SCHOOL OF SCIENCE & ENGINEERING

FALL 2023

CSC535401 NLP for Big data

Project

NLP-driven Automated Test Generation: Generating Test Questions in English Using LLMs Based on Educational Texts

May 3rd, 2025

Realized by:

Noura Ogbi

Supervised by:

Prof. Violetta Cavalli-Sforza

Table of Contents

I. Introduction

This project presents a transformer-based system designed to automatically generate educational test questions from educational texts. The system is expected to produce questions in various formats, including general open-ended questions, fill-in-the-blank (cloze) exercises, and multiple-choice questions (MCQs). It uses input paragraphs and generates one of the question types based on a randomized prompt. The goal is to build a basic automated question generation pipeline that can support different formats of assessment questions.

To build this system, the project leverages the **Stanford Question Answering Dataset (SQuAD)**, a reading comprehension dataset consisting of questions posed by crowdworkers on a set of Wikipedia articles[1]. It provides a rich corpus of context paragraphs aligned with human-authored questions and answers. At the heart of the system is **Flan-T5 Large**, a powerful instruction-tuned variant of the T5 model developed by Google. This model has demonstrated strong performance across a broad range of tasks due to its ability to generalize well when prompted with natural instructions[2]. The model was fine-tuned on a customized version of SQuAD, where examples were dynamically reformatted into multiple question types.

The entire training pipeline was implemented using Hugging Face's Transformers and Datasets libraries and executed on Google Colab equipped with an **NVIDIA A100 GPU**. The training setup involved preparing the data, tokenizing the inputs and targets, and fine-tuning the model using standard sequence-to-sequence training procedures. The system was evaluated using BLEU and ROUGE metrics across validation sample.

II. Literature Review

Text transformer technology has changed educational assessment through Large Language Models (LLMs) like GPT-3. These models show remarkable skill at knowing how to understand and generate human-like text for tasks that only human educators could do before. They create many types of questions—from basic facts to complex critical thinking challenges. These questions line up perfectly with educational goals and give students instant feedback. The transformer architecture powers both text-to-text transformer models and specialized variants like the T5 transformer to create relevant educational content[3]. Zero-shot, few-shot, and chain-of-thought prompting techniques boost the quality of generated questions. Teachers now have a substantially lighter workload and can focus on complex teaching duties instead of routine question creation. LLMs have brought a fundamental change to educational assessment. They create individual-specific learning experiences that adapt to each student's needs[4].

Modern test generation systems are built on encoder-decoder architecture. Traditional methods needed manual rules, but transformer-based models can process full sequences at once through self-attention mechanisms[5]. This makes test question generation quick and effective. Google Research introduced the Text-to-Text Transfer Transformer (T5) in 2019, which became central to educational question

[1] The Stanford Question Answering Dataset
[2] google/flan-t5-large · Hugging Face
[3] https://www.byteplus.com/en/topic/408050
[4] https://ieeexplore.ieee.org/document/10051840/
[5] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). *Attention is all you need*.

generation. T5 turns every NLP task into a text generation challenge, whatever the requirements might be. The model's encoder-decoder structure gets pre-training on massive text datasets, which makes it versatile enough to generate educational assessment questions. FLAN-T5 brought major improvements to few-shot learning capabilities. The base T5 model gets training on general text, while FLAN-T5 receives instruction fine-tuning on tasks of all types using natural language instructions. FLAN-T5's performance on complex reasoning tasks jumped 20–30% compared to the original T5 model[6].

These models work through a self-attention mechanism—their biggest innovation. Each word can "attend" to all other words and calculate relevance through scaled dot-product attention. This creates rich contextual representations that help understand educational content. The positional encoding technique makes sure word order affects meaning—two tokens that sit closer have more similar positional vectors[7]. Transformer architectures shine at test generation by creating detailed test cases from educational materials automatically. They analyze requirements, spot key concepts, and create relevant questions that cover different scenarios. The models even catch edge cases that human testers might miss. This has changed how teachers create educational assessments—they can now quickly generate different types of questions from their teaching materials.

