Automatic generation of learning resources: A case study on the generation of true/false sentences.

EDUARDO MOREIRA MIRANDA, Instituto Superior Técnico, Portugal

In recent years, the education landscape has undergone a transformative shift towards massification, characterized by the widespread availability of educational resources facilitated by technological advancements. The surge of Massive Open Online Courses (MOOCs) has underscored the need for a continuous supply of learning resources, leading to a burgeoning research focus on automatic question generation. Our primary objective is to address this need by automating the generation of questions related to data charts, with a specific emphasis on crafting true or false data science questions. We explore various models designed for visual question generation, categorizing them based on specific dimensions. In response to this exploration, we propose the development of a dataset comprising triples (Image-question-answer) and the introduction of two models: one dedicated to question generation and another for answering those questions. Model evaluation were conducted using machine translation metrics for the question generation aspect and standard evaluation metrics for binary classification tasks in the context of question answering.

Additional Key Words and Phrases: Visual Question Generation, Visual Question Answering, MOOCs, Transformers, LLM, Learning Resources

1 Introduction

Education has undergone a significant transformation with the rise of Massive Open Online Courses (MOOCs) and other digital learning platforms. This shift has created a growing demand for scalable and personalized learning resources. However, as class sizes grow, the ability to provide individualized attention diminishes, making automated systems essential for generating tailored educational content.

MOOCs, in particular, highlight the need for personalization since it allows learners to progress at their own pace [18] and in alignment with their unique learning styles, especially through automated question generation that addresses specific learner needs. This demand is especially pronounced in data-driven fields like data science, where data charts are a common educational tool.

We address the challenge of automating the generation of true/false questions from visual data representations, particularly data charts. We propose a framework that automates the creation of true/false questions by using both visual and language models. Our work contributes to the field of Visual Question Generation (VQG), with a specific focus on applications in data science education.

We use pretrained image-to-text models such as Pix2Struct and GiT, fine-tuning them on diverse types of data charts (e.g., boxplots, histograms, decision trees). These models extract relevant information from charts, which is then inputted into a generative pretrained model like GPT-3.5 to create true/false questions. The same framework is used for answering the questions.

We achieved excellent results in extracting visual information, with scores of 0.9 and 0.85 on the BLEU and ROUGE metrics, respectively. Additionally, we received an average score of 4.4 out of 5 in a manual evaluation conducted by a domain specialist. For the question answering task, we achieved scores of 0.89 and 0.95 on the BLEU and ROUGE metrics for the visual task and an accuracy of 0.85 in answering the questions.

This document is structured as follows: Section 2 provides a review of machine learning concepts important for understanding this work and other studies in the field of Visual Question Generation (VQG). We also present our categorization of different works in VQG and introduce the models used in the training process for our own work. Section 3 defines the problem and explains how we addressed the tasks of question generation and question answering. In Section 4, we detail the creation of a dataset consisting of image-question-answer triples, which was used to train the generative models, along with the auxiliary datasets that were essential for training various components of the architecture. Section 5 describes the global training process used for training several models, and outlines the step-by-step approach to dividing and analyzing the question generation task, including both automated and manual evaluations by a domain expert. Section 6 explains the step-by-step process of dividing and analyzing the question answering task, followed by an evaluation where we compare the results for each chart type. Finally, Section 7 concludes with a summary of our work, offering insights for future research and final considerations on the developed solutions.

2 Literature Review

The generation of educational resources has seen growing interest, particularly in the field of Automatic Question Generation (AQG). AQG systems have traditionally been focused on text-based question generation using various approaches such as rule-based methods, template-based approaches, and more recently, neural network-based models. However, generating questions from visual data is a newer challenge, leading to the emergence of Visual Question Generation (VQG).

2.1 Basic Concepts

A strong foundation in deep learning models is essential for understanding VQG systems, especially when they involve Convolutional Neural Networks (CNNs), the Transformer architecture, and Large Language Models (LLMs).

2.1.1 Convolutional Neural Networks (CNN). Convolutional Neural Networks [13], often abbreviated as CNNs, represent a fundamental component of deep learning, particularly in the domain of computer vision and image analysis. CNNs are a class of artificial neural networks designed to process and analyze visual data, making them highly relevant to a wide range of applications, from image recognition to medical imaging and autonomous vehicles.

Author's Contact Information: Eduardo Moreira Miranda, Instituto Superior Técnico, Portugal, eduardommiranda@tecnico.ulisboa.pt.

2.1.2 The Transformer Architecture. The transformer architecture is a type of neural network architecture introduced in the paper "Attention is All You Need" by Vaswani et al.[21] The transformer architecture has become a fundamental building block for various natural language processing (NLP) tasks, such as machine translation, text summarization, and question answering.

The invention of the attention mechanism solved the problem of how to encode a context into a word, or in other words, how you can present a word and its context together in a numerical vector. The transformer takes this concept a step further, constructing a neural network for natural language translation without recurrent structures. This simplifies the network, making it more trainable, parallelizable, and capable of accommodating complex language models.

2.1.3 Large Language Models. Large Language Models (LLMs) are advanced natural language processing models that are characterized by their extensive scale, capacity, and ability to understand and generate human-like text. These models are trained on vast amounts of textual data and use sophisticated architectures to capture and generate language patterns.

They often use transformer architectures as a foundational component of their design. The transformer architecture, as mentioned before, has become a standard for processing sequential data, especially in natural language processing tasks.

Large Language Models like GPT-3 (Generative Pre-trained Transformer 3) [3] is one example of LLMs that is built on top of a transformer architecture.

2.2 Visual Question Generation

The task of Visual Question Generation (VQG) sits at the intersection of computer vision and natural language processing. Early efforts in VQG focused on generating questions from images of objects and scenes, leveraging CNNs for image understanding and RNNs for question generation. These models were mostly designed to ask simple, fact-based questions such as "What is the color of the object?" or "Where is the cat?" based on visual inputs.

