentence of the test set, by prompting it with each of these 12 fixed exemplars. Based on the gap types we annotated on the test set, we can then compute the precision, recall and F1 score for each of these 12 tenses.

4.5 Experimental Results

In this section, we provide evidence of the effectiveness of our proposed model by reporting and discussing the experimental results. Table 4.3

summarizes the binary gap prediction evaluation of the baseline vs. the example-aware model on the test set. We report our results as the mean and standard deviation over five runs, each using a different random seed for model training. The proposed example-aware model (denoted as *ours*) consistently outperforms the example-unaware *baseline* on all metrics. In general, there is an absolute gain of 8 percentage points in F1 for the proposed model in comparison with the baseline, achieving an average F1 score of 82.4%. This confirms our intention when designing the model, that providing example exercises leads to an increased effectiveness in terms of predicting gap positions compared to the static baseline model.

Table 4.3: Overall binary gap prediction in terms of precision, recall, and F1 (in %) on the test set. Results shown for the static baseline (*baseline*) and our proposed example-aware gap prediction (*ours*).

	Precision	**Recall**	**F1**
Baseline	$74.87_{\pm 2.44}$	$73.11_{\pm 2.00}$	$73.92_{\pm 0.49}$
Ours	$84.30_{\pm 1.70}$	$80.74_{\pm 1.80}$	$82.40_{\pm 0.20}$

In Table 4.2, we show the evaluation of our models in their ability to disentangle the 12 main verb types. We observe that for the tenses with relatively higher support, the example-aware model outperforms the baseline with certainty as demonstrated by the individual F1 scores.

The overall macro F1 score for the example-aware model stands at 24.4%, which is low in absolute value, but considerably higher than the baseline's macro F1 score of 13.9%. We observe that the proposed model is able to recognize verb types such as passé compassé (PC), imparfait (IM), and conditionnel présent (CPR) to some extent with F1 scores of 73%, 43%, and 42%, respectively. However, the low overall scores are not unexpected, because the models are not trained to recognize gap types. Furthermore, some tenses are either very rare (e.g., PQ, CPA, PCP) as indicated by their support, or may appear mainly in combination with other exercise types. This makes achieving a better resolution in disentangling gap types without any explicit gap labels during training an inherently difficult task.

4.6 Conclusion

In this paper, we introduced a new task within the general challenge of training models to automatically create new exercises for use in education, based on existing exercises and without requiring additional manual annotations.

In particular, we introduced a dataset and associated prediction task,

focusing on detecting gaps within a given input text, without knowledge of the exact exercise type, by only relying on an example exercise. We proposed an example-aware neural network model designed for this task, and compared it with a baseline model that does not take into account any example of the desired exercise type. We found that our example-aware model outperforms the baseline model not only in predicting gaps, but also in disentangling gap types despite not being explicitly trained on that task. Our real-world GF2 dataset of French gap-filling exercises will be publicly released together with the code to reproduce the presented empirical results.

The presented work fits with our pursuit towards supporting personalized learning experiences by either suggesting existing or generating new exercises that are tailored to students' needs. Teachers could also benefit from an increased efficiency in creating new exercises. For example, they could make many and diverse drill and practice exercises on chunks of text based on existing standard exercise types without having to provide extra metadata information such as instructions. We hope our benchmark dataset and task will spark new research in the CL and Educational NLP community.

Limitations

We identify two limitations of the current work and make suggestions for future directions. First, while our proposed method is language-agnostic in principle, our evaluation is limited to our French benchmark dataset. Expanding our approach to encompass other languages would bring new and interesting challenges for further investigation. Second, despite topic diversity within our exercise documents (e.g., the first example in Fig. 4.1 consists of independent sentences, while the second is a coherent text centered around the same topic.), it would be interesting to quantify the degree of topical bias introduced during our training process and its impact on our binary task evaluation. For future work, we first aim to adapt seq2seq models for our task particularly text-to-text models such as T5 [33]. There is also potential to explore different prompting strategies for large language models (LLMs), when generating gap-filling grammar exercises. For instance, the utilization of chain-of-thought prompting [34], which involves generating intermediate steps before producing the final response, could be explored for generating grammar exercises. Additionally, an interesting future study would involve investigating the number of example demonstrations that LLMs require in order to accurately mimic example gap exercises.

Ethics Statement

In this research, we posit that the dataset and models introduced are of low-risk in terms of potential harm to individuals. The dataset used is a curated selection of existing educational content enriched with meta-data, and we are confident that our compilation of the dataset has not introduced any additional ethical risks. However, it is crucial to emphasize the need for accountability and the establishment of clear guidelines for the deployment of grammar generation models, such as the ones benchmarked in this paper, for educational purposes.

It should be noted that our models are derived from general-purpose neural language encoders that have been trained on real-world data, which may contain biases or discriminatory content [35]. As a result, our models may have inherited some of these biases and could potentially base their prediction on such biased information. Therefore, it is imperative for educators and researchers to thoroughly consider these ethical issues and ensure that the generated grammar questions align with educational goals and do not perpetuate harmful biases.

Educators should retain the final authority in accepting or modifying grammar question suggestions generated by such models, keeping their educational goals in mind (e.g., in terms of formative and especially summative assessment). In practice, these models are designed to enhance teachers' efficiency in preparing teaching materials, rather than replacing teachers in any way. An important benefit of using AI-supported question generation with increased efficiency is the potential for personalized approaches towards students.

Acknowledgements

This work was funded by VLAIO ('Flanders Innovation & Entrepreneurship') in Flanders, Belgium, through the *imec-icon* project AIDA ('AI-Driven e-Assessment'). This research also received funding from the Flemish Government under the "Onderzoeksprogramma Artificiële Intelligentie (AI) Vlaanderen" programme. We would like to thank the AIDA partners Televic Education and WeZooz Academy for contributing data and use cases.

4.7 Appendix

4.A Training details

In this section we detail our training procedure. As depicted in Fig. 4.3, we first split training exercises into list of sentences, along with their corresponding gap position indications. In order to create new (input, exemplar) pair, we sample 1 sentence from the sentence list to be used as our *input* text, and we uniformly sample 1 up to m (we set $m = 3$) sentences from the remaining sentence list to be used as our exemplar. We form our exemplar by concatenating all the sampled sentences with gap positions indicated by special tokens "[[" and "]]". Then our model is trained by minimizing the binary cross entropy (BCE) loss between predicted gaps and their target labels (1 for valid gaps, and 0 otherwise).

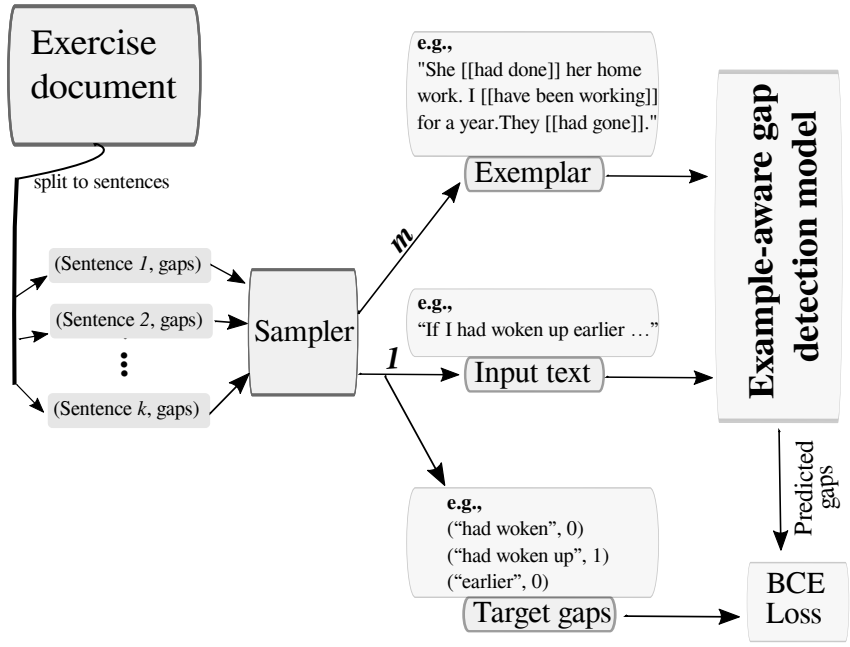


Figure 4.3: Training procedure of our example-aware gap detection model. First, we split exercise documents into list of sentences. Then we create (input, exemplar) training pairs that will be used by our model. We use one sentence as an input, while the exemplar is made up of sentences that are uniformly sampled from the remaining sentences. The exemplar is constructed by concatenating the m sampled sentences. The special symbols "[[" and "]]" in the exemplar indicate the gap positions. Binary cross entropy (BCE) loss is used to train our models.