Question generation with text transformer models relies on effective prompting as its foundation. Different prompting strategies help educators extract specific types of questions from educational texts. The model needs no additional training or modification. Zero-shot prompting lets models generate educational questions without examples. This technique uses the model's pre-existing knowledge and clear instructions guide the response generation. To cite an instance, a simple directive like "Generate three multiple-choice questions about photosynthesis" produces relevant assessment items. Few-shot prompting boosts question generation quality by adding 2–5 examples in the prompt. The T5 transformer learns what type of questions to expect, which creates a template to follow[8]. Educational contexts need consistent formatting across examples. Each example should follow similar patterns. Chain-of-thought (CoT) prompting works especially well to generate complex assessment questions that need multi-step reasoning. The best results often come from combining these approaches. Few-shot examples paired with chain-of-thought instructions create prompts that show both the desired format and reasoning process.

III. Code and Implementation

This project follows a modular and interactive design approach, starting from raw educational data and ending with a deployed demo system that can generate questions into three types of test questions: general open-ended, fill-in-the-blank (cloze), and multiple-choice questions:

Step 1: Preparing the Workspace

The first step is setting up the environment. This includes installing essential NLP libraries like *spaCy, nltk,* datasets, evaluate, and transformers, as well as downloading a pretrained English language model *(en_core_web_sm)* for Named Entity Recognition.

[6] https://www.byteplus.com/en/topic/408050
[7] https://www.ibm.com/think/topics/attention-mechanism
[8] https://www.byteplus.com/en/topic/408050

Step 2: Loading and Cleaning the Dataset

We use the popular SQuAD dataset (Stanford Question Answering Dataset), which includes pairs of context paragraphs and questions with answers. We randomly split it into three parts: training (80%), validation (10%), and test (10%).

Before using the text, we clean it by removing extra white spaces and special characters to make it more suitable for machine processing. This is done using a simple regular expression-based function.

Step 3: Creating Input-Output Examples

To simulate a real-world application, each input is transformed into one of three question types:

- **General** (open-ended questions)
- **Fill-in-the-Blank**
- **Multiple Choice**

A custom function randomly decides which format to generate for each example. It intelligently extracts the correct answer from the context and:

- Blanks it out for cloze questions
- Finds named entities to use as distractors for MCQs
- Or keeps the original open question for general cases

This step creates a new dataset with pairs of prompts (*input_text*) and expected model responses (*target_text*).

Step 4: Tokenizing the Data for the Model

Since the model operates on tokenized input, we use the tokenizer from **Flan-T5-Large**. Both inputs and targets are tokenized, and special padding is handled carefully by masking padded tokens during loss calculation.

Step 5: Fine-Tuning the Flan-T5 Model

We use Hugging Face's Trainer to fine-tune Flan-T5-Large on our custom-formatted dataset. The training uses:

- A small batch size with gradient accumulation to manage memory
- Early stopping to prevent overfitting
- Validation checkpoints every 1000 steps

The model learns to generate context-aware questions based on the input format prompt.

Step 6: Making Predictions

After training, we generate predictions on unseen test samples. The model receives an input prompt (e.g., “Generate a multiple-choice question from this context: ...”) and produces a syntactically and semantically complete question.

We save and compare these outputs with reference questions to evaluate performance.

Step 7: Evaluation and Interface

We use BLEU and ROUGE scores to measure how close the generated questions are to the human-written references. Additionally, we deploy the model with a **Gradio interface**, allowing users to:

- Enter their own paragraph
- Choose a question type
- Receive a generated question instantly

This demo showcases the real-time capabilities of the system.

IV. Evaluation

Once the model was trained, it was essential to check whether it was actually generating good questions, ones that are relevant, clear, and useful for educational purposes. To do this, the project used a mix of **automated scoring**, **example inspection**, and **real-user interaction** via a demo.

Dataset	BLEU Score	ROUGE-L
Validation	0.4254	0.5047
Test	0.5339	0.5292

- The **BLEU score above 0.5** on the test set indicates that the model is often generating questions that are reasonably close to what a human would write.
- The **ROUGE-L score above 0.5** confirms that the structure of the generated questions is also aligned with the references.