In our work, we categorize VQG models into four dimensions inspired by the categorization used in the review by Patil et al.[15]:

- Question Generation Method: We analyze the methods employed for generating questions, ranging from traditional rule-based and template-based approaches to novel question generation techniques. Each method carries distinct advantages and limitations, and understanding their nuances is paramount for comprehending the evolution of VQG algorithms.
- Learning Method: The choice of learning methods plays a pivotal role in the development of VQG models. We explore the spectrum from simple deep neural networks to more advanced generative models and reinforcement learning networks.
- Input Modality: The information fed into VQG models significantly influences their performance. We investigate different input modalities, including image-only, image with contextual information, and image paired with answer data.
- Type of Generated Questions: VQG algorithms produce an array of question types, spanning from questions grounded entirely in visual content to those that draw upon common sense or external knowledge.

In our work we analyzed and categorized the following works in the field of Visual Question Generation: CRIC [5], GQA [6], COCO QA [17], Multimodal differential network [16], K-VQG [19], VAE approach [7], iQAN [9], GRNN [12], CLIP approach [22], Discriminators approach [4].

Given the limitations of earlier VQG approaches, our work leverages the latest advancements in transformer-based architectures and pretrained models. Tools like Pix2Struct and GiT excel at extracting textual information from structured data visualizations like charts, while LLMs such as GPT-3.5 generate syntactically and semantically accurate questions based on the extracted information.

2.3 Models used on the training process

To achieve accurate true/false question generation and answering from data charts, we employed a combination of pretrained models for both visual data extraction and natural language processing. These models were chosen for their ability to handle complex image-to-text and text-generation tasks.

2.3.1 Visual Models.

- **Pix2Struct**: Pix2Struct is a transformer-based model designed for visual understanding and optical character recognition (OCR). It was fine-tuned on various types of data charts, including box plots, histograms, and decision trees, to extract relevant textual information, such as labels and trends, from the images. This visual data extraction is critical for generating meaningful questions from charts.
- **GiT**: GiT is another image-to-text model known for its high performance in structured data recognition. Like Pix2Struct, it was trained to capture key details from visual inputs, allowing for precise chart data extraction.

2.3.2 Language Models.

- **GPT-3.5 Turbo**: GPT-3.5, a large language model, was fine-tuned to generate questions and answer them in the context of data science. Leveraging its vast knowledge of natural language and data-related tasks, GPT-3.5 processes the textual information extracted by visual models and generates grammatically correct and contextually relevant true/false questions.
- **Mistral 7B**: This smaller but highly efficient model complements GPT-3.5 by handling specific fine-tuning tasks related to true/false classification. It was also trained in the data science domain, contributing to accurate question answering.

These models work together to form a pipeline where Pix2Struct and GiT extract textual information from data charts, which is then formatted into prompts for GPT-3.5 and Mistral 7B to generate and answer questions. The fine-tuning process ensured the models' output was highly specialized for the domain of data science.

3 Problem Statement

The growing adoption of MOOCs and other educational platforms has highlighted the need for scalable tools that can generate meaningful, personalized learning resources. One critical area where this can be applied is the automatic generation of true/false questions from data charts, which are a fundamental component of data science education. However, generating such questions manually is time-consuming and not scalable, requiring the development of automated systems to assist educators and enhance student engagement.

The problem we aim to solve is how to automatically generate and answer true/false questions based on data charts, thus providing personalized learning experiences at scale. To achieve this, we propose a methodology that consists of three phases:

- The first phase involves generating the data that will be used to train the model.
- The second phase focuses on training the model to generate questions using that data.
- The third phase entails training a model to answer the questions given an image-question pair.

With this methodology we developed two models. The question generation model takes an image as input and outputs a question, while the question answering model takes both an image and a question as input and outputs an answer. Figure 1 demonstrates the input and output of each model during inference time.

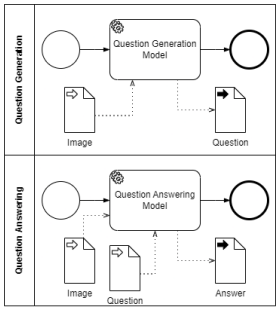

Fig. 1. BPMN diagram showing each model at inference time.

In the following sections, we will address each one of these phases, describing the approach used and evaluating their results.

4 Data Generation

The first phase requires a dataset consisting of Image-Question-Answer triples. This is necessary because we want the question generation model to produce questions that are non-trivial and require knowledge about data science. Therefore, we need a dataset with questions of this nature to train the model effectively. If we only aimed to develop a question generation model, a dataset of Image-Question pairs would suffice. However, since we also aim to develop a question answering model, our dataset must include the answers as well. This allows us to train the question answering model to respond accurately to these data science-related questions.

Since, to the best of our knowledge, there is currently no existing dataset of data science sentences composed of triples (Image-Question-Answer), we must generate it.

4.1 Data Supporting the Questions

To support the task of generating and answering true/false questions from data charts, we developed a dataset comprising Image-Question-Answer triples. This dataset was specifically tailored to train models capable of handling data visualizations, given that no existing dataset met these requirements. The data generation process consisted of several key phases:

4.2 Main Dataset

We constructed a comprehensive dataset using 34 datasets sourced from repositories such as UCI and Kaggle. This process involved generating 459 unique plots, representing a variety of data visualizations, including:

- Bar charts displaying record and variable counts.
- Histograms and boxplots for numeric variables.
- Correlation heatmaps and decision tree diagrams.

For each dataset, we employed a template-based approach, filling predefined templates extracted from data science exams, with data related terms to generate corresponding questions. Table 2 illustrates three templates and how they are filled.

Template	Space1	Space2	Space3
The [1] for the presented tree is [2] than its [3].	[accuracy,recall, precision,specificity]	[higher,lower]	[accuracy,recall, precision,specificity]
The variable [1] can be coded as ordinal without losing information.	[<all_variables>]		
Variable [1] presents some outliers.	[<numeric_variables>]		

Fig. 2. Three examples of templates.

The generated questions were paired with their corresponding plots, creating a dataset of 130,352 Image-Question pairs. To avoid overfitting and redundancy, the dataset was pruned to 2,630 Image-Question-Answer triples, ensuring a balanced representation across chart types.