References

[1] J. W. Oller Jr. *Cloze tests of second language proficiency and what they measure 1*. Language learning, 23(1):105–118, 1973.

[2] R. Mitkov, H. Le An, and N. Karamanis. *A computer-aided environment for generating multiple-choice test items*. Natural language engineering, 12(2):177–194, 2006.

[3] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. *Bert: Pre-training of deep bidirectional transformers for language understanding*. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), 2019.

[4] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. *Language models are few-shot learners*. Advances in neural information processing systems, 33:1877–1901, 2020.

[5] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, et al. *Training language models to follow instructions with human feedback*. Advances in Neural Information Processing Systems, 35:27730–27744, 2022.

[6] T. Daradoumis, J. M. M. Puig, M. Arguedas, and L. C. Liñan. *Analyzing students' perceptions to improve the design of an automated assessment tool in online distributed programming*. Computers & Education, 128:159–170, 2019.

[7] Y. Wang, Q. Yao, J. T. Kwok, and L. M. Ni. *Generalizing from a few examples: A survey on few-shot learning*. ACM computing surveys (csur), 53(3):1–34, 2020.

[8] B. G. Davis. *Tools for teaching*. John Wiley & Sons, 2009.

[9] M. Al-Yahya. *OntoQue: a question generation engine for educational assesment based on domain ontologies*. In 2011 IEEE 11th International Conference on Advanced Learning Technologies, pages 393–395. IEEE, 2011.

[10] A. Papasalouros, K. Kanaris, and K. Kotis. *Automatic Generation Of Multiple Choice Questions From Domain Ontologies*. e-Learning, 1:427–434, 2008.

[11] B. Sun, Y. Zhu, Y. Xiao, R. Xiao, and Y. Wei. *Automatic question tagging with deep neural networks*. IEEE Transactions on Learning Technologies, 12(1):29–43, 2018.

[12] K. Stasaski and M. A. Hearst. *Multiple choice question generation utilizing an ontology*. In Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications, pages 303–312, 2017.

[13] R. Conejo, E. Guzmán, and M. Trella. *The SIETTE automatic assessment environment*. International Journal of Artificial Intelligence in Education, 26(1):270–292, 2016.

[14] D. Pugh, A. De Champlain, M. Gierl, H. Lai, and C. Touchie. *Using cognitive models to develop quality multiple-choice questions*. Medical teacher, 38(8):838–843, 2016.

[15] N. Afzal and R. Mitkov. *Automatic generation of multiple choice questions using dependency-based semantic relations*. Soft Computing, 18(7):1269–1281, 2014.

[16] Y. Susanti, T. Tokunaga, H. Nishikawa, and H. Obari. *Evaluation of automatically generated english vocabulary questions*. Research and practice in technology enhanced learning, 12(1):1–21, 2017.

[17] J. Hill and R. Simha. *Automatic generation of context-based fill-in-the-blank exercises using co-occurrence likelihoods and Google n-grams*. In Proceedings of the 11th Workshop on Innovative Use of NLP for Building Educational Applications, pages 23–30, 2016.

[18] T. Goto, T. Kojiri, T. Watanabe, T. Iwata, and T. Yamada. *Automatic generation system of multiple-choice cloze questions and its evaluation*. Knowledge Management & E-Learning: An International Journal, 2(3):210–224, 2010.

[19] S. K. Bitew, A. Hadifar, L. Sterckx, J. Deleu, C. Develder, and T. Demeester. *Learning to Reuse Distractors to Support Multiple Choice Question Generation in Education*. IEEE Transactions on Learning Technologies, 2022.

[20] X. Du, J. Shao, and C. Cardie. *Learning to Ask: Neural Question Generation for Reading Comprehension*. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1342–1352, Vancouver, Canada, July 2017. Association for Computational Linguistics. Available from: https://aclanthology.org/P17-1123, doi:10.18653/v1/P17-1123.

[21] A. Malinova and O. Rahneva. *Automatic generation of english language test questions using mathematica*. In CBU International Conference Proceedings, volume 4, pages 906–909, 2016.

[22] L. Perez-Beltrachini, C. Gardent, and G. Kruszewski. *Generating grammar exercises*. In The 7th Workshop on Innovative Use of NLP for Building Educational Applications, NAACL-HLT Worskhop 2012, pages 147–157, 2012.

[23] W. L. Taylor. *"Cloze procedure": A new tool for measuring readability*. Journalism quarterly, 30(4):415–433, 1953.

[24] J. Lee and S. Seneff. *Automatic generation of cloze items for prepositions*. In Eighth Annual Conference of the International Speech Communication Association, 2007.

[25] E. Sumita, F. Sugaya, and S. Yamamoto. *Measuring non-native speakers' proficiency of english by using a test with automatically-generated fill-in-the-blank questions*. In Proceedings of the second workshop on Building Educational Applications Using NLP, pages 61–68, 2005.

[26] V. Slavuj, L. N. Prskalo, and M. B. Bakaric. *Automatic generation of language exercises based on a universal methodology: An analysis of possibilities*. Bulletin of the Transilvania University of Brasov. Series IV: Philology and Cultural Studies, pages 29–48, 2021.

[27] J. Pino, M. Heilman, and M. Eskenazi. *A selection strategy to improve cloze question quality*. In Proceedings of the Workshop on Intelligent Tutoring Systems for Ill-Defined Domains. 9th International Conference on Intelligent Tutoring Systems, Montreal, Canada, pages 22–32, 2008.

[28] M. Agarwal and P. Mannem. *Automatic gap-fill question generation from text books*. In Proceedings of the sixth workshop on innovative use of NLP for building educational applications, pages 56–64, 2011.

[29] E. Marrese-Taylor, A. Nakajima, Y. Matsuo, and O. Yuichi. *Learning to Automatically Generate Fill-In-The-Blank Quizzes*. In Y.-H. Tseng, H.-H. Chen, V. Ng, and M. Komachi, editors, Proceedings of the 5th Workshop on Natural Language Processing Techniques for Educational Applications, pages 152–156, Melbourne, Australia, July 2018. Association for Computational Linguistics. Available from: https://aclanthology.org/W18-3722, doi:10.18653/v1/W18-3722.

[30] M. Felice, S. Taslimipoor, and P. Buttery. *Constructing Open Cloze Tests Using Generation and Discrimination Capabilities of Transformers*. In

Findings of the Association for Computational Linguistics: ACL 2022, pages 1263–1273, Dublin, Ireland, May 2022. Association for Computational Linguistics. Available from: https://aclanthology.org/2022.findings-acl.100, doi:10.18653/v1/2022.findings-acl.100.

[31] K. Clark, M.-T. Luong, Q. V. Le, and C. D. Manning. *ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators*. In International Conference on Learning Representations, 2020. Available from: https://openreview.net/forum?id=r1xMH1BtvB.

