Detecting duplicate multiple choice questions (MCQs) in the large question bank

Final Year Project Final Report

A 4th Year Student Name

Phan Duc Manh

Nguyen Tuan Minh

Instructor

Assoc. Prof. Phan Duy Hung

FPT UNIVERSITY

Bachelor of Artificial Intelligence

Hoa Lac campus - FPT University

– Hanoi, 6 May 2023 –

ACKNOWLEDGEMENT

Foremost, we are especially indebted to our advisor Assoc. Prof. Phan Duy Hung for the invaluable guidance throughout this thesis. His dynamism, enthusiasm, motivation, and immense knowledge have deeply inspired our group.

We would like to express our special thanks to University, FPT, for giving us a golden opportunity to study and all of the resources provided.

Finally, our sincere thanks also go to our friends and family for their support and love during this process. We would not have been able to finish our journey without their encouragement and motivation.

ABSTRACT

A question bank is a database of questions in a variety of formats, used as a central repository for building tests. Question banks are the core tool for innovative testing and assessment of learners' learning outcomes. It can be built and added over time, from many sources, and many people. So creating new questions may result in duplicate questions and deciding whether to include that question in the real database will take time to manually search across data files. This study proposes an end-to-end machine learning architecture to combine the information from the text data and the data from the image in question through optical character recognition into an encoded vector. From there, the system can query to find similar questions based on similarity ranking. The machine learning model was evaluated on a part of the question bank of FPT University, a university of information technology in Vietnam. The obtained F1 score of 0.95 proves that the model can be used for intelligently managing the question bank of the FPT education system as well as of other educational institutions.

Keywords: *Question Bank, Duplicate Question Detection, Similarity Score.*

Table of Contents

List of abbreviations and acronyms

Abbreviations	**Meaning**
MC	Multiple choice
MCQs	Multiple choice questions
SAT	Scholastic Aptitude Test
TOEIC	Test of English for International Communication
IELTS	International English Language Testing System
EOS	Exam online system
NLP	Natural language processing
TF-IDF	Term frequency and inverse document frequency
VSM	Vector space model
CQA	Community-based question-answer
CNN	Convolutional neural network
RNN	Recurrent neural network
LSTM	Long short-term memory
WV-LSTM	Word-vector long short-term memory
OCR	Optical Character Recognition
EAST	Efficient and Accurate Scene Text detection
CRAFT	Character Region Awareness for Text detection
CRNN	Convolutional recurrent neural network
POS tagging	Part-of-Speech tagging
BERT	Bidirectional Encoder Representations from Transformers
SBERT	Sentence-BERT
STS task	Semantic Textual Similarity task
nltk	Natural Language Toolkit

List of tables

List of figures

1. Project Introduction

1.1. Problem & Motivation

Multiple choice (MC) objective response or multiple-choice questions (MCQs) was created in the early 20th century by E. L. Thorndike and B. D. Wood. It became widely used after almost fifty years. Nowadays, multiple-choice questions are one of the most commonly used in exam construction and popular survey question types. A multiple-choice question is a form of inquiry where the respondent must select one or more items from a constrained list of options. The component of multiple-choice questions is the question text, answer options, and correct answers. The question text describes the issues with closed-ended questions or an incomplete statement to be finished to make up the complete statement. The correct answer is a single choice or a list of choices in multiple response question type. Multiple choice format is superior to other testing formats since the cost-effective advantage when the outcome is usually checked using scanners and data processing devices, elimination of examiner bias. The set of multiple-choice questions with the same subject is stored and can be repeatedly called question banks. Making a question bank reduces cheating since the pool of questions are large and diverse. The identical questions appear in different sets and in a random order for the students. In recent years, with the high demand for examination, multiple-choice questions have been utilised universally throughout all educational testing, market research, and elections when a person chooses between multiple candidates, parties, or policies. Many international examinations evaluate candidates using a multiple-choice style such as: the Scholastic Aptitude Test (SAT), Test of English for International Communication (TOEIC), International English Language Testing System (IELTS), etc.

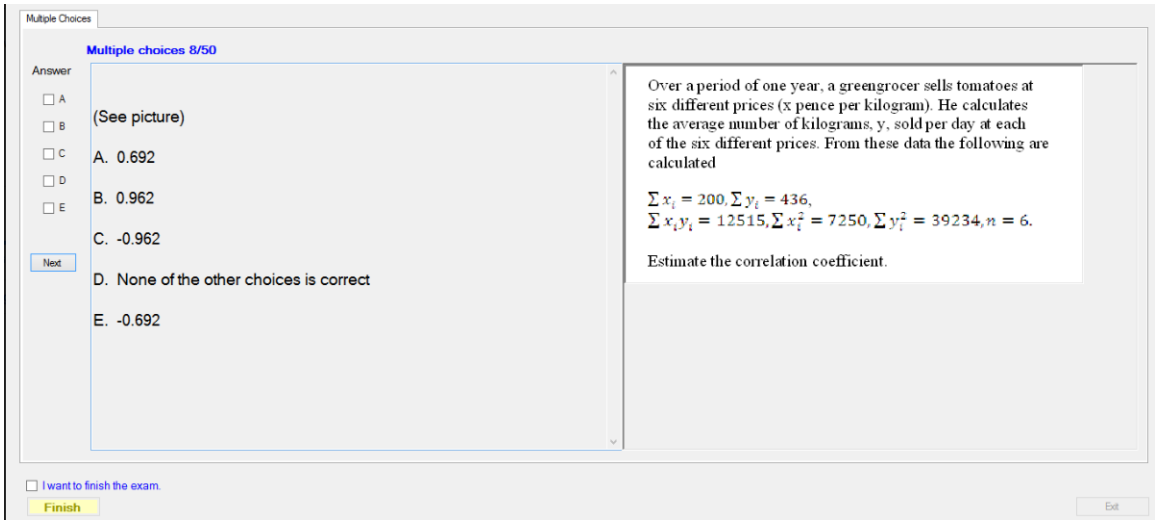

Figure 1. EOS exam software interface

In FPT university [1], the exam online system named EOS (Figure 1) has been used to generate exams for student evaluation. The EOS system takes multiple-choice questions form the question bank which is the main database for storing questions. There are two methods for adding new questions to the question bank at FPT university. In case one (Figure 2), the creation of the question bank is the responsibility of one teacher, and the question bank is generated into a word file and makes all information in the databases accessible. The instructor manually identifies duplicate questions before adding a new question to the database. The created questions must be eliminated if their contents are similar; else, new questions are imported into the question bank. Because the databases are completely accessible, it causes a risk of revealing question information and the import process is costly. In case two (Figure 3), the subject's primary lecturer divides the curriculum into many sub-categories for instructors to create new questions. Due to the variety in the question's

substance and content relationships between subcategories, many teachers may generate similar questions. Additionally, teachers are incapable of engaging with question bank databases, hence they are unable to determine whether or not questions are present in the databases. The head of the department is responsible for grouping all the doc files and manually checking duplicate questions. If the generated questions have similar contents, they must be reported back to the subject's primary lecturer to change or remove duplicate questions before it added to the question bank. In both two approaches, duplicate questions are identified through keyword manual searches, making the identification process difficult and time-consuming. This difficulty is exacerbated when some questions are stored as images, which further complicates the search process.

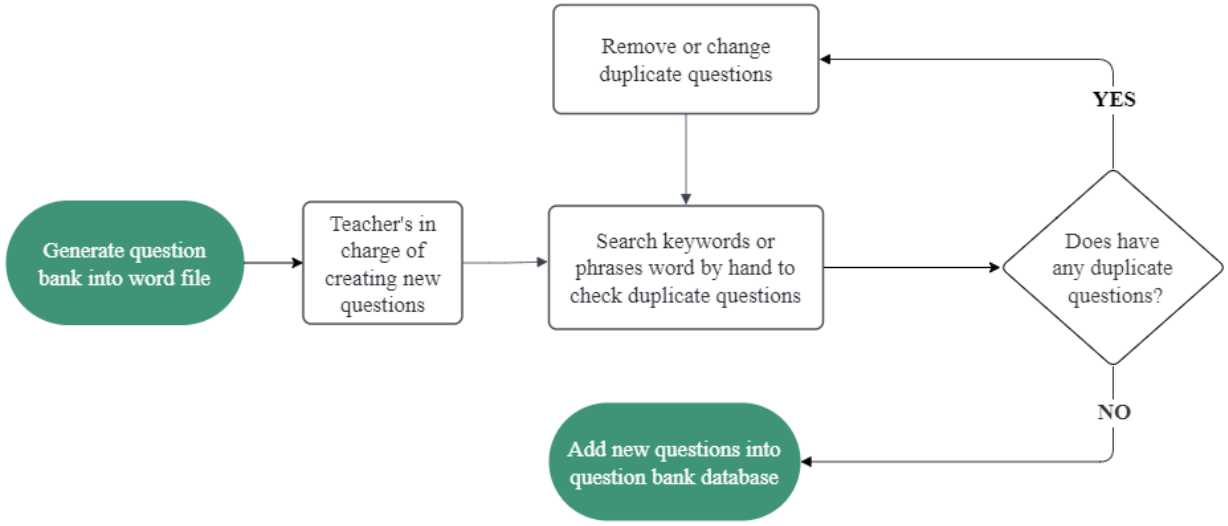

Figure 2. Case one: Adding a new question to the question bank

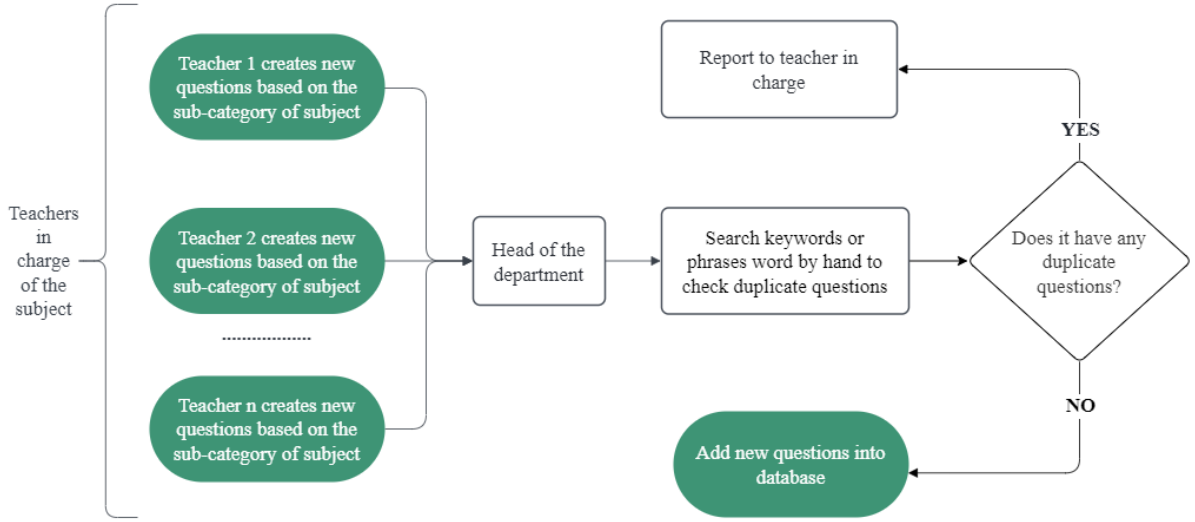

Figure 3. Case two: Adding a new question to the question bank

In general, as the number of questions in the question bank increases, it becomes more difficult to manage their content and impossible to avoid question duplication. Current research on multiple-choice tests solely concentrates on generating exam questions from the available question bank; still, it ignores the necessity of making sure that the questions' substance is unique across question bank. This can cause a massive consequence since it negatively affects the evaluated quality of the overall exam, for instance, two similar multiple-choice questions can appear in the same test. Along with this rising issue, due to the distinctive nature of the multiple-choice question format, the problem of MCQs content duplication is still a significant barrier. Aside from the difficulty of

multiple-choice question structure, due to the security concern and the limitation in exam application, questions are frequently stored in several formats, including a form of a document, an image, and a combination of text and image. Consequently, storing formats is one of the main issues needed to tackle during retrieving and comparing questions. In order to determine how similar the questions are, an additional pre-processing question to extract valuable text from the image must be added before extracting and comparing. With the development of computer vision, especially optical character recognition (OCR) models have been employed for extracting documents from images. The goal of this thesis is to address the issue of difficulties when adding a new question to the question bank. We conducted research on building a model for comparing image and text multiple-choice questions and deployed it on a web application, which made the enlarging question bank task more efficient.