4.3 Plot Generation

To create the visualizations required for the dataset, we used Python [20], Pandas [11] and Matplotlib [1]. For each of the 34 datasets, we generated a variety of chart types, ensuring a broad representation of visual data. The types of plots generated included:

- Bar charts displaying record and variable counts.
- Correlation heatmaps showing relationships between numeric variables.
- Histograms and boxplots for numeric data distribution.
- Decision tree diagrams for simple classification tasks.
- Multi-line charts showing model accuracies, such as decision trees, KNN, random forests, and gradient boosting.

Each dataset was represented with up to 15 different chart types, resulting in 459 charts in total. These charts were pivotal in training models to generate context-specific questions based on different types of visualizations. Figure 3 demonstrates an example of a plot that we generated.

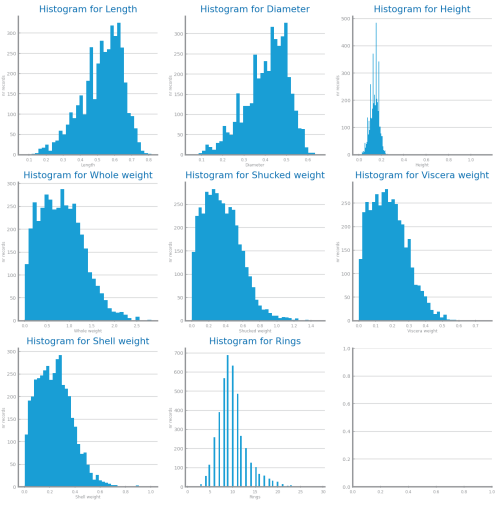

Fig. 3. Example of a plot used in the dataset.

4.4 Auxiliary Datasets

Throughout the development process, we created several auxiliary datasets derived from the main dataset. In this section, we will provide an overview of each of these datasets. Let's briefly explain each dataset, their purposes, and the structure of their records.

- **Chart Caption Dataset**: During the development of our question generation model, we felt the need to divide the task into two subtasks: generating a caption or description of the chart, and then generating a question based on that caption. This need originated the creation of this dataset to train a model for the first subtask. Each entry in this dataset consists of an image of a chart and its corresponding caption.
- **Caption Templates Dataset**: After analyzing the results of the training with the Chart Caption Dataset, we identified an opportunity for further improvement by dividing the caption generation task into two subtasks. The first subtask involves classifying the type of chart (e.g., boxplot, histogram) and assigning it a general caption template that does not include the variable names from the chart. The second subtask focuses solely on identifying the variable names. This dataset originates from the first subtask, enabling us to train a model to classify the input chart and assign an appropriate general template. Each record in this dataset comprises an image of a chart and its corresponding general template.
- **Chart Variables Dataset**: This dataset is derived from the second subtask described previously, where the goal is to train a model to identify the variable names in the input chart, allowing us to replace placeholders in the general template with these variable names. Each record in this dataset includes an image of a chart and a list of the variables represented within that chart.
- **Chart Data Dataset**: While the previous auxiliary datasets were created to develop the question generation model, this dataset was developed to aid in the creation of the question answering model. During the training of the question answering model, we realized the need of extracting more elements than just the variable names and the type of chart to answer the questions effectively. Therefore, this dataset was created to train a model to extract the required elements from the input chart. Each entry in this dataset consists of an image of a chart and its corresponding data. The specifics of the data that need to be extracted vary depending on the type of chart.

5 Sentence Generation

In tackling the task of sentence generation, our approach involved leveraging pre-existing models trained on similar tasks to capitalize on transfer learning. This decision was driven by the recognition that building a solution from the ground up would demand extensive computational resources and data, both of which we have in limited supply. In the subsequent section, we will delve into the training process of the question generator analyzing what composes each component of our solution.

5.1 Global Training Process

During the development of our solution, we trained several different models using a consistent training process, referred to as the global training process. The goal of this process is to train models that, at inference time, take an image as input and output text related to that image.

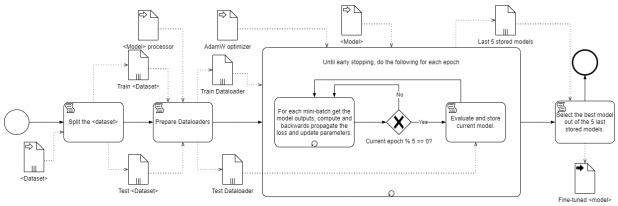

Fig. 4. Our global training process used to train image-to-text models.

The global training process, shown in Figure 4, begins with splitting a dataset of Image-Text pairs, allocating 90% for training and 10% for testing. Dataloaders are prepared by encoding images into tensors and tokenizing the text using the model's processor, such as Pix2Struct, which combines an image processor and a BERT tokenizer. The dataset is then batched in groups of two.

For each epoch, the model processes image encodings to generate tokenized text outputs. We calculate the loss, and update the parameters using backpropagation with the AdamW optimizer. Every fifth epoch, we evaluate the model on the validation dataset, with selecting the best-performing model from the last five saved models. We stop training when performance declines after two consecutive evaluations.

5.2 Specific Sentences Generation Process

The sentence generation process was divided into two distinct tasks to improve model performance:

5.2.1 First task: To Generate Charts Captions. The first task was to generate captions that describe the charts, including the chart type and the variables represented. For this, we created the Chart Caption Dataset of Chart-Caption pairs where the caption contains all the relevant data about the chart, which was used to train both Pix2Struct and GiT models. Even though we got good results with this approach, we decided that it was possible to improve further so the caption generation task was divided into two subtasks:

- Classifying Chart Types: We trained the models to classify chart types using a Caption Templates Dataset, which contained predefined description templates for 15 different chart types. This task involved recognizing distinct visual features of charts such as histograms, bar charts, and decision trees.
- Identifying Variables in Charts: The second subtask involved detecting the variables represented in each chart. For this, we created a Chart Variables Dataset. The models were fine-tuned to identify variables within charts, ensuring that generated questions referenced correct variables.