[32] A. Conneau, K. Khandelwal, N. Goyal, V. Chaudhary, G. Wenzek, F. Guzmán, E. Grave, M. Ott, L. Zettlemoyer, and V. Stoyanov. *Unsupervised Cross-lingual Representation Learning at Scale*. In D. Jurafsky, J. Chai, N. Schluter, and J. Tetreault, editors, Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 8440–8451, Online, July 2020. Association for Computational Linguistics. Available from: https://aclanthology.org/2020.acl-main.747, doi:10.18653/v1/2020.acl-main.747.

[33] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. *Exploring the limits of transfer learning with a unified text-to-text transformer*. The Journal of Machine Learning Research, 21(1):5485–5551, 2020.

[34] J. Wei, X. Wang, D. Schuurmans, M. Bosma, brian ichter, F. Xia, E. H. Chi, Q. V. Le, and D. Zhou. *Chain of Thought Prompting Elicits Reasoning in Large Language Models*. In A. H. Oh, A. Agarwal, D. Belgrave, and K. Cho, editors, Advances in Neural Information Processing Systems, 2022. Available from: https://openreview.net/forum?id=_VjQlMeSB_J.

[35] R. Bommasani, D. A. Hudson, E. Adeli, R. Altman, S. Arora, S. von Arx, M. S. Bernstein, J. Bohg, A. Bosselut, E. Brunskill, et al. *On the opportunities and risks of foundation models*. arXiv preprint arXiv:2108.07258, 2021.

5

Adapting Coreference Resolution to new target languages

In this chapter, we extend our adaptability theme of the thesis to adapting a fundamental NLP task – conference resolution – to new languages. We explore the appealing idea of leveraging translation tools for bootstrapping coreference resolution in languages with limited resources. We propose and analyze two strategies (i) translate the training data in high-resource language to the target language and train a coreference model and (ii) translate the test data into high-resource source language (e.g., English) and use a trained coreference model for inference. Moreover, we study the source of errors for these two strategies and reveal that in fact the quality of contemporary machine translation tools is the main limiting factor.

⋆ ⋆ ⋆

Lazy Low-Resource Coreference Resolution: a Study on Leveraging Black-Box Translation Tools

S.K. Bitew, J. Deleu, C. Develder and T. Demeester

In Proceedings of the Fourth Workshop on Computational Models of Reference, Anaphora and Coreference (CRAC) at EMNLP 2021.

Abstract Large annotated corpora for coreference resolution are available for few languages. For machine translation, however, strong black-box systems exist for many languages. We empirically explore the appealing idea of leveraging such translation tools for bootstrapping coreference resolution in languages with limited resources. Two scenarios are analyzed, in which a large coreference corpus in a high-resource language is used for coreference predictions in a smaller language, i.e., by machine translating either the training corpus, or the test data. In our empirical evaluation of coreference resolution using the two scenarios on several medium-resource languages, we find no improvement over monolingual baseline models. Our analysis of the various sources of error inherent to the studied scenarios, reveals that in fact the quality of contemporary machine translation tools is the main limiting factor.

5.1 Introduction

End-to-end coreference resolution is the task of identifying and clustering all spans of text that refer to the same entity in a document. It serves as an important step for several downstream NLP tasks that involve natural language understanding, including question answering [1], information retrieval, and text summarization [2, 3]. Recent advances in deep learning have resulted in state-of-the-art performance on coreference resolution [4–8]. The performance of these models, however, highly depends on the existence of large annotated datasets. Still, for many languages that lack large annotated coreference corpora, Machine Translation (MT) tools of an ever increasing quality are available. The idea studied in this work, is whether existing black-box translation tools can be readily leveraged for transferring the task of coreference resolution from one language to another.

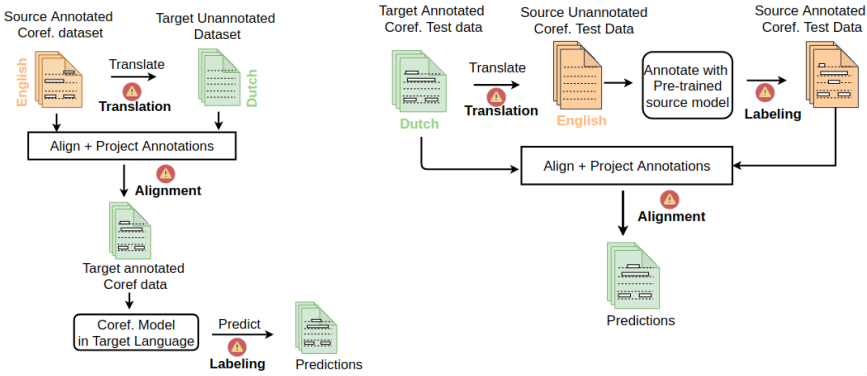
We tackle the setting in which a large labeled corpus exists in a resource-rich language (i.e., the 'source' language) whereas only a smaller corpus exists in a smaller-resource language (called the 'target' language). Specifically, we consider two scenarios in which black-box MT tools can be integrated into a cross-lingual end-to-end coreference resolution system. The first scenario, `Translate-train`, uses an MT tool to translate the large source corpus into the target language, after which a coreference model is trained in the target language. In the second scenario, `Translate-test`, test examples in the target language are first machine translated to the source language, after which a pre-trained coreference model is used to predict the labels. The second scenario has the disadvantage that an MT tool is required at inference time.

Similar transfer learning setups for basic sequence tagging tasks gave

encouraging results (as discussed in Section 5.4), but we find this is no longer the case for the task of coreference resolution.

We analyze the different sources of error related to integrating the MT tool in the pipeline. As it turns out, translation errors have the strongest impact on the effectiveness of the proposed methods, followed by prediction errors and alignment issues.

5.2 Approach



(a) Translate-train procedure

(b) Translate-test procedure

Figure 5.1: Annotation projection approaches, with indication of the main sources of error through the ⚠ icon.

5.2.1 Translate-train

The goal of the `Translate-train` approach (visualized in Fig. 5.1a) is to create a dataset in the target language, on which a model can be trained. We follow the approach used by [9] for NER, but we now apply it for coreference resolution.

We assume access to labeled training data in the source language, an MT tool, an alignment tool, and a test set in the target language. First, we use the MT tool to translate the entire training set from the source to the target language. This results in an unannotated dataset in the target language. Second, we identify and label all mentions of entities in the translated target document by aligning the source and target documents using the alignment tool. Finally, a competitive monolingual method is used to train a coreference model directly in the target language, which is then evaluated on the test data.

5.2.2 Translate-test

The Translate-test approach (see Fig. 5.1b) follows [10], and assumes access to a large training corpus in the source language, an off-the-shelf MT system, an alignment tool, and a test set in the target language. First, the test set is translated into the source language. A competitive model trained on the source language training corpus is used to annotate the translated test set (i.e., identify and cluster mentions into groups). With the alignment tool, the translated documents in the source language are aligned with the original ones in the target language, after which the predicted labels are projected onto them for evaluation.

Table 5.1: SemEval-2010 Dataset Statistics

	Training			Development			Test		
	#docs	**#sents**	**#tokens**	**#docs**	**#sents**	**#tokens**	**#docs**	**#sents**	**#tokens**
Catalan	829	8,709	253,513	142	1,445	42,072	167	1,698	49,260
Dutch	145	2,544	46,894	23	496	9,165	72	2,410	48,007
English	229	3,648	79,060	39	741	17,044	85	1,141	24,206
Italian	80	2,951	81,400	17	551	16,904	46	1,494	41,586
Spanish	875	9,022	284,179	140	1,419	44,460	168	1,705	51,040

Table 5.2: Monolingual and Cross-Lingual results in terms of Average Coreference F1

Method	**Alignment tool**	**Dutch**	**Spanish**	**Catalan**	**Italian**
Translate-train	Fast-Align	0.280	0.410	0.410	0.340
Translate-train	Heuristics	0.260	0.390	0.370	0.307
Translate-test	Fast-Align	0.365	0.461	0.480	0.362
Translate-test	Heuristics	0.358	0.438	0.453	0.347
End2end Coref	-	**0.380**	**0.516**	**0.533**	0.430
Sucre or Tan-l*	-	0.191	0.490	0.482*	**0.607**

5.3 Experimental Evaluation

Data — Our evaluation set was created for the SemEval-2010 [11] shared task, and contains coreference annotations for six languages (see Table 5.1 for dataset statistics). We use Dutch, Spanish, Italian, and Catalan as our target languages, and the corresponding SemEval-2010 datasets are used to train and test the respective monolingual coreference models. As our

large and high-quality source dataset, we use the English OntoNotes 5.0 coreference dataset from the CoNLL 2012 shared task [12].