1.2. Related Work

1.2.1. Question bank management

In the scope of question bank management, much research focuses on the creation, and assessment of question bank management systems for different areas of education, including medicine, computer science, engineering, and online learning environments. These studies emphasise the importance of efficient and effective question bank management systems for assistance in the evaluation of student learning outcomes. Purohit et al. [2] implemented an adaptive question bank management system that queries wisely from an extensive question database and generates a question paper based on the course curriculum by using the Concept Map tool. However, the redundancy of duplicate questions in the question bank typically has a significant effect on the system. Mia et al. [3] employed NLP technologies to construct a system that is suitable for storing and pre-processing multiple-choice questions in the Bangla language. For instance, questions stored in the database are categorised into three classes: cognitive class contains the title and four options; analytical class provides three or more options, and the candidate must choose one of the combinations of these options; higher ability class is a group of multiple-choice questions containing two or three cognitive class or analytical class questions. All questions are stored in a B+ tree to optimise the data access time, the key of the tree is constructed based on the question class and question number. After the pre-processing stage consists of stop-word removal, stemming, and tokenization the term frequency and inverse document frequency (TF-IDF) will be calculated. The term frequency value determines how often a phrase appears in a document, and the inverse document frequency value determines the significance of a word. The vector of a question is constructed based on a vector space model (VSM) which uses TF-IDF weights to calculate the word weighting for each text feature component. Finally, cosine similarity is applied to measure the similarity and the result is the ranking of similar scores. However, it is inappropriate to compare two questions that share the same content but different keywords by building a vector using the TF-IDF weights.

In the field of managing and optimising large-scale question datasets, Wang et al. [4] presented a question-answering model for Stack Overflow, a well-known community-based question-answer (CQA) platform. The model utilised Word2Vec to embed questions and applied deep learning techniques such as convolutional neural network (CNN), recurrent neural network (RNN), and long short-term memory (LSTM) to identify duplicate questions, the evaluation results indicated that the use of CNN and LSTM produced significant improvements compared to traditional machine learning approaches. Particularly, the outcome of the word-vector long short-term memory (WV-LSTM) model outperformed all other models. Li et al. [5] put forth an end-to-end system that can instantly locate questions that are similar to unanswered medical questions. The system initially collects certain metadata from a new question when it is uploaded on the website, like the question's intention and category. Such metadata is useful for highlighting the key phrases in the unsolved

question. Then, the system pulls out a select group of possible questions from the current question corpus using the key terms that are determined as important. Otherwise, the new question will be compared with every solved question in the corpus, which is time-consuming or even impossible when the size of the question corpus is large. The proposed system's effectiveness can be greatly increased by taking this step. The similarity score of the unsolved question with each question in the candidate set will be calculated based on question vector representations produced by a trained LSTM. All candidate questions are then ordered according to their corresponding similarity scores, and the top candidates with similarity scores greater than a predetermined threshold are taken into consideration as similar medical questions for the unanswered question. Nevertheless, these keyword-based algorithms have the drawback of just considering the information in the keywords and ignoring the comprehensive information.

1.2.2. Optical Character Recognition (OCR)

OCR refers to the process of automatically recognizing and converting written or typed text in an image or scanned document into machine-readable text. Related work in this field has been extensive, with numerous research studies and commercial solutions developed over the years. A critical area of related work in the OCR field has been the development of textual detection models. One popular model is TextBoxes proposed by Liao et al. [6], which integrates text region proposal and text classification into a single CNN and trains the network on various picture scales to handle texts of varying sizes, demonstrating fast and efficient results on SynthText, IC11, and IC13 datasets. Zhou et al. [7] present a method for scene text detection called Efficient and Accurate Scene Text detection (EAST) that uses a fully-convolutional neural network adapted for dense word or text line predictions at the pixel level. In 2019, Baek et al. [8] presented a highly efficient text detection model called Character-Region Awareness For Text detection (CRAFT) that is suitable for real-time applications and able to detect text in complex or low contrast backgrounds. The CRAFT method has been shown to be highly efficient, making it suitable for real-time text detection applications. In experiments, the result showed superior performance to other popular text detection methods, including EAST and TextBoxes, on the ICDAR 2013 and ICDAR 2015 datasets. Another important component of OCR is the text recognition model. Jaderberget et al. [9] present the model using a CNN to recognize text in images. The model has shown to outperform existing state-of-the-art text recognition algorithms in terms of accuracy and robustness. In 2015, an end-to-end trainable deep neural network to minimise the recognition error proposed by Shi et al [10], the convolutional recurrent neural network (CRNN) architecture has been used for the model consisting of multiple convolutional, recurrent layers, and transcription layers from bottom to top. This architecture is trained in an end-to-end manner, where the parameters of both the CNN and RNN components are learned simultaneously from the input images and their corresponding text annotations. The model has been demonstrated to attain high accuracy on the IC 2013 and IIIT 5K-Words datasets by combining convolutional and recurrent layers. Overall, it can be stated that the capabilities of OCR have greatly advanced and can now successfully recognise text in various font styles, orientations, and languages, even in complex scenes with noise and low resolution. OCR can serve as a preliminary processing stage where it can transform image-based sentences into a text format for further processing and comparison.

1.2.3. Two-sentence comparison

Related to the evaluation of two-sentence comparison, Gokul et al. [11] propose a method utilising cosine similarity to find the similarity score between two Malayalam sentences and determine whether two input sentences are two paraphrases or not. First, the input sentence is split into separate words like $S_1 = \{W_1,\ W_2, W_3, \dots, W_n\}$ where W_n represents the last word of the input sentence. Then, the study apply Malayalam WordNet and POS tagging (Part-of-Speech tagging) for determining the functions of each word to choose a standard set of tags (N, V, Prep, Det, Adj,...). After that, cosine similarity is employed to calculate similarity scores and determine whether they

have the same meaning or not. The study obtained an accuracy of 0.8 in test data of 900 sentence pairs of FIRE 2016 Malayalam corpus. In 2018, Dhar et al. [12] suggested using TF-IDF to represent the text as a collection of word frequency vectors and computing text similarity using cosine similarity. The study based on Bangla text documents is normalised and utilised as the experiment's inputs after being received from various online web sources. However, this approach disregards the semantic details connected to the text's phrases.

Devlin [13] introduced Bidirectional Encoder Representations from Transformers (BERT) employing a cross-encoder: The transformer network receives two sentences, and it predicts the target value. Due to a large number of combinations, this approach is unsuitable for many pair regression tasks. This led to the time of searching similar questions issue when this process is time-consuming and inappropriate for a large question bank. To address the time-consuming issue of BERT, Reimers [14] introduced Sentence-BERT (SBERT) which adds one of three pooling strategies such as CLS-token, MAX-strategy, and MEAN-strategy to the output of BERT. After that, the study employed a siamese network structure [15] to fine-tune and construct semantically meaningful sentence embeddings for the STS (semantic textual similarity) [16] task. Then, applying a similarity measure method such as cosine similarity or Euclidean distance to calculate the semantic textual similarity between two sentence embeddings. These similarity measures can be performed extremely efficiently on modern hardware, allowing SBERT to be used for semantic similarity search and clustering. The study also emphasised that SBERT uses siamese network structures that share their weights and uses optimised index structures, so the number of operations decreased significantly compared to BERT. Thanh et al. [17] conducted research on different SBERT strategies for STS and they suggested that the triplet loss function is the most effective function for training and fine-tuning SBERT. Therefore, SBERT should be used to identify similar questions in order to increase the effectiveness of managing multiple-choice questions in question bank.

1.3. Contribution

This study aims to improve question bank management at FPT University by addressing the problem of question duplication and purpose a new method for lecturers when enlarging the question bank. In this thesis, we primarily focus on theoretical research to improve the effectiveness of a model that can identify duplicate questions, whether they are in text or image form. The research improves the accuracy and the time consumption of similarity question search by using SBERT for encoding the question. Then, similarity score will be calculated using the similarity measurement method. For questions containing images, the OCR model which contains two components: CRAFT to identify and locate text regions, and CRNN for text recognition. That framework will be applied to extract text from images and then concatenate them with the questions. By combining SBERT and OCR, our goal is to build an end-to-end machine learning architecture that represents questions in text or image form into a vector to detect duplicate multiple-choice questions in the question bank containing images in minimal time. A new dataset has been built for the purpose of model evaluation, which will be covered in the following section. According to the architecture's pipeline, the model processes both the question and the image by first converting the image into text and then combining it with the text part. This combined input runs through the SBERT model to obtain embeddings and compare the similarity with existing questions in the database. Finally, for the practical perspective, we utilise Django as a framework to deploy the model on a question bank web application. The website allows the storage of numerous questions in image and text format, along with two specific role permissions (manager, teacher), making the process of import questions and managing question bank in the FPT University more efficient.