5.2.2 Second task: To generate the sentences from captions. After generating captions, the second task was to generate true/false questions from these captions using text-to-text models. We tested two models for this task:

- GPT-3.5 Turbo: A powerful language model fine-tuned for generating contextually accurate data science true/false questions.
- Mistral-7B: A smaller, more lightweight model that was also fine-tuned for this task.

We explored two approaches for this task:

- Zero-Shot Approach: In this approach, we used GPT-3.5 and Mistral to generate questions without fine-tuning, relying only on prompt engineering and a few examples in the prompt.
- Fine-Tuned Approach: In the fine-tuned approach, we trained the models on the Caption-Question Dataset, which linked chart captions with their corresponding true/false questions.

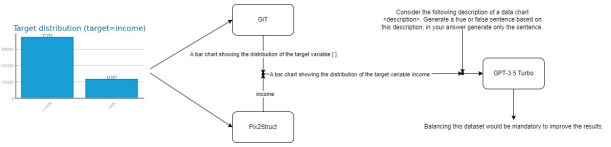

Fig. 5. The final model at inference time.

The best performing question generation model is illustrated in figure 5. The GiT model, fine-tuned for the chart classification sub-task, generates the caption template while the Pix2Struct model, fine-tuned for the variable identification sub-task, generates the variable list, these two components are then joined to form the caption, which in turn is integrated in a prompt and transfered to GPT-3.5 Turbo fine-tuned on the sentence generation task that generates the sentence.

5.3 Evaluation of models performance

We evaluated the performance of our models using well-established natural language generation metrics. These metrics included BLEU, ROUGE, and METEOR, which are commonly used in tasks such as machine translation and text summarization. Each metric focuses on different aspects of the generated text:

- BLEU [14]: Focuses on precision, ensuring the generated text uses the correct words and phrases found in the reference text.
- ROUGE [2]: Emphasizes recall, ensuring the generated text captures the important parts of the reference text.
- METEOR [10]: Balances precision and recall, also considering synonyms and stemming, providing a more semantic evaluation.

5.3.1 First attempt: Naive approach. As a baseline for the other approaches we will start with a naive approach by showing the results of both the Pix2Struct and GiT models when trained directly on the main dataset of Image-Question pairs.

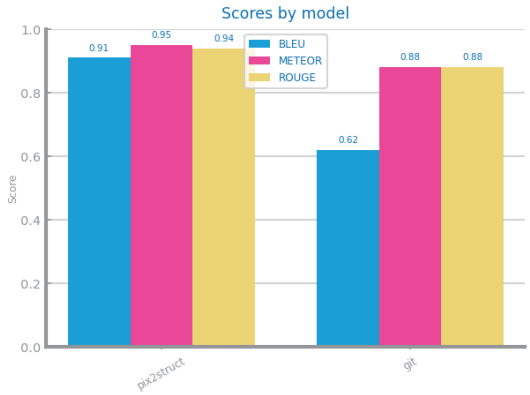

Fig. 6. Results of the GiT and Pix2Struct model when trained directly on the questions dataset.

Figure 6, shows the results obtained for both models, with the Pix2Struct model exhibiting slightly superior performance on the METEOR and ROUGE metrics with 0.95 and 0.94 and a much better performance on the BLEU metric with 0.91 when compared to GiT's 0.62. This can be due to the Pix2Struct model's capability to recognize text within the charts, something that the GiT model lacked.

While the results appear promising, two significant challenges persist. Firstly, the generated sentences lack creativity, as the models strictly adhere to generating sentences identical to the reference ones. Secondly, the models fail to identify the variables represented in each chart, leading to sentences with variable names that are not represented in the chart.

5.3.2 Second attempt: Dividing the problem into two tasks. In this attempt, we divided the question generation problem into two tasks: caption generation and sentence generation from captions. This section presents the results of training the Pix2Struct and GiT models on the Chart Caption Dataset to address the caption generation task.

Fig. 7. Results of the GiT and Pix2Struct model when trained on the image captions dataset.

At first glance the results of this caption generation task may seem comparable to the first attempt of directly generating questions, as seen in Figure 7. The Pix2Struct model achieved scores of 0.85 for BLEU, 0.88 for METEOR, and 0.98 for ROUGE, while the GiT model scored 0.75 for BLEU, 0.84 for METEOR, and 0.86 for ROUGE.

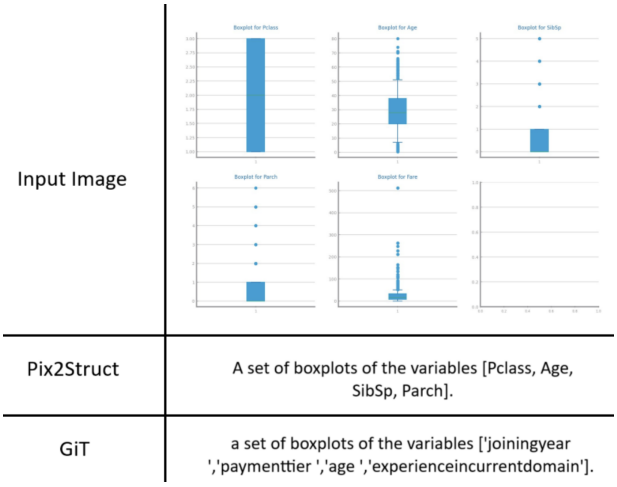

Fig. 8. Examples of captions generated by both models trained directly on the images captions dataset. We can see that the GiT model did not learn how to identify the variables in the charts, while the Pix2Struct model learned how to identify some variables.

However, if we delve deeper and manually examine the generated captions (one example is shown in Figure 8), we observe that the Pix2Struct model demonstrated the ability to identify variables depicted in the charts, which did not occur in the previous attempt.

5.3.3 Final Process. While the Pix2Struct model demonstrated promising results it still failed in a considerable amount of instances. Consequently, the core challenge still persisted in accurately identifying the variables depicted within the charts. To address this, we chose to streamline the task by dividing the generation of chart captions into two sub-tasks. We trained two separate models: one aimed at identifying the variables present in the chart, and the other focused on assigning a caption template to each chart.