Coreference Models — For the `Translate-train` scenario, we use the end-to-end neural coreference resolution method from [4] to train and evaluate on the target languages. This model considers all spans of text as potential mentions and finds the most probable antecedents for each span. For each span, a span ranking model is used to decide which of the previous spans are good antecedents, whereby a trained pruner eliminates less likely mentions. During training, the marginal log-likelihood of all correct antecedents in the gold clusters is optimized. In our `Translate-test` experiments, we use `SpanBert` [7], an English end-to-end coreference resolver, trained on the OntoNotes corpus.

Translation Tool — In both scenarios, we use Google Translate[1] as our publicly available MT tool of choice.

Alignment — For the alignment step, we compare Fast-Align from [13], a simple unsupervised statistical word alignment model, with the `Heuristics` method from the work of [9].

Baselines — We compare our `translate-train` and `translate-test` approaches with a model trained on annotated data in the target language (i.e., `End2end Coref`). We also consider two alternative baseline systems for which end-to-end coreference results were reported on the SemEval 2010 shared task data: `Sucre` and `Tan-l`. The `Sucre` system [14] uses engineered features for words, mentions and mention pairs and uses classical machine learning classifiers to cluster mentions. It reports the best results for Spanish, Italian and Dutch. The `Tan-l` system [15] uses dependency parse trees to detect mentions and trains a binary classifier to decide the pairwise relationship between the extracted mentions and reports the best result for Catalan. Works such as [16] and [17] are not used as baselines because they make use of external resources (mention detectors, NER, Alpino parse trees[2], etc.).

Metrics — For evaluation, we report the average F1 of the MUC, B3, and CEAF4 coreference resolution metrics, as proposed in [18].

5.3.1 Results

Our end-to-end monolingual baseline outperforms the `Sucre` and `Tan-l` systems on Catalan, Spanish and Dutch, as shown in Table 5.2. For Italian, our baseline shows inferior performance, possibly due to the small number of training examples (i.e., only 80 documents). Interestingly, our cross-lingual models remain unable to surpass the effectiveness of their monolingual

[1]The translation of documents using Google Translate was done on 02-12-2020.

[2]`http://www.let.rug.nl/vannoord/alp/Alpino/`

counterparts, although the former leverage a much larger coreference corpus than the latter. The Translate-test is consistently better than Translate-train, which we hypothesize is due to the superior quality of the English SpanBert model, especially in comparison with the End2end Coref models trained on the translated (i.e., noisy) source corpus. The Fast-Align alignment strategy consistently outperforms the Heuristics based alignment method in both the Translate-train and Translate-test approaches. [9] showed that the Heuristics improved on their Fast-align and indicated the reason to be that named entities are low-frequency words. To improve its performance, we trained Fast-Align on the additional parallel corpus Europarl [19].

5.3.2 Error Analysis

In this section we discuss the contributing factors to the low performance of the Translate-test setup (being the better of both scenarios). From 10 randomly sampled test documents, which contain a total of 424 mentions and 127 mention clusters, we quantify three particular sources of error (see Fig. 5.1b):

Table 5.3: Literal translation error (1 & 2) and pronoun mistranslation (3 & 4) examples

	Source text	**Google Translate**	**Correct translation**
1.	Het gesprek ging onder meer over *[Punt].*	The conversation was about *[Dot].*	The conversation, amongst others, was about *[Punt]*
2.	*[Mark Grammens]* ...	*[Mark Grams man]* ...	*[Mark Grammens]* ...
3.	...en *[zijn]* stelling is bekend	...and *[its]* position is well known	...[his] position is well known
4.	*[Die]* nam daar genoegen mee.	*[Which]* was content with that.	*[He]* was content with that

Translation Error — To measure the impact of the imperfect translation step, we annotate the Dutch-to-English translated documents with coreference labels (i.e., perfect annotation on the noisy translations). We also manually align the noisy English documents with the original Dutch documents (i.e., to simulate perfect alignment).

Automatic Labeling Error — To see the impact of the prediction model, we use SpanBert to annotate the manually translated documents (i.e., assuming

Table 5.4: Error breakdown for a random sample of 10 Dutch SemEval-2010 documents.

Model	F1
Translate-test	0.415
only *translation* error	0.490
only *labeling* error	0.613
only *alignment* error	0.896

perfect translation), again followed by a manual alignment step (i.e., to avoid alignment errors).

Alignment Error — To quantify the noise induced by the alignment step, we manually translate the documents to English and manually assign the coreference labels, after which Fast-Align is applied for alignment with the original Dutch documents for evaluation.

Our analysis on the error breakdown is shown in Table 5.4. The largest source of error for the translate-test model appears to be the MT step followed by the labeling error, whereas the impact of the alignment error is rather small. We looked into the translation errors, and observed that the coreference results are most degraded due to incorrectly translated pronouns, and literal translations of (parts of) named entities.

The labeling error leads to a hypothetical F1 (i.e., in the absence of other errors) of 0.613 on the selected documents. This is considerably below the reported SpanBert performance of 0.796 on the Ontonotes test set [7]. We hypothesize this is partly due to the shift in domain between the English Ontonotes data and the SemEval data in Dutch, as well as some differences in coreference annotation guidelines between both datastets. For example, coreference relations with verbs are annotated in Ontonotes but not in SemEval.

5.4 Related Work

The key concept used in the presented transfer learning scenarios, is *annotation projection*, as originally proposed by [20] for part-of-speech tagging. It relies on the transfer of annotations from the source language to the target language. Most annotation projection methods depend on parallel corpora in which the source data is labeled using a trained model before projecting the labels onto the data in the target language [21–28].

Alternatively, other works relied on the use of bilingual dictionaries for annotation projection [29, 30]. The Translate-train idea of creating a

noisy translated corpus with projected annotations has been proposed as well [9, 31], for the task of dependency parsing and NER, respectively. [10] used MT in the other direction (`Translate-test`) for the task of NER.

A common problem in both annotation projection scenarios is the alignment of text spans between languages, for which unsupervised statistical alignment models can be used [10, 27], such as the IBM models 1-6 [32, 33]. A few recent works [29, 30] perform translation on a word or span level to avoid the alignment problem. Others explored alignment heuristics such as matching words based on their surface forms and translations [9, 24], or using external information such as Wikipedia links [34, 35].

The prior works applied annotation projection to the tasks of NER, POS, or dependency parsing, and proved relatively successful (i.e., close to monolingual models in the target language). [22, 25] are notable prior works that applied the idea of annotation projection to the task of coreference resolution. Unlike our work, they depend on the existence of parallel corpora and are focused on a single language pair to test their ideas. Moreover, they have a pipeline that extracts mentions using external annotation tools, or even manually, before clustering them into coreference chains. We, however, perform both the mention identification and clustering in a span-based end-to-end fashion.

For the task of end-to-end coreference resolution, we explore using machine translation for annotation projection, especially with medium-resource languages for which strong MT systems exist. We investigate if MT systems can be used for transferring coreference knowledge (model, dataset) without having to rely on parallel corpora.