2. Methodology

2.1. Overview pipeline

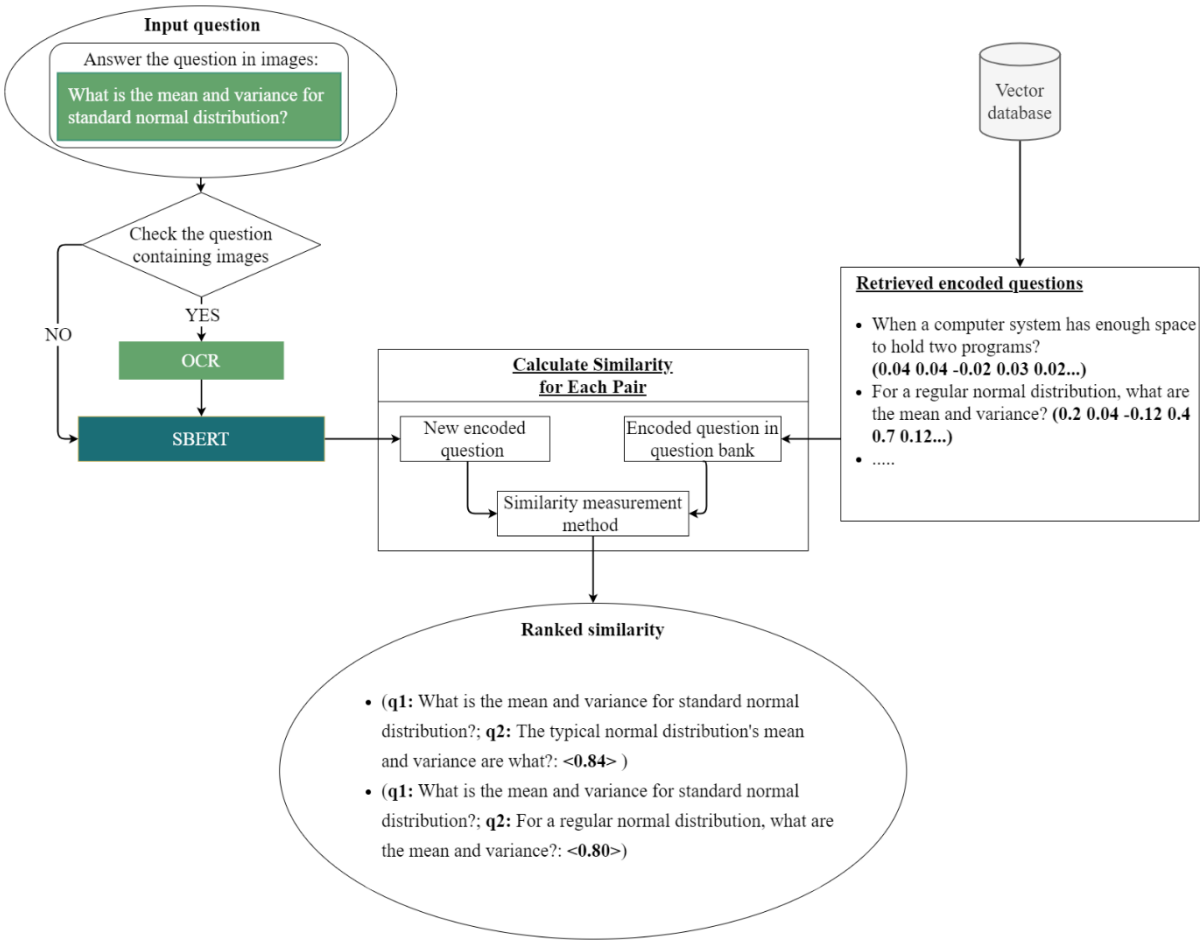

Figure 4. Question bank architecture

To provide a clearer explanation of the proposed system, a high-level overview is introduced. Figure 4 describes the flow of the whole architecture that is designed to detect duplicates of multiple-choice questions. When a new question is added, the presence of images is checked, and then the OCR model is applied to extract relevant information. Then the model joins the extracted information with the textual question. In many cases, the image questions also include options, so the text part and options will be merged to improve accuracy. Finally, all the compared questions will be sorted in descending order according to similarity scores, and all the top questions that exceed the threshold are marked as duplicate questions. The detail about each component is described in the following sections.

2.2. OCR

The OCR model used to extract information from images is the EasyOCR [18] framework. It is designed to highly accurate and, easy to use even for low-quality images. The OCR task is divided into several stages, including image pre-processing for noise reduction, character segmentation, and character recognition. The detection model - CRAFT, which is introduced by Baek et al. [8], is applied to localize each individual character in the image. After detecting the character region, the result is the input of the recognition model which is a CRNN model proposed

by Shi et al. [10]. CRNN is an end-to-end trainable neural network, takes advantage of CNN and RNN to achieve high performance on image-based sequence recognition.

2.2.1. Text detection with CRAFT

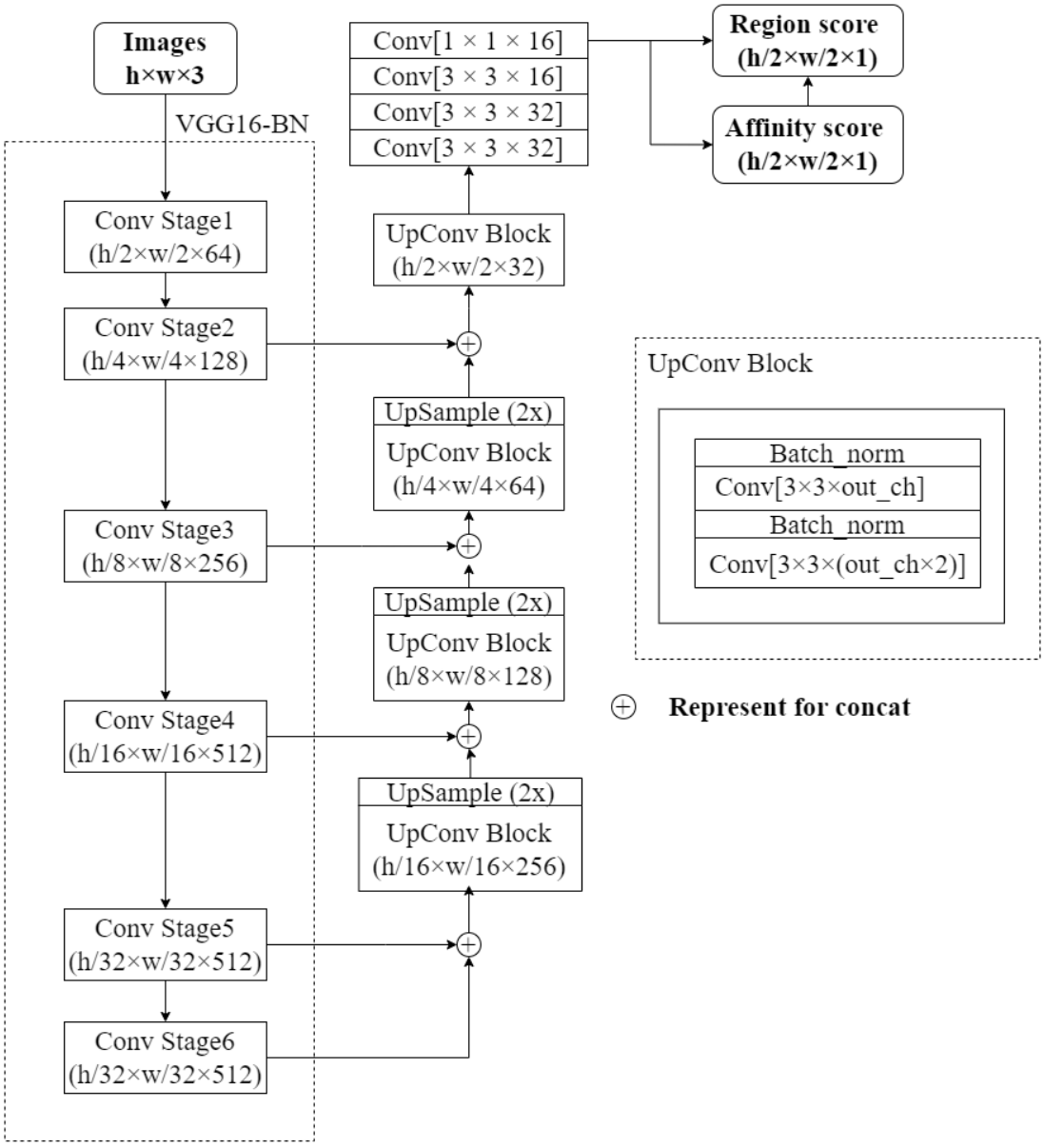

Figure 5. CRAFT architecture [8]

Scene text detection is a task to detect text regions in the complex background and mark them with bounding boxes. The main objective of CRAFT is to localize the character region in the image and group detected characters to a text instance. The backbone of CRAFT's design is a fully convolutional network architecture based on VGG-16. CRAFT architecture primarily uses VGG-16 to extract features from the input of the network and encode them into a specific feature representation. The decoding segment of the CRAFT network is similar to UNet which skips connections that aggregate low-level features (Figure 5). Moreover, to surmount the missing character-level annotations datasets, the CRAFT model is trained using the weakly supervised learning method. The result of the models is two scores for each character: the region score which represents the probability that the given pixel is the centre of the character and the affinity score represents the centre probability of the space between adjacent characters.

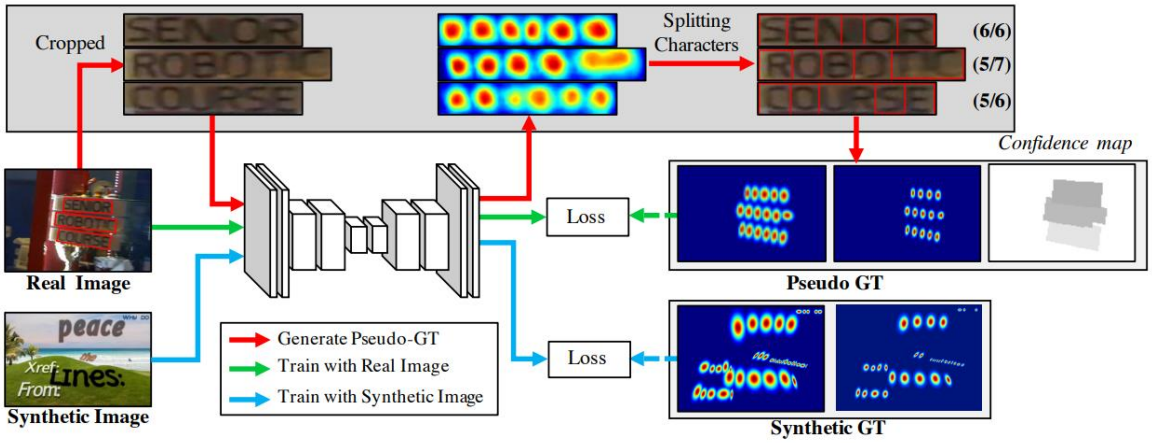

Figure 6. Training stream in CRAFT [8]

CRAFT employed Gaussian heatmap representation to learn both the region score and the affinity score. Typically, perspective projections deform the character bounding boxes on an image so three steps have been deployed to approximate and generate the ground truth for the region score and the affinity score: 1) prepare a 2-dimensional isotropic Gaussian map; 2) compute perspective transform between the Gaussian map region and each character box; 3) warp the Gaussian map to the box area [8]. For the affinity score, the model draws diagonal lines connecting opposite vertices of the character box, creating 2 triangles - upper and lower, and the affinity box is created with the vertex central point of four triangles from two adjacent character boxes. Unlike synthetic datasets that have character-level annotations, real images usually have word-level annotations. So, the CRAFT creates a character box from each word box. When a real image with word-level annotations is provided, the transient model (under training) will predict the character region score of the words that have been clipped to create character-level bounding boxes. The steps for training real images: first cut the word boxes, then use the temporary model to predict the region score (heat map), next the watershed [19] algorithm is applied to separate the characters to create the character bounding box, finally the character boxes' coordinates are changed back into the original image coordinates. Figure 6 illustrates the training phase of CRAFT for both real images and synthetic images in a weakly-supervised style.

2.2.2. Scene Text Recognition

After detecting text by using CRAFT, the input for CRNN is the bounding boxes and the objective is to recognise the letter inside the bounding boxes. From bottom to top, the network architecture of CRNN is made up of three parts: convolutional layers, recurrent layers, and a transcription layer (Figure 7). The convolutional layer of CRNN is based on taking convolutional and max-pooling layers from a standard CNN model and the fully-connected layer is removed. So, a sequential feature representation is extracted from an input picture using such a component. Before being fed into the network, the images are scaled to have the same height. After that, the component of convolutional layers extracts the sequence of feature vectors from the feature maps. The feature vectors are generated by taking a column of feature maps from left to right, the width of each column is a single pixel. Therefore, each column of the feature map, which is called the receptive field of the original image, corresponds to a rectangle region.

Next, the feature sequence which is made up of feature vectors is the input of the recurrent layer. The recurrent layer has the advantage of capturing contextual information within a sequence. RNN uses backward propagation enabling the model to train the convolutional and recurrent layers together in a single network. Using RNN, the input sequence can have a flexible length, but the recurrent layer suffers from a vanishing gradient and causes the issue of limiting the range of

context it can store. So, the study employed a bidirectional LSTM consisting of two components, one forward and one backward. Each part is made up of three multiplicative gates: the input, output, and forget gates. Compared to a one-directional LSTM structure, the bi-directional allows for more levels of abstraction and has significantly improved performance.