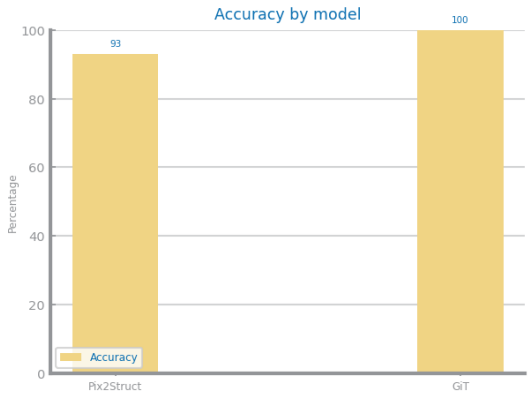

Fig. 9. Results of the GiT and Pix2Struct model when trained on the caption templates dataset.

In figure 9 we see the results for the sub-task of classifying the charts. We can see that both models achieved excellent results with the GiT model obtaining a perfect accuracy score and Pix2Struct achieving a score of 0.93.

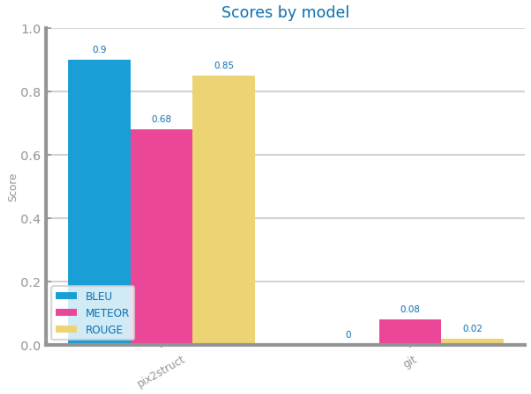

Fig. 10. Results of the GiT and Pix2Struct model when trained on the chart variables dataset.

The performance of both models in the sub-task of identifying variables in charts is illustrated in Figure 10. The performance of GiT was underwhelming, while for the Pix2Struct model, at first glance

it may seem that the results did not improve from the previous section. However, this isn't the case. Previously, we assessed the complete generated and reference captions, inflating the metrics, as it's easier to generate the caption template than to identify the variables. For instance, consider the comparison of two captions:

- Generated: "A set of histograms of the variables ['Age', 'pH', 'Specific Gravity']."
- Reference: "A set of histograms of the variables ['Age', 'pH']."

The BLEU score between the generated and the reference caption is 0.70. However, if we only consider the list of variables, the BLEU score drops to 0.37, showing a significant decrease.

Now let's move on to the task of sentence generation. As a quick reminder, we used two approaches for this task: the zero-shot approach and the fine-tuned approach.

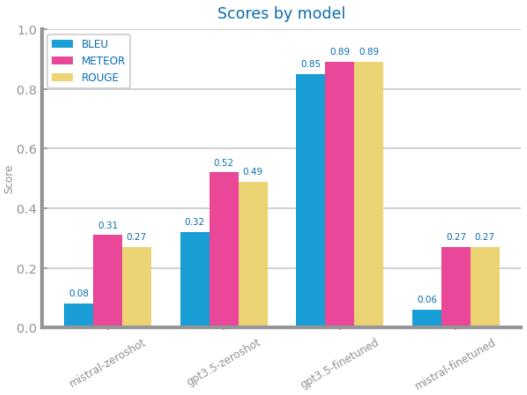

Fig. 11. Results of the mistral and GPT-3.5 Turbo models using both approaches.

In analyzing the results (refer to Figure 11), it becomes clear that fine-tuned GPT-3.5 Turbo achieved the best performance overall (BLEU: 0.85, METEOR: 0.89, ROUGE: 0.89). The zero-shot version of GPT-3.5 Turbo followed behind (BLEU: 0.32, METEOR: 0.52, ROUGE: 0.49), while the Mistral model did not yield satisfactory results in either the zero-shot (BLEU: 0.08, METEOR: 0.31, ROUGE: 0.27) or fine-tuned (BLEU: 0.06, METEOR: 0.27, ROUGE: 0.27) approaches.

Now let's see what were the results for the final_model, the model where we assemble all the best performing models to solve each one of the subtasks.

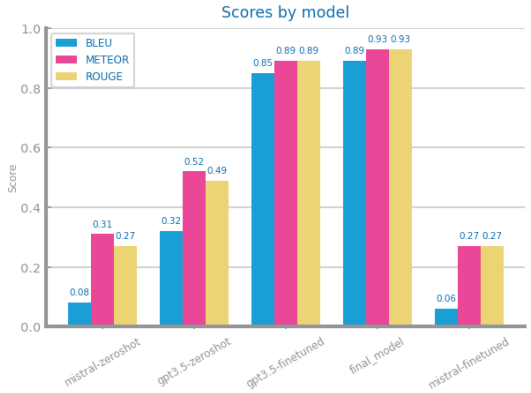

Fig. 12. The final model results compared to the previous models.

The results indicate that the final model performed well, even surpassing the GPT3.5 fine-tuned model. Although initially confusing, it's crucial to note that the reference sentences employed for evaluating the model were manually crafted using templates, incorporating numerous possible combinations of values to fill them. Hence, slight variations in scores between both models may occur, as they might generate correct sentences not present in the reference set. To account for this, we conducted a manual evaluation, the details of which will be discussed in the subsequent section.

5.4 Expert Evaluation

Although metrics serve as a useful tool for quantifying model performance, they primarily measure the similarity between the generated text and the reference text. Consequently, they may not accurately assess or even penalize the creativity exhibited in the generated sentences. To address this limitation and provide a more comprehensive evaluation of creativity while maintaining sentence quality, we enlisted the expertise of a data scientist to assign a score ranging from 1 to 5 to each generated sentence.