5.5 Conclusion and Future work

While the idea of leveraging MT to improve NLP task performance for low resource languages is not new, this idea to the best of our knowledge has not been pursued for coreference resolution. We contribute by comparing two conceptually different methods; the `Translate-train` and `Translate-test` approaches. We further present a rigorous quantitative error analysis. From our work, we conclude that (i) for coreference resolution the MT approaches are not very successful. (ii) our error analysis suggests this is mainly due by translation errors followed by labeling and alignment errors.

We believe MT models can still be leveraged in cross-lingual transfer learning for coreference resolution, but we speculate that access to the internals of the models, such as attention weights, will be needed. Moreover, future work will need to investigate hybrid strategies, combining transfer learning from other languages with the available data in the target lan-

guage, to override issues due to MT uncertainty or differences in annotation guidelines.

References

[1] T. S. Morton. *Using coreference for question answering*. In Coreference and Its Applications, 1999.

[2] S. Azzam, K. Humphreys, and R. Gaizauskas. *Using coreference chains for text summarization*. In Coreference and Its Applications, 1999.

[3] B. Baldwin and T. S. Morton. *Dynamic coreference-based summarization*. In Proceedings of the Third Conference on Empirical Methods for Natural Language Processing, pages 1–6, 1998.

[4] K. Lee, L. He, M. Lewis, and L. Zettlemoyer. *End-to-end Neural Coreference Resolution*. In M. Palmer, R. Hwa, and S. Riedel, editors, Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pages 188–197, Copenhagen, Denmark, September 2017. Association for Computational Linguistics. Available from: https://aclanthology.org/D17-1018, doi:10.18653/v1/D17-1018.

[5] H. Fei, X. Li, D. Li, and P. Li. *End-to-end deep reinforcement learning based coreference resolution*. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 660–665, 2019.

[6] B. Kantor and A. Globerson. *Coreference resolution with entity equalization*. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 673–677, 2019.

[7] M. Joshi, D. Chen, Y. Liu, D. S. Weld, L. Zettlemoyer, and O. Levy. *Spanbert: Improving pre-training by representing and predicting spans*. Transactions of the Association for Computational Linguistics, 8:64–77, 2020.

[8] W. Wu, F. Wang, A. Yuan, F. Wu, and J. Li. *CorefQA: Coreference resolution as query-based span prediction*. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 6953–6963, 2020.

[9] A. Jain, B. Paranjape, and Z. C. Lipton. *Entity Projection via Machine Translation for Cross-Lingual NER*. In K. Inui, J. Jiang, V. Ng, and X. Wan, editors, Proceedings of the 2019 Conference on Empirical Methods in

Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pages 1083–1092, Hong Kong, China, November 2019. Association for Computational Linguistics. Available from: https://aclanthology.org/D19-1100, doi:10.18653/v1/D19-1100.

[10] R. Shah, B. Lin, A. Gershman, and R. Frederking. *SYNERGY: a named entity recognition system for resource-scarce languages such as Swahili using online machine translation*. In Proceedings of the Second Workshop on African Language Technology (AfLaT 2010), pages 21–26, 2010.

[11] M. Recasens, L. Màrquez, E. Sapena, M. A. Martí, M. Taulé, V. Hoste, M. Poesio, and Y. Versley. *Semeval-2010 task 1: Coreference resolution in multiple languages*. In Proceedings of the 5th International Workshop on Semantic Evaluation, pages 1–8, 2010.

[12] S. Pradhan, A. Moschitti, N. Xue, O. Uryupina, and Y. Zhang. *CoNLL-2012 shared task: Modeling multilingual unrestricted coreference in OntoNotes*. In Joint Conference on EMNLP and CoNLL-Shared Task, pages 1–40, 2012.

[13] C. Dyer, V. Chahuneau, and N. A. Smith. *A simple, fast, and effective reparameterization of ibm model 2*. In Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 644–648, 2013.

[14] H. Kobdani and H. Schütze. *Sucre: A modular system for coreference resolution*. In Proceedings of the 5th International Workshop on Semantic Evaluation, pages 92–95. Association for Computational Linguistics, 2010.

[15] G. Attardi, M. Simi, and S. Dei Rossi. *TANL-1: coreference resolution by parse analysis and similarity clustering*. In Proceedings of the 5th International Workshop on Semantic Evaluation, pages 108–111, 2010.

[16] A. van Cranenburgh. *A Dutch coreference resolution system with an evaluation on literary fiction*. Computational Linguistics in the Netherlands Journal, 9:27–54, 2019.

[17] A. Rahman and V. Ng. *Translation-based projection for multilingual coreference resolution*. In Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 720–730, 2012.

[18] P. Denis and J. Baldridge. *Joint determination of anaphoricity and coreference resolution using integer programming*. In Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference, pages 236–243, 2007.

[19] P. Koehn. *Europarl: A parallel corpus for statistical machine translation*. In MT summit, volume 5, pages 79–86. Citeseer, 2005.

[20] D. Yarowsky, G. Ngai, and R. Wicentowski. *Inducing multilingual text analysis tools via robust projection across aligned corpora*. Technical report, Johns Hopkins University Baltimore MD Department of Computer Science, 2001.

[21] R. Hwa, P. Resnik, A. Weinberg, C. Cabezas, and O. Kolak. *Bootstrapping parsers via syntactic projection across parallel texts*. Natural language engineering, 11(3):311–326, 2005.

[22] O. Postolache, D. Cristea, and C. Orasan. *Transferring Coreference Chains through Word Alignment*. In LREC, pages 889–892, 2006.

[23] D. Zeman and P. Resnik. *Cross-language parser adaptation between related languages*. In Proceedings of the IJCNLP-08 Workshop on NLP for Less Privileged Languages, 2008.

[24] M. Ehrmann, M. Turchi, and R. Steinberger. *Building a multilingual named entity-annotated corpus using annotation projection*. In Proceedings of the International Conference Recent Advances in Natural Language Processing 2011, pages 118–124, 2011.

[25] J. G. C. de Souza and C. Orăsan. *Can projected chains in parallel corpora help coreference resolution?* In Discourse Anaphora and Anaphor Resolution Colloquium, pages 59–69. Springer, 2011.

[26] R. Fu, B. Qin, and T. Liu. *Generating Chinese named entity data from parallel corpora*. Frontiers of Computer Science, 8(4):629–641, 2014.

[27] J. Ni, G. Dinu, and R. Florian. *Weakly supervised cross-lingual named entity recognition via effective annotation and representation projection*. arXiv preprint arXiv:1707.02483, 2017.

[28] Y. Grishina. *Assessing the applicability of annotation projection methods for coreference relations*. PhD thesis, Universität Potsdam, 2019.

[29] S. Mayhew, C.-T. Tsai, and D. Roth. *Cheap translation for cross-lingual named entity recognition*. In Proceedings of the 2017 conference on

empirical methods in natural language processing, pages 2536–2545, 2017.

[30] J. Xie, Z. Yang, G. Neubig, N. A. Smith, and J. Carbonell. *Neural Cross-Lingual Named Entity Recognition with Minimal Resources*. In E. Riloff, D. Chiang, J. Hockenmaier, and J. Tsujii, editors, Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pages 369–379, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. Available from: https://aclanthology.org/D18-1034, doi:10.18653/v1/D18-1034.

[31] J. Tiedemann, Ž. Agić, and J. Nivre. *Treebank translation for cross-lingual parser induction*. In Eighteenth Conference on Computational Natural Language Learning (CoNLL 2014), 2014.

[32] P. F. Brown, S. A. Della Pietra, V. J. Della Pietra, and R. L. Mercer. *The mathematics of statistical machine translation: Parameter estimation*. Computational linguistics, 19(2):263–311, 1993.