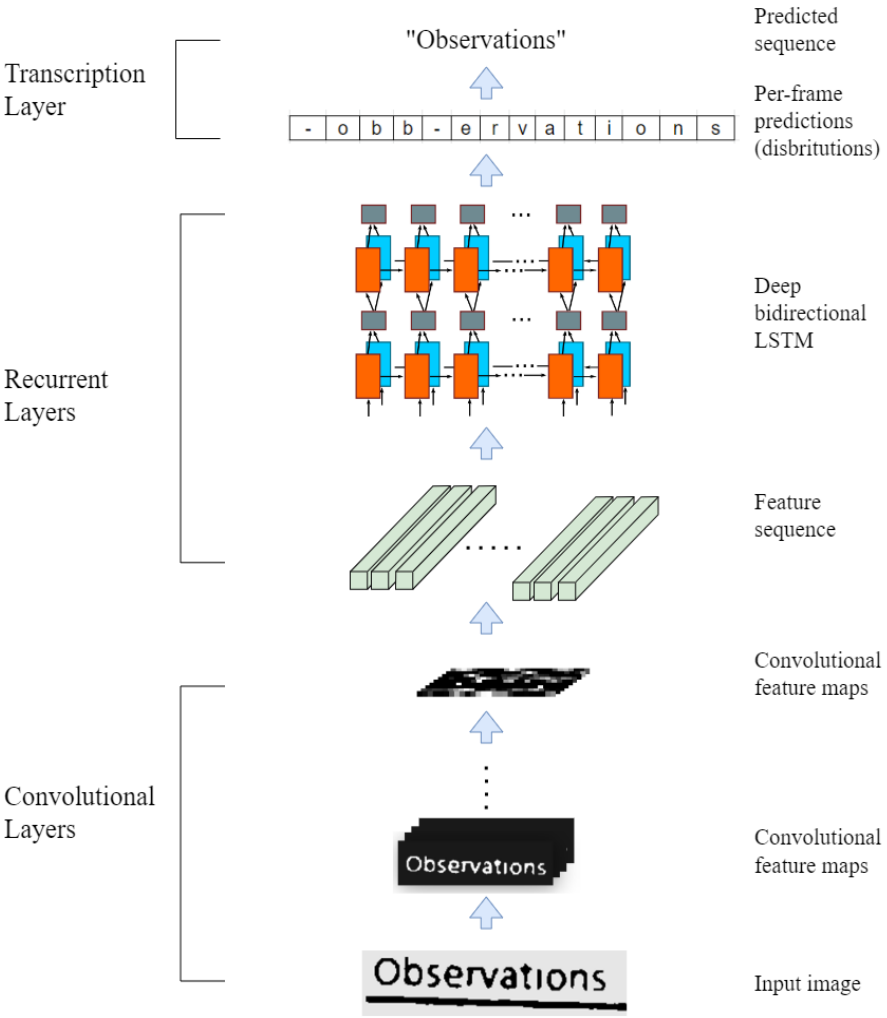

Figure 7. The CRNN network architecture [10]

Transcription is the process of converting frame predictions made by the RNN to a label sequence. Mathematically, the goal of transcription is to identify the label sequence that has the highest probability given the per-frame predictions. Two types of transcription used in practice are lexicon-free and lexicon-based transcriptions. A lexicon is a collection of label sequences to which prediction is limited, such as a dictionary used for spell checking. By using the lexicon-free mode, predictions are made without any lexicon. But in the lexicon-based mode, the highest probability label sequence is used to make predictions. For the training phase, in the formular (1) the CRNN minimizes the negative log-likelihood of the conditional probability of ground truth:

$$L = -\sum_{I_i, l_i \epsilon \chi} \log p(l_i | y_i), \quad (1)$$

where $\chi = \{I_i, l_i\}_i$ denotes the training dataset, I_i denotes the training images, l_i denotes the ground truth label sequence, and y_i is the sequence produced by the recurrent and convolutional layers from image I_i.

2.3. BERT

BERT stands for Bidirectional Encoder Representation from Transformer and is constructed to pre-trained deep bidirectional vector representations from the unlabeled text by combining left and right context of a sentence. BERT has the unique ability to balance the context in both the left and right directions in a sentence based on the transformer technique. BERT model architecture is a multi-layer bidirectional transformer based on the encoding process of the transformer. BERT stacks the encoder to produce word embeddings. Let L denote the number of stacked encoders, H denotes the number of hidden sizes, A is the number of self-attention heads, there are two main models, which are $BERT_{BASE}$ with the parameter L = 12, H = 768, A = 12, the total parameter = 110 million and $BERT_{LARGE}$ with the parameter, L = 24, H = 1024, A = 16, the total parameter = 340 million (Figure 8). Natural processing tasks are neural machine translation, question answering, sentiment analysis, and question answering necessary comprehension of human language. So BERT has two phases, pre-training for understanding language and then fine-tuning depending on a specific task.

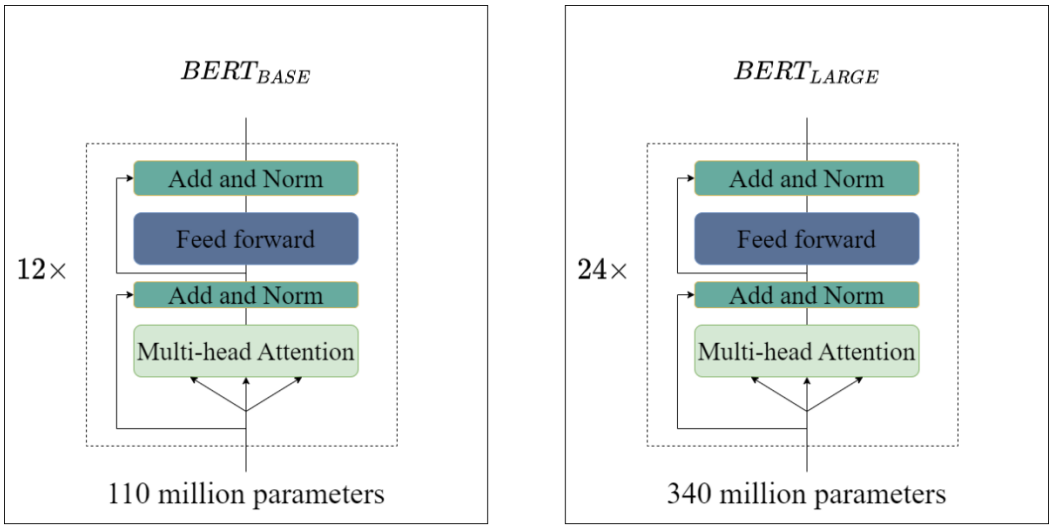

Figure 8. BERT architecture

2.3.1. Transformer in BERT

BERT uses a stack of 12 identical encoder layers [13] for $BERT_{BASE}$ and 24 encoder layers for $BERT_{LARGE}$, each layer made-up of two sub-layers which are multi-head self-attention and position-wise fully connected feedforward layers. The residual connections are employed around two sublayers followed by a normalization layer. The output of each layer has a dimension of 512, which can be defined in the formular (2):

$$LayerNorm\big(x + Sublayer(x)\big), \tag{2}$$

where $LayerNorm()$ is layer normalisation, x is the input matrix, $Sublayer(x)$ is multi-head self-attention or feed-forward layer of matrix x.

First, the input embedding in the transformer works as a look-up table to convert a word to a vector representation. The embedding layer produces an embedding vector for each word in the sentence. For example, the word “I” can be represented in the formular (3):

$$I = [0.1;\ 0.54;\ 0.29\ldots]^T, \tag{3}$$

Next, the positional encoding injects positional information of a word into word embeddings so that the model can use the order of sequence. The progress of positional encoding is applied after the input embedding at the encoder stacks. Let t be the desired position in an input sentence $\overrightarrow{p_t} \epsilon R^d$ be its corresponding encoding and d be the dimensions of the model. The formular (4) defined the function of positional encoding:

$$\begin{cases} \overrightarrow{p_t}^{(i)} = \sin(w_k.t) \; if \; i = 2k \\ \overrightarrow{p_t}^{(i)} = \cos(w_k.t) \; if \; i = 2k+1 \end{cases} where \; w_k = 1/10000^{2i/d}, \quad (4)$$

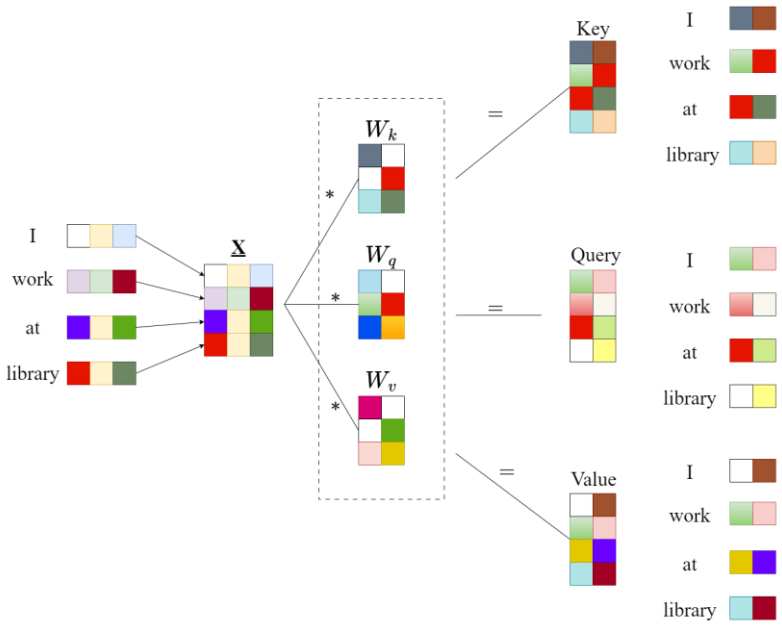

Figure 9. Self-attention

Self-attention (Figure 9) allows a model to associate and adjust the weights of each individual word to other words in the input. After the input embedding and positional encoding, the input of the attention layer is a matrix that has a size of $m \times n$, where m is the length of sentences and n is the dimension of the word embedding vector. The input matrix is multiplied with three weight matrixes W_k, W_q, and W_v which are learned through the training process. Three results are called key, query, and value. The attention score of each word pair (w_i, w_j) is calculated by using the dot-product of the query and the key divided by $\sqrt{d_k}$ where d_k is the dimension of the vector key. The higher attention score is, the more the association of the word w_i and the word w_j is. After that, the attention score has to go through the softmax operation to be normalised to be in the range of [0; 1]. Then, the attention score with softmax is multiplied by the value vector to obtain the final result. The goal of multiplying with a value vector is to preserve the value of the important word while fading out the irrelevant word by multiplying them with a small number. In actual implementation, the embedding vector of an individual word is stacked together to create one matrix for each sentence. The query, key, and value vectors of each word are also stacked to create three matrixes Q, K, and V and the attention score is calculated in the formular (5):

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V, \quad (5)$$

In the transformer architecture, to expand the ability to focus on different positions, the study employed multi-head attention [20] (Figure 10). The calculation is similar to self-attention, but in the multi-head attention layer, the model does self-attention many times with different weight matrices W_k, W_q, and W_v. Then it concatenates eight matrixes and multiplies by an additional weight matrix W_o to return the final matrix having the same dimension as the input matrix. The formula of multi-head attention (6) can be expressed as:

$$MultiHead(Q,K,V) = concatenate(head_0, head_1, head_2, \ldots, head_h)W_o$$
$$\text{where } head_i = Attention(Q_i, K_i, V_i) \tag{6}$$

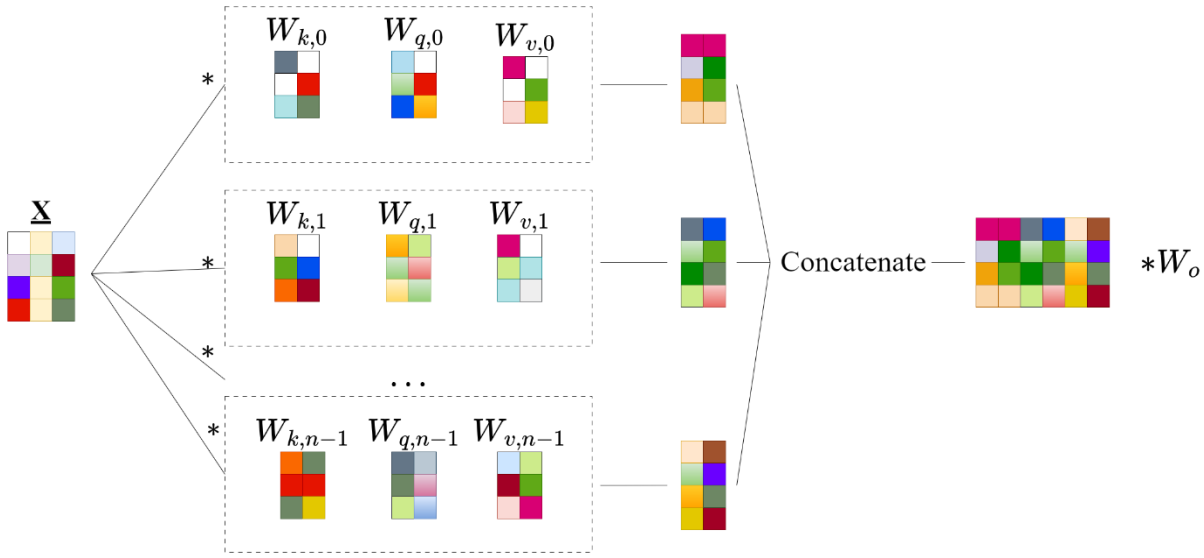

Figure 10. Multi-head attention

Finally, the fully connected feed-forward is a position-wise transformation that consists of two linear transformations and a ReLU activation function. The purpose is to process the output from one attention layer in a way to better fit the input for the next attention layer.