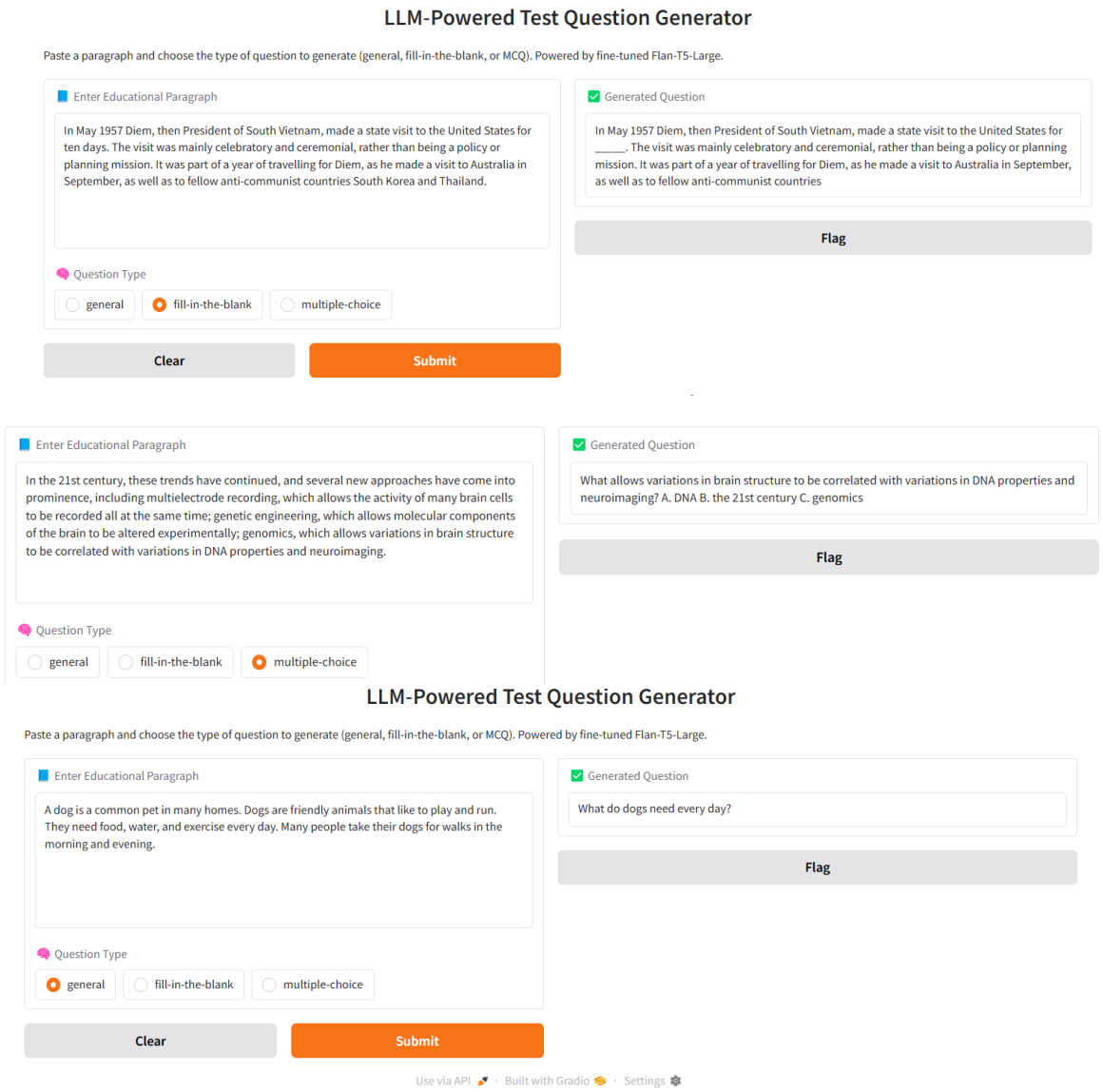
In addition to quantitative metrics, a manual review of several generated examples was conducted to qualitatively assess the system’s performance.

Here's what was noticed:

- **General questions** were often accurate but occasionally vague or too broad.
- **Fill-in-the-blank** questions sometimes failed to insert the blank correctly — especially if the correct answer wasn't clearly matched in the sentence.
- **MCQs** occasionally included distractors that were either too obvious or not well related.

V. Demo

To make the question generation system more interactive and easier to test, a simple Gradio demo was implemented. This interface allows users to enter a custom paragraph and select the type of question they want to generate, either a general question, a fill-in-the-blank, or a multiple-choice question. Once submitted, the system processes the input and returns a generated question in real time.



VI. Challenges Faced

At the beginning of the project, the initial goal was to build a question generation system for Arabic educational texts. The motivation behind this was both personal and academic, as Arabic remains significantly underrepresented in natural language processing research. However, once implementation began, it quickly became clear that working with Arabic involved several unexpected hurdles. The scarcity of high-quality, labeled datasets, the difficulty of handling right-to-left script formatting, and the need for specialized tokenization and preprocessing tools made the task more complex than anticipated. Despite the enthusiasm, these challenges proved too time-consuming to overcome within the project's timeframe. After investing considerable effort, I decided to switch to English in order to

focus on building a fully functional and evaluable system—a necessary but costly shift in terms of time and planning.