[33] F. J. Och and H. Ney. *Improved statistical alignment models*. In Proceedings of the 38th annual meeting of the association for computational linguistics, pages 440–447, 2000.

[34] J. Nothman, N. Ringland, W. Radford, T. Murphy, and J. R. Curran. *Learning multilingual named entity recognition from Wikipedia*. Artificial Intelligence, 194:151–175, 2013.

[35] R. Al-Rfou, V. Kulkarni, B. Perozzi, and S. Skiena. *Polyglot-NER: Massive multilingual named entity recognition*. In Proceedings of the 2015 SIAM International Conference on Data Mining, pages 586–594. SIAM, 2015.

6

Conclusions and Future Research

This chapter outlines the main conclusions for each of the chapters described in this thesis. Specifically, we describe the main contributions for every presented model and we conclude by presenting future research directions that alleviate some of the limitations of the proposed models.

⋆⋆⋆

6.1 Conclusions

6.1.1 Adapting Language Models to Distractor Ranking for Educational Multiple-Choice Questions

In Chapter 2 of this thesis, we introduced and evaluated context-aware distractor retrieval models. These models were developed by adapting multilingual pretrained language models to reuse distractor candidates that can facilitate the educational task of MCQs creation. Particularly, we proposed three models: (1) The D-SIM model that learns similar contextual representations for similar distractors, (2) The Q-SIM model that requires similar questions to have similar representations, and (3) The DQ-SIM model that linearly combines the previous two models benefiting from their respective

strengths. Importantly, the DQ-SIM model showed a considerably reduced nonsense distractor rate, which we consider a useful asset in terms of trust in the model by teachers. We also asked teachers to evaluate the quality of distractors using a four-level annotation scheme that we introduced. As a result, teachers considered 3 out of 10 suggested distractors as high-quality, to be readily used. Additionally, they found two more distractors to be within topic, albeit of lower quality, and useful as inspiration for teachers to come up with their own good distractors. Finally, we released a test consisting of 298 educational MCQs with annotated distractors covering six subjects and a 77K distractor vocabulary to promote further research.

6.1.2 Leveraging Large Language Models for Distractor Generation

In Chapter 3, we proposed a novel strategy to guide an instruction-tuned large language model for the task of distractor generation in educational multiple-choice questions. We direct LLMs to generate plausible and effective distractors by prompting them with well-chosen in-context example question items. These items are automatically retrieved by the Q-SIM ranker introduced in Chapter 2. We combine the original question with these chosen items to give the LLM a prompt for creating distractors. We show a significant improvement over LLMs that use random in-context examples and the methods in Chapter 2. Teachers rated 5 out of 10 of our distractors as high-quality on average, better than the 3 out of 10 for earlier methods. Also, the production of nonsensical distractors dropped to 16%, a significant decrease from those produced by the ranking models in Chapter 2.

6.1.3 Adapting Language Models to Gap-filling Exercise Generation for Language Learning

In Chapter 4, our focus shifted to tailoring a pre-trained language Model (PLM) for a specific educational task: generating gap-fill exercises in French. This task was more specialized compared to the generic task of distractor generation across various domains, subjects, and languages that we explored in Chapters 2 and 3. We introduced a real-world dataset called GF2, which consists of French gap-filling exercises and a related prediction task. This task involves identifying potential gaps in a given text, without prior knowledge of the specific type of exercise, relying solely on an example exercise. We proposed an example-aware neural network model specifically designed for this task, and compared it with a baseline model that does not take into account any example of the desired exercise type. We found that our example-aware model outperforms the baseline model not only in

predicting gaps, but also in distinguishing between different elementary exercise types, even though it was not explicitly trained for this secondary task.

6.1.4 Adapting Coreference Resolution to new target languages

In Chapter 5, we extended the adaptability theme of the thesis and focused on adapting a fundamental task of coreference resolution to new languages. Our approach involved leveraging black-box translation tools for bootstrapping coreference resolution in languages with limited resources. We compared and analysed two conceptually different strategies; (i) `Translate-train` - translate the training data in high-resource language to the target language and train a coreference model and (ii) `Translate-test` - translate the test data into high-resource source language (e.g., English) and use a trained coreference model for inference. Moreover, we studied the source of errors for these two strategies and found that the quality of machine translation tools is the main limiting factor.

6.2 Future Directions

Although the methods and techniques we proposed have shown considerable performance improvements in their respective tasks, there are several promising research directions that emerge from our contributions. In the following paragraphs, we will outline some potential future directions for enhancing the models and methods discussed in this thesis.

Distractor generation: In Chapters 2 and 3, we presented our proposed models and techniques to generate textual distractors for educational MCQs. Yet, other avenues seem promising. One future research direction is expanding the current work into a multimodal system. This system would consider additional sources of information, such as images and speech data accompanying MCQs in digital learning tools. This approach could provide a more engaging and interactive way for students to interact with educational content, enhancing their understanding and retention of concepts. Moreover, exploring the educational impact of multimodal distractor generation can inform the design of future educational MCQs and assessments. For example, the effectiveness of image-text questions in various subject areas, like science and history could be evaluated to identify the best practices for multimodal MCQ generation.

Additionally, there is potential to explore alternative prompting strategies for large language models (LLMs) in distractor generation. One such

strategy is the use of a self-correcting mechanism [1], which involves revising the initial output of an LLM by evaluating specific aspects of the text. This approach could offer new insights in the context of distractor generation.

Another practical research direction is to investigate the design of a more detailed evaluation framework for distractor quality. This framework would consider various factors, including the student's level and the difficulty of questions. For instance, to incorporate the student's level, one might look at their performance history or academic standing. Similarly, the difficulty of a question can be estimated based on the percentage of times students answer it correctly versus incorrectly. Another area to explore is ensuring that a single MCQ's complete set of distractors are sufficiently diverse. In our present study, we focused on retrieving (or generating) a list of plausible distractors independently of each other. However, ideally, distractors in MCQs should be not only plausible but also diverse. The evaluation frameworks should also be made to account for the diversity of distractors.

Gap-filling exercise generation: First, while our proposed method in Chapter 4 is language-agnostic in principle, our evaluation is limited to our French benchmark dataset. Expanding our approach to encompass other languages would bring new and interesting challenges for further investigation. Second, despite topic diversity within our exercise documents (e.g., some examples consist of independent sentences, while others are coherent texts centered around the same topic.), it would be interesting to quantify the degree of topical bias introduced during our training process and its impact on our binary task evaluation. Another natural modeling extension is to adapt Sequence to sequence (SEQ2SEQ) models for our task, particularly text-to-text models such as T5 [2]. As for the distractor generation task, there is a potential to explore different prompting strategies for large language models (LLMs), when generating gap-filling grammar exercises. For instance, using chain-of-thought prompting [3], which involves generating intermediate steps before producing the final response, could be explored for generating grammar exercises. Additionally, an interesting future study would involve investigating the number of example demonstrations that LLMs require in order to accurately mimic example gap exercises.

Coreference resolution: In Chapter 5, our `Translate-test` involves using a coreference model trained on original (human-generated) data. However, the input fed into the model during testing is produced by a machine translation tool. Recent studies [4] have shown that original and machine-translated data possess different properties, and this mismatch can harm

performance. An intriguing research direction is to adapt machine translation (MT) tools to mitigate this mismatch in the context of coreference resolution. Although recent research [5] has proposed addressing the distribution shift between machine-translated and human-generated text, this has so far only been explored for classification tasks. Investigating similar adaptations for coreference resolution could be a valuable extension of this work.

Ethical and practical considerations: PLMs bring substantial advantages but also present significant ethical and practical challenges. This thesis has not directly addressed key concerns like data privacy, biases inherent in the training datasets for PLMs, and the widening digital divide. It is crucial to complement the deployment of these technologies with rigorous critical evaluation and human supervision to ensure their positive impact within educational settings. Furthermore, this work does not explore energy usage or evaluate the carbon footprint associated with employing PLMs in educational contexts. Future research involving PLMs should aim to assess and report on aspects like fairness, the digital footprint, and environmental impact, specifically carbon emissions.