2.3.2. Pre-trained BERT

The goal of pretraining is to make BERT understand language and context. BERT is trained on the Wikipedia dataset with 2.5 billion words and Google's BooksCorpus with 800 million words [21]. Two unsupervised tasks for simultaneous training are the masked language model and next-sentence prediction.The masked language model enables BERT to bidirectionally learn from text. The task is hiding a word (masking) in a sentence and forcing BERT to use the words on either side of the sentence to predict the masked word. A random of 15% tokenized words is hidden during the training phase. NSP (Next Sentence Prediction) is used to help BERT learn about relationships between sentences by predicting whether a given sentence follows the previous sentence. For constructing input for pre-training, the beginning of the sentence will be denoted with the token [CLS] and separated by the pair sentence with the token [SEP]. For example, the pair "[CLS] I worked at the library. [SEP] I borrowed a new book" is related to each other and has the label "IsNext". In contrast, the pair "I worked at the library. The coffee tastes like earth" has the label "NotNext". In the training phase, 50% of related sentence pairs are mixed with 50% of random sentence pairs to increase the BERT accuracy.

2.3.3. Fine-tuning

After pre-training, BERT can be fine-tuned for a specific task. At the input, the data of the task simply feed into BERT and fine-tune all the parameters. The input during fine-tuning can be sentence pairs in paraphrasing, question-passage pairs in question answering, and a degenerate text pair in text classification or sequence tagging. The [CLS] representation is fed into an output layer for classification, such as entailment or sentiment analysis, while the token representations are fed into an output layer for token-level tasks, such as sequence tagging or question answering.

2.4. SBERT

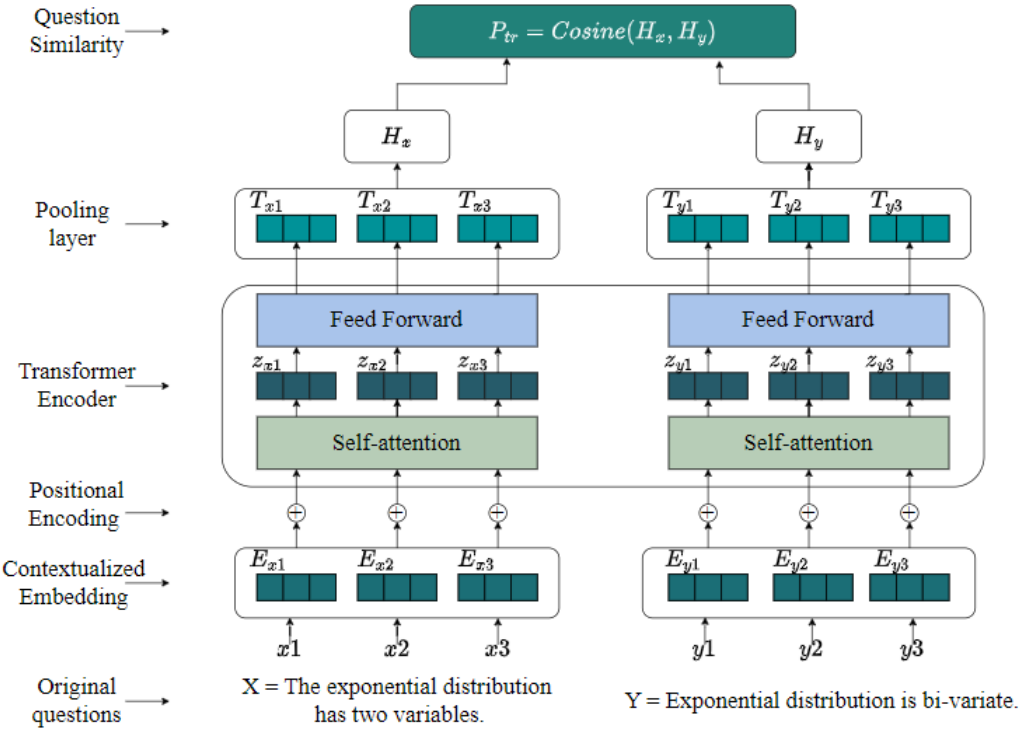

Figure 11. SBERT architecture

Most of the previous research in STS (semantic textual similarity) is not appropriate for a massive range of pair regression tasks. The drawbacks of BERT are independent sentence embeddings cannot be computed directly so it is challenging to derive sentence embeddings from BERT. One typical way to get around this problem is to push single sentences through BERT, and then create a fixed-sized vector by averaging the results (much like average word embeddings), or by utilizing the result of the first token (the CLS (classification)] token) [22]. However, the produced sentence embeddings are quite bad and cause high computational costs and time-consuming search time. To bypass this issue, we used SBERT (Figure 11), which adds a pooling layer to the BERT model's output to create fixed-sized sentence embeddings. Three pooling layers can be used as CLS-token, computing the mean of all output vectors (MEAN-strategy) and computing a max of the output vectors (MAX-strategy). To construct semantically relevant phrase embeddings and to update the weights, a siamese network is applied to fine-tuned in SBERT. There are several loss functions that can be used, and this study focuses on three loss functions: triplet loss, cross-entropy loss, and multiple negative loss functions. The approach is efficient in terms of search time while preserving the accuracy of BERT on STS tasks [17].

2.4.1. SBERT with triplet loss function

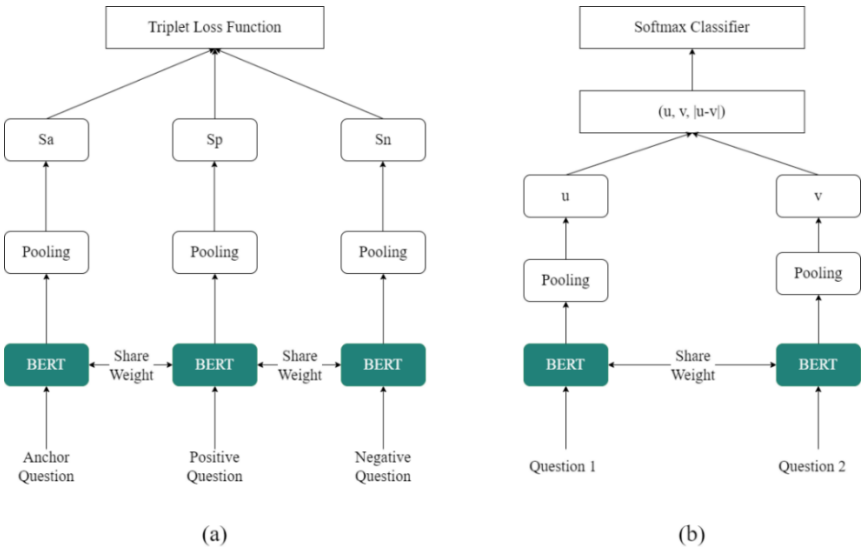

Figure 12. SBERT with triplet loss function (a), cross-entropy loss function (b) [14]

The triplet loss function [23] in SBERT (Figure 12(a)) employs the anchor question, the positive question, and the negative question. The triplet loss function tries to reduce the distance of the anchor question with positive question embedding and increase the distance with negative question embedding simultaneously. The desired embedding is obtained when the distance of the anchor-positive gets longer than the anchor-negative equals a margin m. The loss function can be defined in the formular (7):

$$L = \sum_{i=1}^{N} \max\left(\left||s_a - s_p|\right| - \left||s_a - s_n|\right| + m, 0\right), \quad (7)$$

with s_a, s_p, s_n are the sentence embeddings for anchor, positive, and negative questions respectively; $||\dots||$ represents the distance metric; margin m ensures that s_p is closer to s_a than s_n.

2.4.2. SBERT with cross-entropy loss function

The cross-entropy loss function is applied by adding a softmax layer and a cross-entropy loss to learn weight and directly predict a label. The structure of SBERT with cross-entropy loss function [14] is shown in figure 12(b). After two sentence embeddings u and v is obtained, the architecture concatenates two sentences embedding with the element wise difference $|u - v|$ then multiply with the trainable weight $W_t \in \mathbb{R}^{3n \times k}$ where k is number of labels and n is the dimension of the sentence embeddings (formular (8)):

$$L = softmax\left(W_t(u, v, |u - v|)\right), \quad (8)$$

2.4.3. SBERT with multiple negatives loss function

SBERT utilising a multiple negatives loss function [24] (Figure 13) is used when the training set only has positive pairs. The function focuses on balancing the distance of a positive question over multiple negative questions simultaneously. The loss function defined in the formular (9):

$$L = -\frac{1}{K}\sum_{i=1}^{K}[S(x_i, y_i) - log \sum_{j=1}^{K} e^{S(x_i, y_j)}] \; where \; (i \neq j) \quad (9)$$

where (x_i, y_i) represent the pair of anchor question and positive questions, (x_i, y_j) represent the pair of anchor question and negative question and $i \neq j$. $S(x_i, y_i)$ is a distance of two questions.

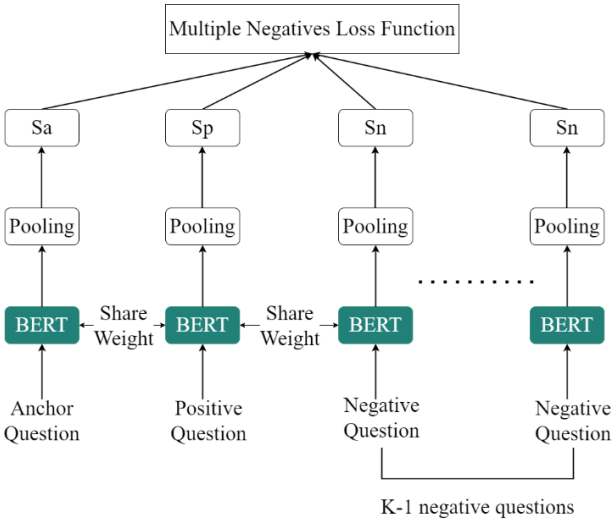

Figure 13. SBERT with multiple-negatives loss function [17]

2.5. Similarity measurement method

We utilise similarity measurement methods to determine the similarity score between each vector representation of the input question computed by SBERT and each encoded question from the database. Three proposed methods are Cosine-similarity, Euclid distance, and Spearman rank correlation coefficient.