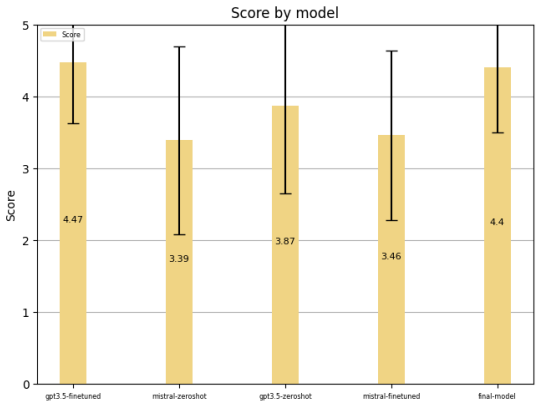

Fig. 13. The data scientist score mean by model, this mean only considers sentences that are possible to answer.

In Figure 13, we observe the mean scores across models, computed exclusively from sentences deemed answerable. Consistent with other metrics, both the final model and the GPT-3.5 fine-tuned one emerged as top performers, exhibiting lower variance in scores indicated by their smaller standard deviations.

This trend can be explaned by several factors. For the zero-shot models, the prevalence of non-interesting sentences likely contributes to their lower mean scores and higher variance. Meanwhile, the Mistral 7B model's inferior performance across all metrics can be attributed to its smaller parameter count compared to GPT-3.5.

5.5 Considerations

Through this manual evaluation we concluded that balancing creativity and adherence to reference standards proves challenging, as overly creative models may generate theoretical or overly simplistic sentences. Thus, models that closely align with reference sentences while incorporating a degree of creativity tend to yield better results. However, caution is warranted, as models may replace data science terms with others that may confuse students. One potential research avenue involves initially training the model on domain knowledge, perhaps using a data science terms dictionary to ensure consistent terminology usage. Additionally, attention to chart generation is crucial, as charts must be tailored for educational purposes to minimize ambiguity in generated sentences, offering ample opportunities for future research in this area.

6 Question Answering

For the task of question answering, we built upon the process we used for question generation. However, the question answering task brings additional challenges, such as extracting more complex data from charts and leveraging domain knowledge to accurately classify questions as true or false. So, our approach is composed of two core steps: data extraction and question classification, used a combination of pretrained models fine-tuned for these tasks.

6.1 Training Process

To tackle this task, we divided the question-answering process into two sub-tasks: data extraction and question answering. In the data extraction task, we generate a caption containing all the necessary elements to answer questions about the chart. In the question answering task, given a question and a caption with the chart's data, we classify the question/sentence as true or false.

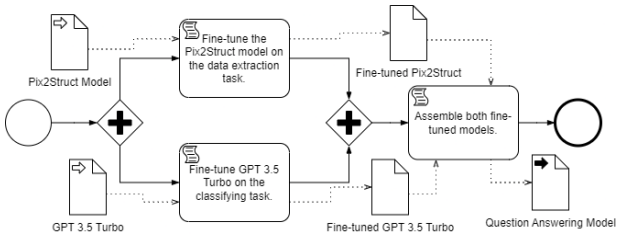

Fig. 14. The global overview of the training process.

The training process, illustrated in Figure 14, begins with fine-tuning the Pix2Struct model for the task of extracting data from charts. This process is detailed in section 6.2. Next, we fine-tune the GPT-3.5 Turbo model for the question answering task, as described in section 6.3. Once both models are fine-tuned, we combine them to create the final question-answering model.

At inference time, the process starts with the input chart being transfered to Pix2Struct, which generates a caption containing the necessary chart data. This caption is formatted into a prompt and sent to GPT-3.5 Turbo, along with the question. The model then classifies the sentence as true or false based on the chart data.

6.2 Extract Chart Data

Our initial intuition for tackling this task was to reuse the component that generates chart captions from the visual question generation process. However, we encountered a problem with this approach: answering questions requires more detailed information than generating questions. For example, consider the sentence, "Variable Age is balanced.". To generate this question, we only need to extract the variable name from the chart. However, to classify the sentence as true or false, we need to determine whether the variable "Age" is actually balanced, which requires extracting more detailed information beyond just the variable names.

6.2.1 Process description. The first step in addressing the task of extracting data was to create a new auxiliary dataset consisting of Chart-Data pairs, where "Data" is a caption encapsulating all the data elements needed to answer questions about the chart. This auxiliary dataset is referred to as the Chart Data Dataset, explained in section 4.4.

With the dataset prepared, we chose to finetune the Pix2Struct model, due to its superior performance in the task of identifying variables on charts, as shown in section 5.3.3.

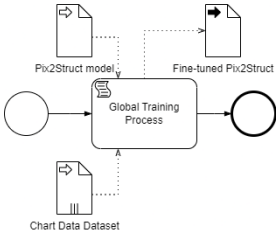

Fig. 15. Training process of the model.

The training process is illustrated in Figure 15. Using our Global Training Process described in section 5.1, we provided the Pix2Struct model and the Chart Data Dataset as input and then the Global Training Process outputs the fine-tuned Pix2Struct model.

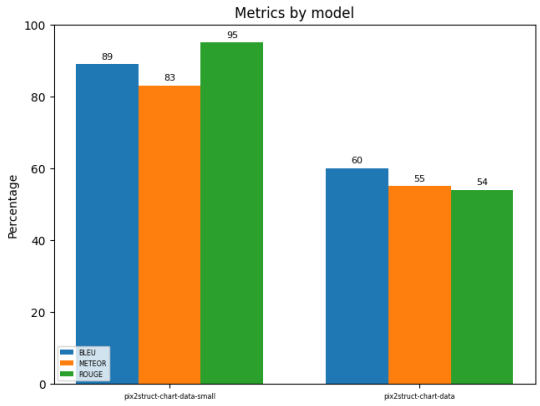

Fig. 16. Results of the models: on the left, the results using only the charts where the model performed better; on the right, the overall results.

6.2.2 Evaluation. To evaluate the results of this task, we used the same metrics as before: BLEU, METEOR, and ROUGE. The model's performance, shown in Figure 16, indicates that when considering all types of charts, the results are not satisfactory (BLEU: 0.6, METEOR: 0.55, ROUGE: 0.54). However, if we focus only on charts where the model successfully extracted all the necessary elements (decision trees, PCA histograms, and all bar charts), the results improve significantly (BLEU: 0.89, METEOR: 0.83, ROUGE: 0.95).