References

[1] R. Wang, H. Wang, F. Mi, Y. Chen, R. Xu, and K.-F. Wong. *Self-Critique Prompting with Large Language Models for Inductive Instructions*. arXiv preprint arXiv:2305.13733, 2023.

[2] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. *Exploring the limits of transfer learning with a unified text-to-text transformer*. The Journal of Machine Learning Research, 21(1):5485–5551, 2020.

[3] J. Wei, X. Wang, D. Schuurmans, M. Bosma, brian ichter, F. Xia, E. H. Chi, Q. V. Le, and D. Zhou. *Chain of Thought Prompting Elicits Reasoning in Large Language Models*. In A. H. Oh, A. Agarwal, D. Belgrave, and K. Cho, editors, Advances in Neural Information Processing Systems, 2022. Available from: https://openreview.net/forum?id=_VjQlMeSB_J.

[4] M. Artetxe, G. Labaka, and E. Agirre. *Translation Artifacts in Cross-lingual Transfer Learning*. In B. Webber, T. Cohn, Y. He, and Y. Liu, editors, Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 7674–7684, Online, November 2020. Association for Computational Linguistics. Available from: https://aclanthology.org/2020.emnlp-main.618, doi:10.18653/v1/2020.emnlp-main.618.

[5] M. Artetxe, V. Goswami, S. Bhosale, A. Fan, and L. Zettlemoyer. *Revisiting Machine Translation for Cross-lingual Classification*. In H. Bouamor, J. Pino, and K. Bali, editors, Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, pages 6489–6499, Singapore, December 2023. Association for Computational Linguistics. Available from: https://aclanthology.org/2023.emnlp-main.399, doi:10.18653/v1/2023.emnlp-main.399.

A

Predicting Suicide Risk from Online Postings in Reddit: The UGent-IDLab submission to the CLPysch 2019 Shared Task A

In this chapter we describe our contribution to CLPsych 2019 shared task where we achieve competitive results using linear models and ensemble models to predict the degree of suicide risk of people based on their posts on Reddit.

⋆ ⋆ ⋆

S.K. Bitew, G. Bekoulis, J. Deleu, L. Sterckx, K. Zaporojets, T. Demeester and C. Develder

In Proceedings of CLPsych, 2019.

Abstract This paper describes IDLab's text classification systems submitted to Task A as part of the CLPsych 2019 shared task. The aim of this shared task was to develop automated systems that predict the degree of suicide risk of people based on their posts on Reddit.[1] Bag-of-words features,

[1] www.reddit.com

emotion features and post-level predictions are used to derive user-level predictions. Linear models and ensembles of these models are used to predict final scores. We find that predicting fine-grained risk levels is much more difficult than flagging potentially at-risk users. Furthermore, we do not find clear added value from building richer ensembles compared to simple baselines, given the available training data and the nature of the prediction task.

A.1 Introduction

The goal of the CLPysch 2019 shared task is to predict the degree of suicide risk based on online postings of users. This shared task is motivated by the long-term lack of progress in predicting suicide risk. [1], after reviewing more than 70 studies, argues that suicidality cannot be predicted effectively using traditional standard procedures, e.g., questions of clinicians about suicidal thoughts: the authors claim that a large fraction of patients (i.e., 80%) who committed suicide, did not admit contemplating suicide when asked by a general practitioner. Another study by [2] also concludes that prediction of suicide risks has not improved over the last 50 years and suggests that machine learning learning methods can contribute towards solving that challenge.

Typically, there are long periods of time between clinical encounters of patients. During these periods, some patients are engaged in frequent use of social media. [3] states that such usage of social media can be exploited to build binary risk classifiers. However, when such systems are deployed, the number of people flagged as “at risk" will exceed clinical capacity for intervention. This in turn motivates the design of more fine-grained prediction models, predicting various risk levels, as proposed for the current shared task.

Our system uses a combination of (i) bag-of-word features, (ii) emotion labels, and (iii) information derived from post-level risk features (see Section A.3.1 for more details). Using these features, we apply linear models to predict the scores. We explore different combinations to evaluate the performance of the different models.

The remainder of the paper is organized as follows: Section A.2 describes the data and the shared task. Section A.3 presents the details of the implemented system and the features. Section A.4 shows the experimental results obtained from the test data. To compare our results to other participants in the shared task, we refer the reader to [4]. To conclude, we summarize our findings and present future directions in Section A.5.

A.2 Data and Task A

The dataset used in the shared task is sampled from the University of Maryland Reddit Suicidality Dataset [5]. It is constructed using data from Reddit, an online site for anonymous discussion on a wide variety of topics. Specifically, the UMD dataset was extracted from the 2015 Full Reddit Submission Corpus[2],using postings in the r/SuicideWatch subreddit (henceforth simply SuicideWatch or SW) to identify anonymous users who might represent positive instances of suicidality and including a comparable number of non-SuicideWatch controls. The dataset is annotated at user level, using a four-point scale indicating the likelihood of a user to commit suicide: (a) no risk, (b) low risk, (c) moderate risk, and (d) severe risk. The corpus includes posts from 21,518 users and is subdivided into 993 labelled users and 20,525 unlabelled users. Out of the 993 labeled users, 496 have at least posted once on the SuicideWatch subreddit. The remaining 497 users are control users (i.e., they have not posted in SuicideWatch or any mental health related subreddits). The data is provided in a comma-separated values file that includes the post titles, content, timestamps, and anonymized unique user ids. The goal of shared Task A is to predict users' suicide risk into one of the four classes (i.e., (a)-(d)) given the fact that he/she has posted on SuicideWatch.

A.3 Systems Description

This section provides an overview of features extracted from posts, followed by a short system description of our submitted runs.

A.3.1 Features

TF-IDF features: We used the TF-IDF weighting scheme as text representation. The TF-IDF feature vectors of *n*-grams were generated for our dataset. We experimented with *n*-grams for *n* ranging from 1 to 5. In our preliminary investigations, we explored various kinds of features, such as character level *n*-grams, or textual statistical features (such as the total number of posts), but these did not lead to increased performance metrics.

Emotion features: We hypothesize that individuals contemplating suicide will tend to express emotions with negative sentiment, more than individuals without suicidal thoughts. Therefore, we use a pre-trained model

[2] https://www.reddit.com/r/datasets/comments/3mg812/full_reddit_submission_corpus_now_available_2006/

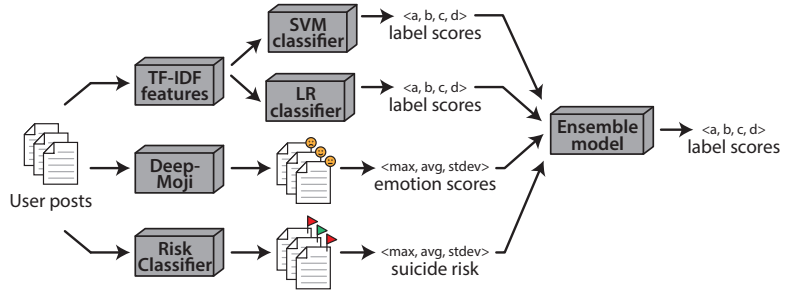


Figure A.1: Main elements of the presented system setup.

called *DeepMoji*[3] that predicts emotions from text [6]. For an individual post of a user, a 64-dimensional emotion feature vector is generated by the model, with each dimension corresponding to the probability for one out of 64 different emojis. We take the element-wise maximum, average and standard deviation of this vector as features to represent a user's emotions.