2.5.1. Cosine-similarity

Cosine similarity is a metric for calculating the similarity of two non-zero vectors specified in an inner product space. In other words, it is the dot product of the vectors divided by the product of their lengths. Cosine similarity is the cosine of the angle between two vectors and the result is constrained to the range [0, 1]. The formula of cosine-similarity (10) can be defined as:

$$sim(\vec{A},\vec{B}) = cos(\theta) = \frac{A.B}{||A||.||B||} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}} \quad (10)$$

where $\vec{A}$ and $\vec{B}$ are two sentence embedding vectors for multiple-choice questions A and B respectively.

2.5.2. Euclid distance

Euclidean distance (formular (11)) is the square root of the sum of squared differences between corresponding elements of the two vectors. In a multidimensional space, the closer two-word vectors are to one another, the more probable it is that their meanings are similar:

$$sim(\vec{A},\vec{B}) = d(\vec{A},\vec{B}) = \sqrt{\sum_{i=1}^{n}(A_i - B_i)^2} \quad (11)$$

2.5.3. Spearman's rank correlation coefficient

The Spearman's rank correlation coefficient is a nonparametric measurement method of the correlation between two variables. The value is constrained in the range of [-1,1] which means the closer a value is to 1, the more similar the two variables are. The formula (12) represents the similarity score between vectors $\vec{A}$ and $\vec{B}$ using Spearman's rank correlation coefficient:

$$sim(\vec{A}, \vec{B}) = \rho = 1 - \frac{6\sum_{i=1}^{n}(A_i - B_i)^2}{n(n-1)^2} \quad (12)$$

where A_i, B_i is the value of two sentence embedding vectors $\vec{A}$ and $\vec{B}$.

2.6. Evaluation metric

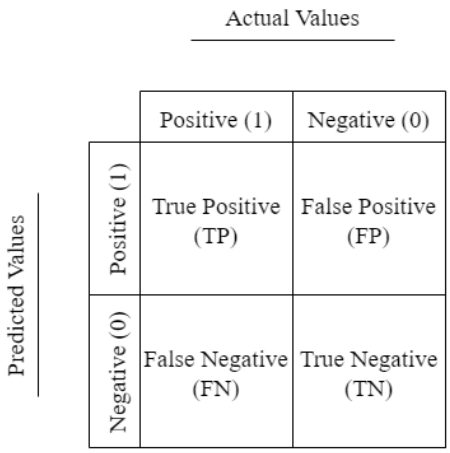

Figure 14. The confusion matrix

The F1 score is a machine learning assessment statistic that focuses on a model's performance inside each class rather than its overall performance to evaluate a model's predictive ability. F1 score combines two competing metric precision, and recall scores of a model. Precision and recall are constructed based on a confusion matrix consisting of four parts (Figure 14): the number of samples that were correctly predicted as "positive" (true positives; TP), the number of samples that were incorrectly predicted as "positive" (false positives; FP), the number of samples that were correctly predicted as "negative" (true negatives; TN), and the number of samples that were incorrectly predicted as "negative" (false positives; FN) (Figure 14). F1 score is useful when the classes are imbalanced, especially in finding duplicate question cases, when the number of samples duplicates class is significantly smaller than non-duplicates ones. The formula (13) and (14) represent precision, recall, and F1 score:

$$Prescision = \frac{TP}{TP + FP} \text{ and } Recall = \frac{TP}{TP + FN} \quad (13)$$

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall} \quad (14)$$

3. Experiment

3.1. Data collection

Table 1. Data statistic

Information	Value
Corpus size	312811
Vocabulary size	10306
Average length of question	252

We collected data from 1242 multiple-choice questions based on the FPT university question bank, the datasets include two types: English which indicates the questions refer to studying skills, and common situations in university; Math implies that the problems are probability and statistics, function definitions, etc. The number of English questions and Math questions are 682 and 560 respectively. Table 1 shows the general information of the data statistics. 398 Math questions are in the form of an image or a combination of text and images where the images represent figures, functions, and special characters. Figure 15 shows an illustration of a few multiple-choice questions from the dataset.

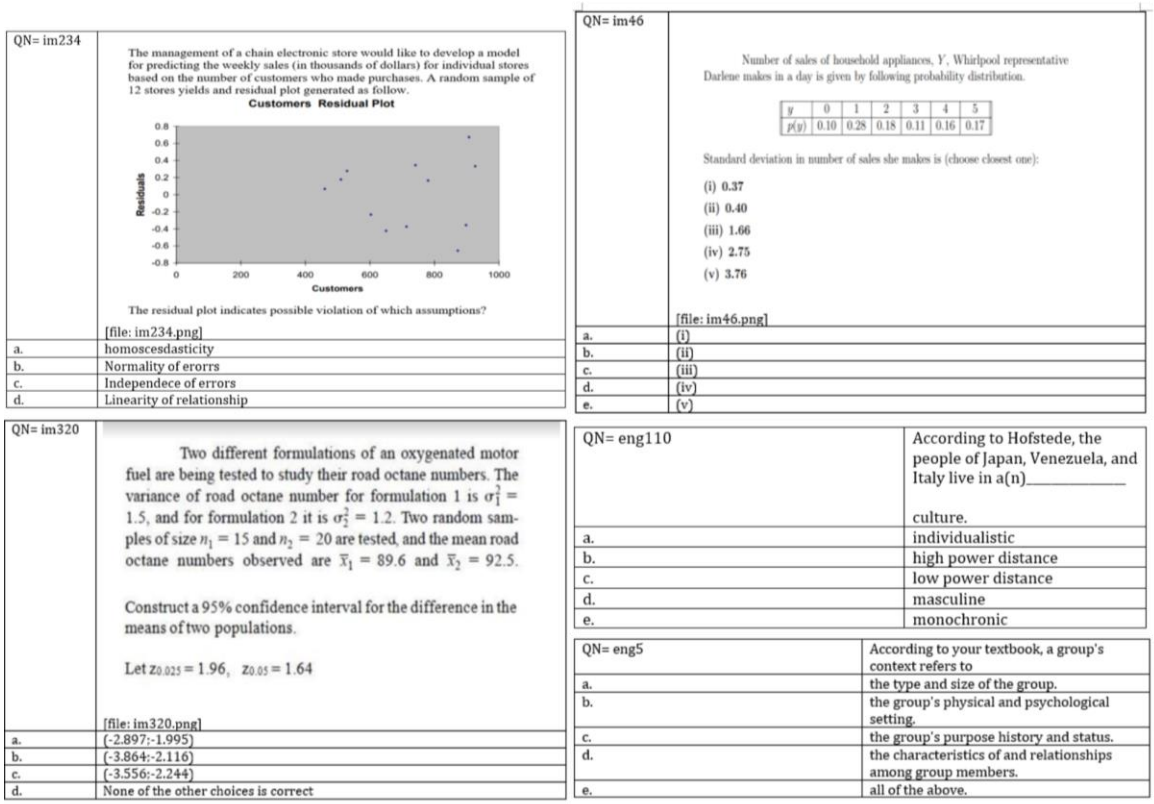

Figure 15. Examples of the MCQs in the database

To better understand the question domain, we use nltk (Natural Language Toolkit) to display word frequency that appears commonly in the dataset. Tables 2 and 3 represent the top 5 common words for each question type. In the English type, the domain of words is mostly based on the request for

candidates to choose an option when answering the multiple-choice questions, while the domain in the Math type represents the words in the probability and statistics fields.

Table 2. Word frequency for English question types

Word	Value
group	244
following	143
members	132
check	107
option	99

Table 3. Word frequency for Math question types

Word	Value
probability	216
random	193
none	187
distribution	180
mean	176

3.2. Implementation

Machine learning is implemented in PyTorch. The experiments were conducted on a GPU Nvidia Tesla T4 15Gb GPU. The encoding model uses two pre-trained sentence transformations models SBERT. The all-mpnet-base-v2 model is trained on a 1 billion sentence pairs dataset with a sequence length limited to 128 tokens, batch size of 1024, and learning rate of 2e−5 with 100,000 steps. The multi-qa-mpnet-base-dot-v1 model is concatenated from total 215 million question-answer pairs dataset to trained and fine-tuned. We fine-tuned SBERT for detecting duplicate multiple-choice questions on our datasets with multiple-negative loss function, 5 epochs, and batch size of 8. The fine-tuned SBERT was compared with other models with different similarity measurement methods. For deciding whether the pair multiple choices are duplicated or not, we set a threshold after calculating the similarity score. If the score crosses the threshold the pair is considered a duplicate.

3.3. Result & Analysis

We apply two pre-trained models SBERT [25] multi-qa-mpnet-base-dot-v1, all-mpnet-base-v2 before and after fine-tuning, with softmax loss function, triplet loss function, and multiple-negative loss function following by using three measurement methods for computing similarity score. The result is listed in table 4. In three loss functions, both multi-qa-mpnet-base-dot-v1 and all-mpnet-base-v2 models perform well on the Cosine and Spearman method, but when it comes to the Euclidean distance method, the multi-qa-mpnet-base-dot-v1 model yields poor results. After fine-tuning with softmax loss and triplet loss, the obtained result is significantly lower than pre-trained models. In the first experiment with softmax loss, we obtained an F1 score of around 0.77 for two models and 0.44 for multi-qa-mpnet-base-dot-v1 model with the Euclid measurement method. The F1 score is significantly improved by around 0.7 for both models with triplet loss function since it required anchor question, positive question, and negative question for training simultaneously so the models better understand the context of multiple-choice questions. When applied multiple negative loss function, both models achieved outstanding F1 scores around 0.94, and the all-mpnet-base-v2 model with Spearman measurement methods achieved the highest result with a 0.95 F1 score. In our dataset, the number of positive questions is lower than the number of negative questions, SBERT with a multi-negative loss function outperforms other approaches in terms of F1 score because it better manages skewed data.

Table 4. SBERT with F1 Score

Model	Cosine	Spearman	Euclid
multi-qa-mpnet-base-dot-v1	0.89	0.9	0.12
all-mpnet-base-v2	0.89	0.9	0.9
multi-qa-mpnet-base-dot-v1 fine-tune softmax loss	0.77	0.77	0.44
all-mpnet-base-v2 fine-tune softmax loss	0.75	0.76	0.76
multi-qa-mpnet-base-dot-v1 fine-tune triplet loss	0.88	0.89	0.25
all-mpnet-base-v2 fine-tune triplet loss	0.80	0.80	0.80
multi-qa-mpnet-base-dot-v1 fine-tune multiple negative loss	0.94	0.92	0.15
all-mpnet-base-v2 fine-tune multiple negative loss	**0.94**	**0.95**	**0.93**

4. Web Application Deployment

Based on the result after the experiment phase, to tackle the problem of time-consuming and improve the effectiveness when enlarging the question bank, we propose a web-based called Queslet. The web application has a user-friendly application that uses the Django framework to manage the question bank and supports the professor in the adding question phase. The architecture applied Pinecone and PostgreSQL as a database for retrieving and storing multiple-choice questions.