The switch to English brought its own set of technical obstacles. Installing and configuring essential libraries like transformers, evaluate, and accelerate proved difficult, especially when working in hosted environments such as Kaggle and Colab where pre-installed packages were either outdated or conflicting. To reduce resource usage, I first experimented with smaller models like BERT, T5-Small, and T5-Base. While these models allowed me to test the pipeline structure, they didn't deliver the quality needed for question generation, especially for multiple-choice or reasoning-based outputs. Eventually, I subscribed to Colab Pro to access A100 GPUs and run Flan-T5-Large. This greatly improved training speed and performance. Nonetheless, I still had to manage long training times, session timeouts, and environment inconsistencies, which made the development process more demanding than initially expected. Also, I experimented with two datasets—**SQuAD** and **HotpotQA**—to improve the variety and depth of the generated questions. While HotpotQA offered valuable multi-hop reasoning examples, processing both datasets simultaneously quickly exceeded the available computational resources.

VII. Possible Enhancements

There are several ways this project could be improved in the future. One idea is to go back to using the **HotpotQA** dataset, which includes more complex questions that require reasoning across multiple sentences. I had tried to include it earlier, but it was too heavy for the available resources. With better computing power, it could help the system generate more advanced and meaningful questions.

Another improvement would be to use **Optuna** for automatic hyperparameter tuning. Instead of manually trying out different settings like learning rate and batch size, Optuna could help find better values faster and more efficiently. It would also be interesting to add **multilingual support** in the future and revisit the original idea of generating questions in Arabic.

VIII. Conclusion

This project successfully demonstrated the feasibility of using Large Language Models (LLMs) to automate the generation of educational test questions from textual content. By fine-tuning the Flan-T5-Large model on a reformatted version of the SQuAD dataset, the system was able to generate three distinct types of questions: general, fill-in-the-blank, and multiple-choice. The implementation pipeline incorporated data cleaning, format-specific prompt generation, and named entity recognition to enhance the quality and diversity of the output.

Evaluation using BLEU and ROUGE-L scores, supplemented by manual inspection, indicated that the model produced syntactically coherent and semantically relevant questions in many cases. However, certain limitations were noted, such as occasional blanking issues in cloze questions and inconsistent distractor quality in MCQs. These observations highlight areas for future refinement.

Overall, the project validates the potential of LLMs as practical tools in the educational technology domain, offering promising support to educators in streamlining the assessment design process.

IX. References

TempestVanSchaik, "Evaluation metrics," Microsoft.com, Jun. 25, 2024. https://learn.microsoft.com/en-us/ai/playbook/technology-guidance/generative-ai/working-with-llms/evaluation/list-of-eval-metrics Raffel, et al. "Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer." *JMLR* (2020).

Brown, et al. "Language Models are Few-Shot Learners (GPT3)." *NeurIPS* (2020).

Vaswani, et al. "Attention Is All You Need." *NeurIPS* (2017).

Bergmann, Dave and Cole Sryker. *What is an attention mechanism?* 4 December 2024.

*"FLAN-T5 vs T5: Key Differences Explained," Byteplus.com, 2025. https://www.byteplus.com/en/topic/408050?title=flan-t5-vs-t5-key-differences-explained (accessed May 04, 2025).*Maity, Subhankar and Aniket Deroy. "The Future of Learning in the Age of Generative AI: Automated Question Generation and Assessment with Large Language Models." *arxiv* (n.d.). <https://arxiv.org/html/2410.09576v1>.

Maity, Subhankar, Aniket Deroy and Sudeshna Sarkar. "Can large language models meet the challenge of generating school-level questions?" *ScineceDirect* (2025).

A. Hadifar, S. K. Bitew, J. Deleu, C. Develder and T. Demeester, "EduQG: A Multi-Format Multiple-Choice Dataset for the Educational Domain," in IEEE Access, vol. 11, pp. 20885-20896, 2023

Van Campenhout, R., Hubertz, M., Johnson, B.G. (2022). Evaluating AI-Generated Questions: A Mixed-Methods Analysis Using Question Data and Student Perceptions. AIED 2022.