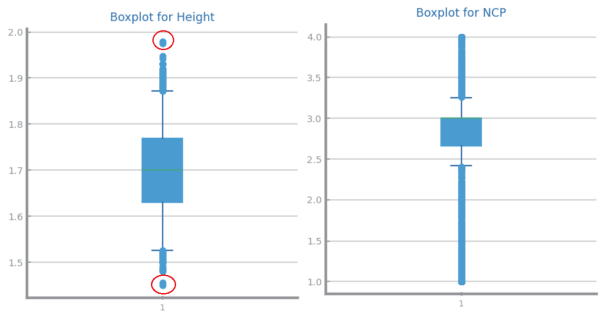

Fig. 17. The encircled dots are outliers. According to our definition, the chart on the right has no outliers.

From a manual evaluation, we discovered that the model struggles with identifying spacing and colors. For example, in terms of spacing, one of the problems was that the model cannot accurately determine whether a boxplot has outliers. Normally, any dots beyond the whiskers would be considered outliers. However, our definition considers a dot as an outlier only if there is a gap between it and the main distribution. This distinction, as shown in Figure 17, proved too challenging for the model to learn.

The takeaway from this evaluation is that the Pix2Struct model faces challenges with tasks requiring spatial awareness. This is due to its pretraining approach, which represents the underlying structure of the image in HTML [8]. While this method is effective for representing visually situated text, it does not perform well with tasks involving spatial awareness and color recognition. However, this architecture excels in chart-related tasks where information extraction relies heavily on visually situated text. Examples include decision trees, histograms, and bar charts both with numbers on top of the bars. As shown in Figure 16, the results for these types of charts were significantly better (BLEU: 0.89, METEOR: 0.83, ROUGE: 0.95).

6.3 Classifier

The next step is to train a model to classify input sentences as true or false based on the data extracted from the chart. For the model to perform this classification, it needs to have domain knowledge about data science. While several approaches could be employed, such as representing domain knowledge in a graph structure or using a dictionary with definitions, we chose a more generalist approach. We opted to use a model pre-trained on a large corpus of text including a diverse range of topics, like GPT-3.5 Turbo, as its pre-training includes data science concepts necessary for classifying the questions.

6.3.1 Process Description. We employed two approaches:

- Zero-Shot Approach: In this approach, GPT-3.5 received the formatted prompt and classified the sentence without fine-tuning.
- Fine-Tuned Approach: We fine-tuned GPT-3.5 using a prompt dataset containing both chart data and correct answers (true or false).

The prompt used for both the zero-shot and the fine-tuned approaches was the same and is as follows:

- system: You are a student doing a data science exam.
- user: Consider the following chart [formatteddata], classify the following sentence about the chart as true or false: [sentence].

To fine-tune the GPT-3.5 Turbo model using the OpenAI API, we created a dataset of prompts using the format described above, with the addition of the answer: "assistant: [True or False]".

6.3.2 Evaluation. For evaluation, we used standard classification metrics, since ultimately this is a binary classification task where we need to classify the input sentence as true or false: accuracy, recall, precision, specificity, and F1 score.

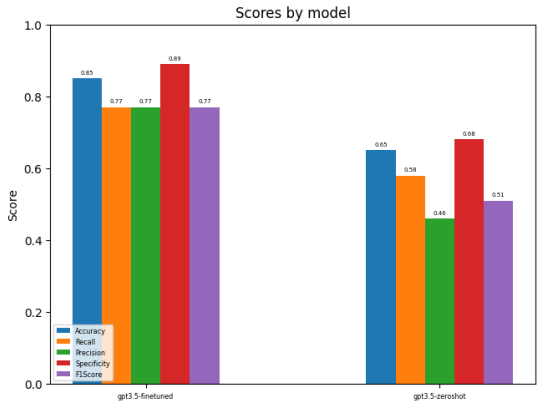

Fig. 18. Results of both the zero-shot and finetuned approaches.

In Figure 18 we can see the results of both zero-shot and fine-tuned approaches. We found that the zero-shot model underperformed (Accuracy: 0.65, Recall: 0.58, Precision: 0.46, Specificity: 0.68, F1Score: 0.51). In contrast, the fine-tuned model achieved significantly better results (Accuracy: 0.85, Recall: 0.77, Precision: 0.77, Specificity: 0.89, F1Score: 0.77).

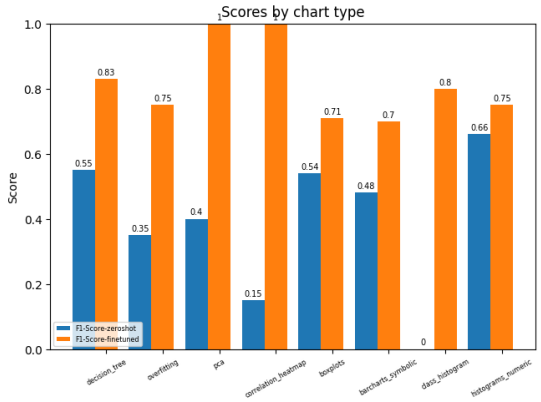

Fig. 19. F1-Score of both the fine-tuned and zero-shot approaches by chart type.

To gain deeper insights and determine whether improvements were consistent across all chart types, we compared the F1-Score for each chart type between both models (Figure 19). We chose this metric instead of accuracy because the test dataset was unbalanced, with 105 true labels and 218 false labels. We observed that, for all chart types, the fine-tuned model achieved a higher F1-Score (Decision Trees: 0.83, Overfitting Charts: 0.75, PCA Histograms: 1, Correlation Heatmap: 1, Boxplots: 0.71, Bar Charts: 0.7, Class Histograms: 0.8, Histograms: 0.75) compared to the zero-shot model (Decision Trees: 0.55, Overfitting Charts: 0.35, PCA Histograms: 0.4, Correlation Heatmap: 0.15, Boxplots: 0.54, Bar Charts: 0.48, Class Histograms: 0, Histograms: 0.66).