4.1. Overall question bank architecture

4.1.1. Database ERD

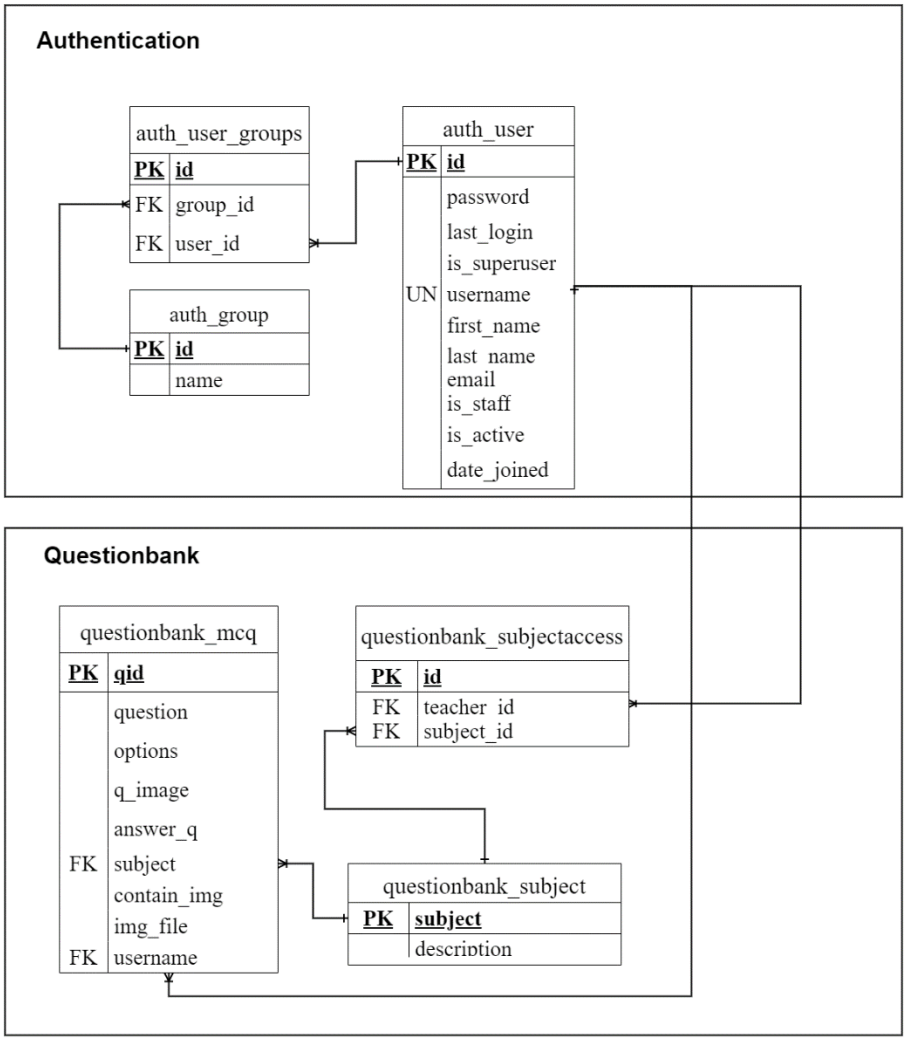

Figure 16. The schema in PostgreSQL

Overall, to better manage the question bank, the tables auth_group and auth_user_group divide users into two main groups with different roles including teacher and manager. The table auth_user contains data of all users with the unique filed username for them to login to the system. Each user must be assigned to access the subject by the manager and store it in the questionbank_subjectaccess table to interact with specific subject in the question bank. The information of the subject is stored in the table questionbank_subject, currently there are two main subjects which are Math and English corresponding to the dataset that built for the SBERT model.

The question is designed based on multiple-choice questions in FPT university that contains question, options, answer, and image with a corresponding subject (Figure 16).

4.1.2. Use case specifications

The first role is the manager which represents the head of the subject. The manager has permission to import, delete, view, search, update, and export questions in the database. The manager can view all subjects that exist in the question bank. Also, they can appoint teachers to access appropriate subjects (Figure 17 (b)). The second group is the teacher who can view and search assigned subjects (Figure 17 (a)).

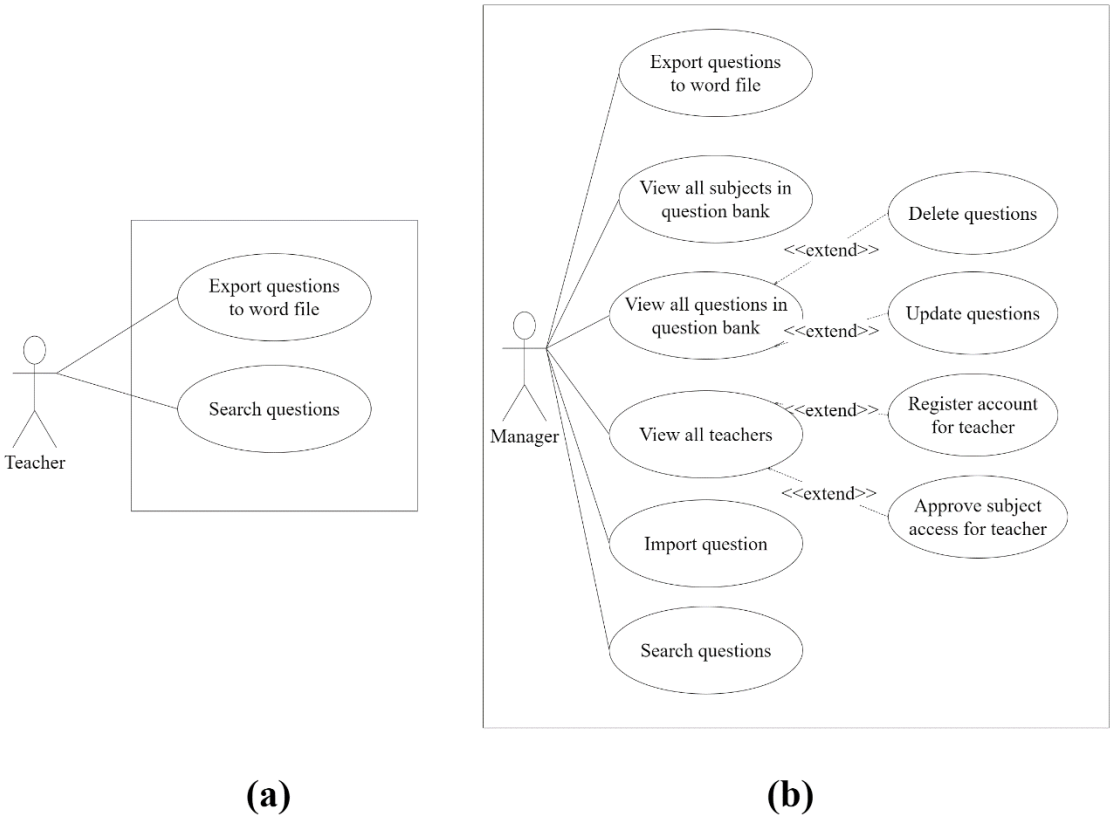

Figure 17. Use case of teacher role (a), Use case of manager role (b)

4.2. Pinecone

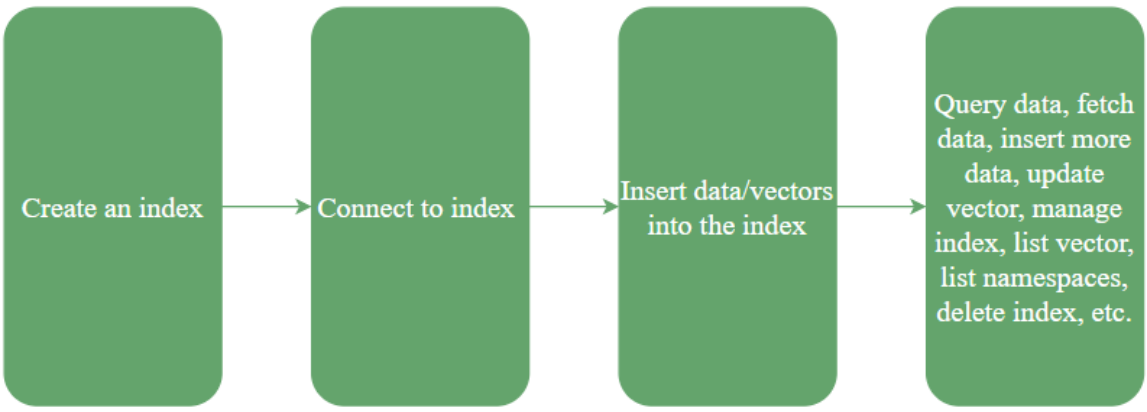

Figure 18. Workflow in Pinecone

To tackle the problem of retrieving and searching questions for comparison in the large question bank, the Pinecone vector database is employed to store encoded questions. Pinecone is a vector database, which has features like CRUD operations, metadata filtering, horizontal scaling, indexes, and saves vector embeddings for fast retrieval and similarity searches. Pinecone performs semantic search as opposed to employing keyword lookups, focusing on the meaning of the search question. The meaning of a specific word, sentence, paragraph, or lengthy document can be effectively encoded into a data structure by applied vector embedding using pre-trained neural networks on vast collections of text data. Semantic search better comprehends the content and context of multiple-choice questions in order to produce more precise search results. Furthermore, to better assist the searching operation, Pinecone enables image similarity search which transforms image data into vector embeddings and allows users to search images through keywords.

The workflow of Pinecone is illustrated in Figure 18. First, a Pinecone API key, the name of an index, and the number of dimensions for each vector are needed in order to create a Pinecone Index. In this study, a 768-dimensional vector for maps of each piece of text is employed. When connecting to the index, the Pinecone vector database requires vector embeddings so we use the trained SBERT model as a method for embedding multiple-choice questions when interacting with Pinecone after finishing creating and connecting to the index.

4.3. Process adding new questions

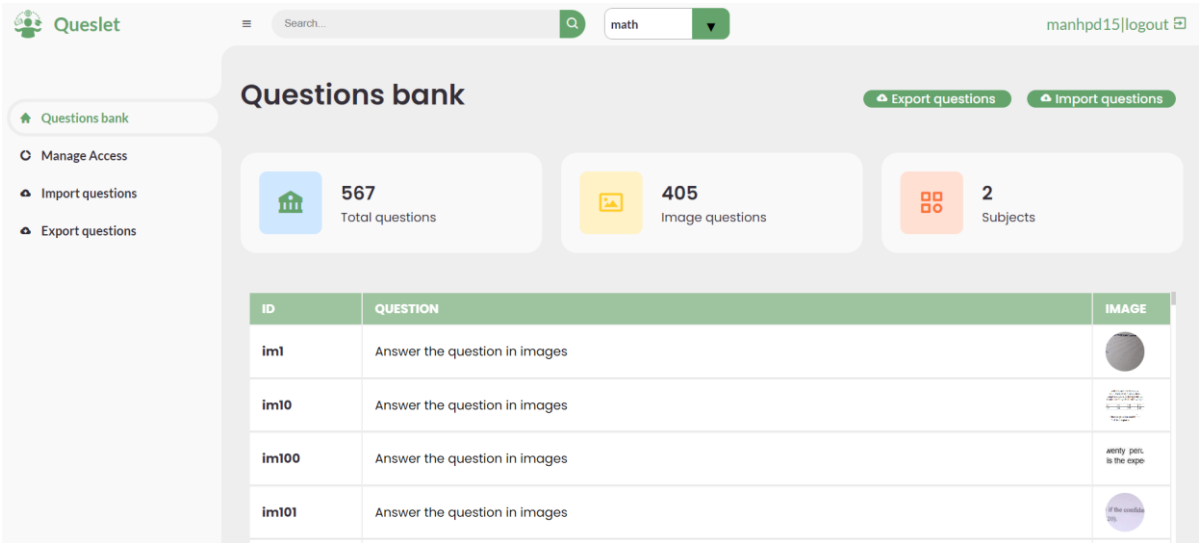

Figure 19. Website homepage with Manager account

Overall, the homepage (Figure 19) contains all information of question bank include total questions, the number of multiple-choice questions containing images, and the accessed subject. When adding a new question to the database, the lecture follows the second case of FPT university and replaces the manual keyword searching with the proposed comparison method. Each teacher prepares a document file containing the multiple-choice questions based on their assigned sub-category. The docx file will be stored in the same folder with the images belonging to the MCQs. In order to understand the scope and reduce the likelihood of producing duplicate questions, the teachers can view and search questions in the question bank since Pinecone optimises searching by using a vector database. However, the teacher can only access the subjects they are accepted by the manager. The manager sums up all the prepared materials including the images and a word file following the template illustrated in figure 20 and imports them to the website.