Suicide risk features: We reason that post-level binary risk estimates can help in making the user-level risk level prediction. To achieve this, we semi-manually annotated 605 posts from the unlabelled dataset as follows. First, we trained a TF-IDF based logistic regression classifier to predict the four class labels (a)–(d), using labelled data for 496 users. We adopt that classifier to assign four probabilities, one for each class (a)–(d), to each post in the unlabelled dataset. We take a random sub-sample of the automatically labelled posts, order it in terms of no-risk probability, and manually label posts taken in turn from the top and bottom of the ordered list. We thus obtain a balanced set of 605 annotated posts (302 'risk', 303 'no-risk'), spending a total annotation time of 5 hours. Subsequently, a TF-IDF based logistic regression binary classifier was trained on these manually annotated posts. Finally, the post-level binary predictions were then aggregated into user-level suicide risk features by taking the maximum, mean, and standard deviation of the predicted post-level scores. The motivation behind this annotation experiment was to investigate the effectiveness of a cheap additional annotation effort in boosting the final model's prediction accuracy. By 'cheap' annotation effort, we refer to annotations on the *post-level* as opposed to user-level, *binary* as opposed to 4-label, and *directly balanced* as opposed to a larger random sample to obtain the same amount of at-risk posts.

[3] https://github.com/bfelbo/DeepMoji

A.3.2 Models

Three different systems were explored for our submission to the shared task. A logistic regression classifier and two ensemble-based classifiers.

1. **Baseline classifier**: a LR classifier [7] is trained based on TF-IDF weighted bag-of-word features.

2. **Ensemble without Risk classifier**: this ensemble combines the scores from the baseline logistic regression classifier, a linear Support Vector Machine (SVM) classifier and the emotion classifier. The linear SVM, included in scikit-learn [7] is trained on the TF-IDF representations. This ensemble uses an additional logistic regression classifier (at the next level) to predict the final classes.

3. **Ensemble (all)**: this model combines the scores from all classifiers as illustrated in Fig. A.1. This ensemble uses a second level Logistic Regression classifier similar to the previous ensemble.

With this system choice, we are able to measure the impact of combining linear classifiers with emotion features compared to a simple linear model (second vs. first run), and to measure the added value from the additional post-level annotations (third vs. second run).

A.4 Experimental Results

In this section, we present the final test results of the three submitted systems on the official test set. The test set consists of a total of 189 posts from 125 different users. The official evaluation metric used in the shared task is the macro F_1 score on all four classes. Table A.1 depicts the official models' performance on the test data. Our baseline classifier outperforms the ensemble models. This can be explained by (i) bias in the training/test split during development, (ii) the small number of annotated training instances, or (iii) the partly subjective nature of the task, and in particular the distinction between fine-grained levels such as 'low risk' and 'moderate risk'. Note that, however, our most advanced model did perform best for the simpler task of detecting potentially at-risk ('flagged') users. Further research is required to investigate these potential issues.

In addition, two more metrics were used. The first metric is the F_1 score for *flagged versus non-flagged* users. The flagged vs. non-flagged F_1 is relevant for a use case in which the goal is to distinguish users that can be safely ignored (category (a), no risk) from those that require attention (i.e., categories (b), (c), (d)), such as when human moderators need to investigate

Table A.1: Official results

Models	Precision	Recall	F_1
Baseline	0.444	0.457	**0.445**
Ensemble w/o Risk	0.428	0.402	0.407
Ensemble (all)	0.445	0.419	0.426

the risk further. Table A.2 shows the performance of the models in binary classification of flagged and non-flagged users, whereby the ensemble with sentiment features ('Ensemble w/o Risk) outperforms the linear baseline, but the overall ensemble with binary post-level risk predictions performs slightly better still. Given the much higher scores, the task of flagging potentially at-risk users appears much simpler than making fine-grained risk-level predictions.

Table A.2: Flagged vs Non-flagged

Models	Precision	Recall	F_1
Baseline	0.904	0.806	0.852
Ensemble w/o Risk	0.848	0.903	0.875
Ensemble (all)	0.850	0.914	**0.881**

The second metric is the *urgent versus non-urgent* F_1 score that measures distinction between users who are at a severe risk of suicide (category (c) and (d)) and other users. Table A.3 shows the models' performance for classifying users into urgent and non-urgent classes. The overall higher scores in Table A.3 indicate that the binary classification of urgent from non urgent users is fairly simpler task when compared to the fine-grained risk level classification.

Table A.3: Urgent vs Non-urgent

Models	Precision	Recall	F_1
Baseline	0.833	0.750	**0.789**
Ensemble w/o Risk	0.795	0.725	0.758
Ensemble (all)	0.792	0.762	0.777

A.5 Conclusion and Future work

In this paper, we described the Ghent University-IDLab submission to the CLPysch 2019 shared Task A. We found that the baseline classifier based on

logistic regression outperformed the ensemble of classifiers. Specifically, our baseline model obtained a macro F_1-score of 0.445 on the shared task. Our system also achieves a macro F_1-score of 0.881 and 0.789 on flagging non-risk users and distinguishing urgent from non-urgent users, respectively. The more advanced models (i.e., ensembles) did not bring any added value in the fine-grained user level risk prediction. This can be due to the limited number of training examples in the provided dataset, bias in train/test splits during development and the subjective nature of the task.

As next steps, we plan on investigating alternative ways of splitting train from test data such as stratified cross-validation (i.e., to avoid different distributions of the target variable in the train/test splits). We also want to explore more sophisticated ways of ensembling and stacking techniques while also taking into account the time stamp meta-data of posts.

Acknowledgments

We would like to thank the CLPsych 2019 shared task organizers for organizing the competition and providing us with the online postings of users data from Reddit.

Ethical Review

To meet the ethical review criteria as discussed in the [4] overview paper, this study was evaluated by the Ethics Committee of the faculty of Psychology and Educational Sciences of Ghent University. The committee concluded that ethical approval was not needed for conducting the research.

References

[1] C. M. McHugh, A. Corderoy, C. J. Ryan, I. B. Hickie, and M. M. Large. *Association between suicidal ideation and suicide: meta-analyses of odds ratios, sensitivity, specificity and positive predictive value*. BJPsych open, 5(2), 2019.

[2] J. C. Franklin, J. D. Ribeiro, K. R. Fox, K. H. Bentley, E. M. Kleiman, X. Huang, K. M. Musacchio, A. C. Jaroszewski, B. P. Chang, and M. K. Nock. *Risk factors for suicidal thoughts and behaviors: a meta-analysis of 50 years of research.* Psychological Bulletin, 143(2):187, 2017.

[3] G. Coppersmith, C. Hilland, O. Frieder, and R. Leary. *Scalable mental health analysis in the clinical whitespace via natural language processing*.

In 2017 IEEE EMBS International Conference on Biomedical & Health Informatics (BHI), pages 393–396. IEEE, 2017.

[4] A. Zirikly, P. Resnik, Ö. Uzuner, and K. Hollingshead. *CLPsych 2019 Shared Task: Predicting the Degree of Suicide Risk in Reddit Posts*. In Proceedings of the Sixth Workshop on Computational Linguistics and Clinical Psychology: From Keyboard to Clinic, 2019.

[5] H.-C. Shing, S. Nair, A. Zirikly, M. Friedenberg, H. Daumé III, and P. Resnik. *Expert, crowdsourced, and machine assessment of suicide risk via online postings*. In Proceedings of the Fifth Workshop on Computational Linguistics and Clinical Psychology: From Keyboard to Clinic, pages 25–36, 2018.

[6] B. Felbo, A. Mislove, A. Søgaard, I. Rahwan, and S. Lehmann. *Using millions of emoji occurrences to learn any-domain representations for detecting sentiment, emotion and sarcasm*. In Conference on Empirical Methods in Natural Language Processing (EMNLP), 2017.

[7] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. *Scikit-learn: Machine Learning in Python*. Journal of Machine Learning Research, 12:2825–2830, 2011.