6.4 Full evaluation

With all the necessary components in place, we can now assemble the final model using the fine-tuned Pix2Struct for extracting chart data and the fine-tuned GPT-3.5 for classifying sentences. The results of the final model are shown in Figure 20.

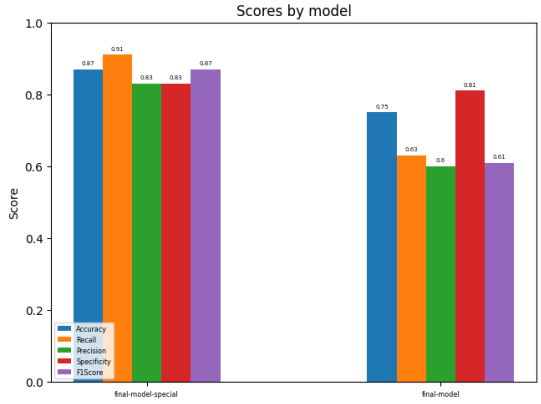

Fig. 20. Results of the final model with all types of charts (on the right) and on the left, using only the charts where the Pix2Struct model performed better (decision trees, pca histogram and all bar charts).

While the final model achieved good results in accuracy (0.75) and specificity (0.81), it did not perform as well in recall (0.63), precision (0.6), and F1 score (0.61). This is due to the Pix2Struct model's poorer performance on charts requiring spatial awareness. However, when these charts are removed from the evaluation, the results improve significantly across all metrics (Accuracy: 0.87, Recall: 0.91, Precision: 0.83, Specificity: 0.83, F1Score: 0.87), aligning with the fine-tuned GPT-3.5 model's results.

7 Conclusion

The need for online learning personalization and a continuous supply of learning resources demands the automatic generation of learning materials, particularly questions. Despite promising advances in generating textual questions through NLP tools, the results for visual-based questions remain unsatisfactory.

We developed a pipeline that processes datasets to create data plots and corresponding true/false sentences, used to train two models: one for generating questions and another for classifying them as true or false. Our architecture, combining Pix2Struct and GPT-3.5, achieved high-quality results in both tasks. The question generation model balanced creativity with quality, although further refinement is needed to ensure consistent use of domain-specific terminology. The question answering model demonstrated strong performance in classifying questions for certain chart types, supported by Pix2Struct's effective data extraction.

Future work could focus on enhancing chart data extraction, representing domain knowledge in graph structures, and developing specialized models for different chart types. Expanding the system to handle multi-chart question generation and classification is also

a promising direction. The architecture developed serves as a foundation for more advanced models capable of better handling the complexities of question generation and answering from visual data.

References

[1] [n. d.]. Matplotlib, a Python library for visualization. https://matplotlib.org/.

[2] Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*. 65–72.

[3] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems* 33 (2020), 1877–1901.

[4] Zhihao Fan, Zhongyu Wei, Siyuan Wang, Yang Liu, and Xuan-Jing Huang. 2018. A reinforcement learning framework for natural question generation using bi-discriminators. In *Proceedings of the 27th International Conference on Computational Linguistics*. 1763–1774.

[5] Difei Gao, Ruiping Wang, Shiguang Shan, and Xilin Chen. 2022. Cric: A vqa dataset for compositional reasoning on vision and commonsense. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45, 5 (2022), 5561–5578.

[6] Drew A Hudson and Christopher D Manning. 2019. Gqa: A new dataset for real-world visual reasoning and compositional question answering. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 6700–6709.

[7] Ranjay Krishna, Michael Bernstein, and Li Fei-Fei. 2019. Information maximizing visual question generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2008–2018.

[8] Kenton Lee, Mandar Joshi, Iulia Raluca Turc, Hexiang Hu, Fangyu Liu, Julian Martin Eisenschlos, Urvashi Khandelwal, Peter Shaw, Ming-Wei Chang, and Kristina Toutanova. 2023. Pix2struct: Screenshot parsing as pretraining for visual language understanding. In *International Conference on Machine Learning*. PMLR, 18893–18912.

[9] Yikang Li, Nan Duan, Bolei Zhou, Xiao Chu, Wanli Ouyang, Xiaogang Wang, and Ming Zhou. 2018. Visual question generation as dual task of visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 6116–6124.

[10] Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*. 74–81.

[11] Wes McKinney et al. 2010. Data structures for statistical computing in python. In *Proceedings of the 9th Python in Science Conference*, Vol. 445. Austin, TX, 51–56.

[12] Nasrin Mostafazadeh, Ishan Misra, Jacob Devlin, Margaret Mitchell, Xiaodong He, and Lucy Vanderwende. 2016. Generating natural questions about an image. *arXiv preprint arXiv:1603.06059* (2016).

[13] Keiron O'Shea and Ryan Nash. 2015. An introduction to convolutional neural networks. *arXiv preprint arXiv:1511.08458* (2015).

[14] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*. 311–318.

[15] Charulata Patil and Manasi Patwardhan. 2020. Visual question generation: The state of the art. *ACM Computing Surveys (CSUR)* 53, 3 (2020), 1–22.

[16] Badri N Patro, Sandeep Kumar, Vinod K Kurmi, and Vinay P Namboodiri. 2018. Multimodal differential network for visual question generation. *arXiv preprint arXiv:1808.03986* (2018).

[17] Mengye Ren, Ryan Kiros, and Richard Zemel. 2015. Exploring models and data for image question answering. *Advances in neural information processing systems* 28 (2015).

[18] Burr Settles. 2009. Active learning literature survey. (2009).

[19] Kohei Uehara and Tatsuya Harada. 2023. K-VQG: Knowledge-aware Visual Question Generation for Common-sense Acquisition. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 4401–4409.

[20] Guido Van Rossum and Fred L. Drake. 2009. *Python 3 Reference Manual*. CreateSpace, Scotts Valley, CA.

[21] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[22] He Zhu, Ren Togo, Takahiro Ogawa, and Miki Haseyama. 2023. A Medical Domain Visual Question Generation Model via Large Language Model. In *2023 International Conference on Consumer Electronics-Taiwan (ICCE-Taiwan)*. IEEE, 163–164.

Received 13 October 2024