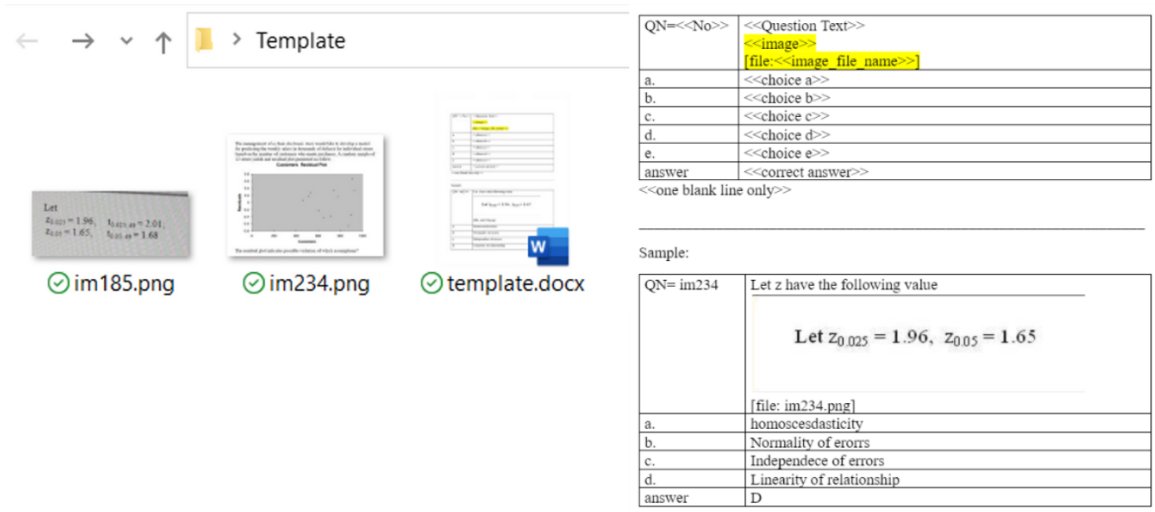

Figure 20. Folder import and template of a question in a word file (.docx)

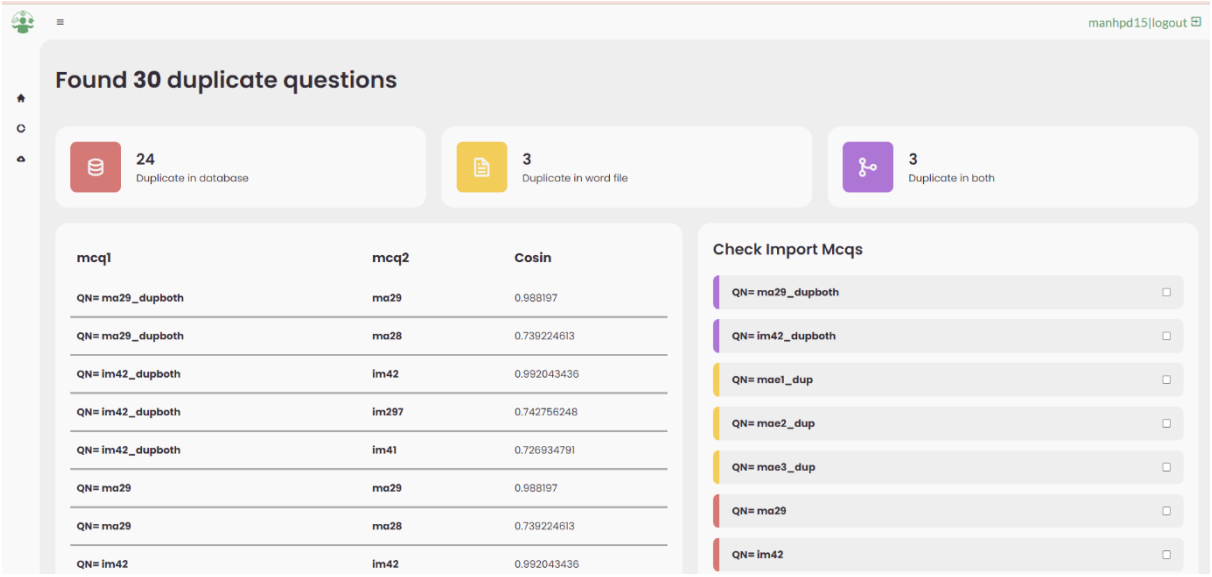

Figure 21. List of duplication question pairs

After uploading the required materials, the web application encodes the uploaded questions using OCR and SBERT. The encoded results are compared to each uploaded encoded question in the Pinecone question bank. Cosine similarity is used in the comparison phase because it better supports the Pinecone vector database. If the similarity score exceeds the threshold of 0.7, the question pairs are considered duplicates. The list of duplication pairs is listed for the manager to decide whether report back the uploaded question or add it to the question bank (Figure 21). If the imported questions are unique, the import questions are stored in a PostgreSQL database, and the question encodes are preserved in Pinecone for use in future searching and retrieval operations.

5. Conclusion and Future Work

This study focuses on enhancing the administration of the question bank at FPT University by addressing the issue of question duplication and aiming for a new approach for lectures to add a new question to the question bank. An end-to-end machine learning architecture is fine-tuned for representing questions in text or image form into a vector. The model takes advantage of OCR to extract information from images and vectorize the combination of data in images and text using SBERT. Various similarity measures were examined on the collected data set and in the end, the Spearman measure was the best with an F1 score of 0.95. The model is deployed on a web application using the Django framework applied role permissions, PostgreSQL as a database, and Pinecone vector database as a search engine. The result of a model and an application show that the proposed method can effectively support question bank management at FPT University which makes the process of adding questions more efficient by reducing the rate of duplicate multiple-choice questions.

Currently, due to the constraint on time and the limitation of human and finance resource, the constructed test bank just covers a small portion of the questions which are written only in English. Additionally, the question pairs having a similarity score between 0.7 and 0.9 cause a huge confusion for lecturers because two different questions share the same context might be still considered as duplication. In future work, we aim to tackle the problem of question duplication on an actual question bank of FPT University with different languages such as Vietnamese, Chinese, and Japanese. The presence of audio-based multiple-choice questions and the mentioned issue of similarity score are also taken into more consideration.

References

1. FPT Education. Retrieved from https://fpt.com.vn/en/business/education.
2. Purohit, V.K., Kumar, A., Jabeen, A., Srivastava, S., Goudar, R.H., Shivanagowda., Rao, S.: Design of adaptive question bank development and management system. In: 2nd IEEE International Conference on Parallel, Distributed and Grid Computing, Solan, India, pp. 256-261 (2012).
3. Mia, M.R., Latiful Hoque, A.S.M.: Question Bank Similarity Searching System (QB3S) Using NLP and Information Retrieval Technique. In: 1st International Conference on Advances in Science, Engineering and Robotics Technology (ICASERT), Dhaka, Bangladesh, pp. 1-7 (2019).
4. Wang, L., Zhang, L., Jiang, J.: Duplicate Question Detection With Deep Learning in Stack Overflow. In IEEE Access, vol. 8, pp. 25964-25975 (2020)
5. Li, Y., Yao, L., Du, N., Gao, J., Li, Q., Meng, C., Zhang, C., Fan W.: Finding Similar Medical Questions from Question Answering Websites. arXiv.1810.05983 (2018).
6. Liao, M., Shi, B., Bai, X., Wang, X., Liu W.: TextBoxes: A Fast Text Detector with a Single Deep Neural Network. arXiv:1611.06779 (2016).
7. Zhou, X., Yao, C., Wen, H., Wang, Y., Zhou, S., He, W., Liang J.: EAST: An Efficient and Accurate Scene Text Detector. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, pp. 2642-2651 (2017).
8. Baek, Y., Lee, B., Han, D., Yun, S., Lee, H.: Character Region Awareness for Text Detection. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, pp. 9357-9366 (2019).
9. Jaderberg, M., Simonyan, K., Vedaldi, A., Zisserman, A.: Reading Text in the Wild with Convolutional Neural Networks. arXiv:1412.1842 (2014).
10. Shi, B., Bai, X., Yao, C.: An End-to-End Trainable Neural Network for Image-Based Sequence Recognition and Its Application to Scene Text Recognition. arXiv:1507.05717 (2015).
11. Gokul, P.P, Akhil, B.K., Shiva, K.K.M.: Sentence similarity detection in Malayalam language using cosine similarity. In: 2017 2nd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), Bangalore, India, pp. 221-225 (2017).
12. Dhar, A., Dash, N.S., Roy, K.: Application of TF-IDF Feature for Categorizing Documents of Online Bangla Web Text Corpus. In: Bhateja, V., Coello Coello, C., Satapathy, S., Pattnaik, P. (eds) Intelligent Engineering Informatics Advances in Intelligent Systems and Computing, vol 695. Springer, Singapore (2018).
13. Devlin, J., Chang, M.W., Lee, K., Toutanova K.: BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding. arXiv:1810.04805 (2018).
14. Reimers, N., Gurevych, I.: Sentence-BERT: Sentence Embeddings Using Siamese BERT-Networks. arXiv: 1908.10084 (2019).
15. Ghojogh, B., Sikaroudi, M., Shafiei, S., Tizhoosh, H, R., Karray, F., Crowley, M.: Fisher Discriminant Triplet and Contrastive Losses for Training Siamese Networks. In: International Joint Conference on Neural Networks (IJCNN), Glasgow, UK, pp. 1-7 (2020).
16. Cer, D., Diab, M., Agirre, E., Lopez-Gazpio, I., Specia, L.: SemEval-2017 Task 1: Semantic Textual Similarity - Multilingual and Cross-Lingual Focused Evaluation. arXiv:1708.00055 (2017).
17. Thanh, T.N., Nha, N.K., Hieu, N.K., Anh, N.K., Khoat, T.Q.: Utilizing SBERT For Finding Similar Questions in Community Question Answering. In: 13th International Conference on Knowledge and Systems Engineering (KSE), Bangkok, Thailand, 2021, pp. 1-6 (2021).

18. GitHub - JaidedAI/EasyOCR: Ready-to-use OCR with 80+ supported languages and all popular writing scripts including Latin, Chinese, Arabic, Devanagari, Cyrillic and etc. [online] Available at: https://github.com/JaidedAI/EasyOCR.
19. Vincent, L., Soille, P.: Watersheds in digital spaces: an efficient algorithm based on immersion simulations. In IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 13, no. 6, pp. 583-598 (1991).
20. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, N, A., Kaiser, L., Polosukhin, I.: Attention Is All You Need. arXiv:1706.03762 (2017).
21. Zhu, Y., Kiros, R., Zemel, R., Salakhutdinov, R., Urtasun, R., Torralba, A., Fidler.: Aligning Books and Movies: Towards Story-like Visual Explanations by Watching Movies and Reading Books. In IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, pp. 19-27 (2015).
22. Zhang, T., Kishore, V., Wu, F., Weinberger, K. Q., Artzi, Y.: BERTScore: Evaluating Text Generation with BERT. arXiv:1904.09675 (2019).
23. Hadsell, R., Chopra, S., LeCun, Y.: Dimensionality Reduction by Learning an Invariant Mapping. In IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06), New York, NY, USA, pp. 1735-1742 (2006).
24. Henderson, M., Strope, B., Sung, Y., Lukacs, L., Guo, R., Kumar, S., Miklos, B., Kurzweil, R.: Efficient Natural Language Response Suggestion for Smart Reply. arXiv:1705.00652 (2017).
25. Pretrained Models — Sentence-Transformers documentation. [online] Available at: https://www.sbert.net/docs/pretrained_models.html.