İstanbul
Bilgi Üniversitesi

Bubble Sheet Scanner Using Deep Learning

by

Beyza Akgün, 121200137
İlayda Kasapçopur, 121200136

Supervised by

Özgür Özdemir

Submitted to the
Faculty of Engineering and Natural Sciences
in partial fulfillment of the requirements for the

Bachelor of Science

in the
Department of Computer Engineering

June 2025

Abstract

OMR is a technology that is being used by people all around the world for different purposes within the same old traditional systems. Traditional approaches to OMR are not compatible with today's technology. Having traditional approaches despite being a global technological system creates traditional challenges. On the other hand, Deep Learning is becoming widely popular and useful for various reasons in today's world. The versatility of Deep Learning adds superiority to its impact. However, the connection between OMR and Deep Learning remains underexplored because of the lack of public datasets and domain-based models. In this study, we proposed a novel approach that combines semantic segmentation for area detection with image captioning for answer prediction. Previous research primarily focused on either segmentation or classification, leaving a huge area to work on the integration of generative models for character-level answer prediction. The proposed pipeline consists of two main stages: answer area segmentation and character-level answer prediction. Two segmentation models, a CNN-based U-Net and a transformer-based SegFormer, were trained and evaluated to detect the answer regions on the original sheet with different setups. Segformer provided a higher performance on the results (Dice: 0.9980, IoU: 0.9959, Precision: 0.9981, Recall: 0.9978), and more accurate bounding boxes than the CNN architecture. This resulted in the selection of the SegFormer model for achieving the cropped images of the answer areas. The cropped images were then padded and fed into a custom image captioning model structured with an EfficientNet-B0 encoder and a Transformer decoder. The vocabulary was designed at the character-level to capture short-string answers. Two experimental setups were used to evaluate the impact of dataset filtering and preprocessing. Segmentation results showed that the SegFormer outperformed the CNN model in both Dice (0.9980 vs. 0.9772) and IoU (0.9959 vs. 0.9559) scores. In the captioning task, Setup 2, which excluded images with the format that linked to question kinds that could not be read, yielded higher accuracy (0.7185 vs. 0.6282) and F1 (0.4814 vs. 0.5449) scores compared to Setup 1. Nevertheless, limitations were observed in accurately predicting full-length answers, especially in handling variable-length sequences. To further evaluate the model, we aimed to integrate a vision-language model (Qwen-VL) for direct answer prediction with forwarded image and prompt. It revealed that current multimodal models are insufficient for this domain without fine-tuning, highlighting the novelty and future potential of our proposed pipeline.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

e.g.	Exempli gratia (Latin: for example)
Et al.	Et alii (Latin: and others)

1 Introduction

Optical Mark Recognition (OMR) [3] is a widely used technology that automates the reading of marked data from documents. Manual processes are being replaced by OMR because of their time-consuming and error-prone features. This technology is commonly used in surveys, examinations, healthcare, and marketing. As technology has evolved, OMR has also been affected by transitioning from basic hardware-based solutions to more flexible and cost-effective software implementations.

In the past, OMR systems used specialized hardware, such as scanners and optical readers [3]. These old systems were limited in flexibility and often required predefined templates. For larger implementations, it also required a lot of time to process. After, the advancements in image processing and computer vision, it creates a space for OMR solutions to become more adaptable and accessible. However, there are still some challenges, such as scanner quality, light distortion, and skew errors.

In this project, we proposed a new approach to OMR: a bubble sheet scanner using deep learning. Using the power of deep learning techniques, including answer area segmentation using convolutional neural network (CNN) and transformer, answer prediction using image captioning, and evaluation using vision-language model.

We compared the results of different architectures and experimental setups for segmentation to find the better choice for our project. Unlike traditional approaches, we used image captioning directly to predict the answers by analyzing the given cropped answer areas. For evaluation, we used a large vision-language model with full-page answer sheets with textual prompts to create a comparison with our predicted results.

Our contributions include a hybrid deep-learning pipeline offering a novel approach for flexible, template-free OMR systems. Our aim is to create a unique perspective to the traditional methods.

2 Related Works

The field of OMR has witnessed significant advancements, transitioning from old hardware-based systems to more flexible software and deep learning approaches. This section explores related works that shows the evolution of OMR, emphasizing machine learning methods and the impact of datasets on system performance. Furthermore, it builds on the techniques that we adopted through our project, providing insight information for the decisions we made through.

A total review of 35 OMR studies by de Elias et al. [3] identified several critical challenges, including sensitivity to noise, alignment issues, and the lack of standardized datasets. Although the study reported an average accuracy of 98.85%, these metrics may not fully represent the effectiveness of OMR systems with traditional approaches. A significant limitation is based on the datasets used. Some studies relied on datasets that were either too small or lacked diversity, a dataset including five answer sheets used in the study by Talib et al.[4]. This restricted scope of datasets reduces error variability during evaluation, creating an unrealistic accuracy of system performance and failing to account for real-world complexities.

The review also points out the reliance on traditional OMR processing and marker techniques, such as template matching, pattern recognition, pixel counting, and template comparisons. Moreover, the majority of datasets remain non-public. Key constraints in existing systems include the explicit use of fiducial markers, reliance on geometric delimiters like triangles and bounding boxes, and dependency on specific templates or predefined options. These limitations create core challenges, such as sensitivity to noise and skew, and the lack of standardized datasets, which continue to restrict the adaptability and scalability of OMR technology.

In another study, Loke et al. [5] introduced a software-based Optical Mark Recognition (OMR) system that addresses critical challenges such as noise, and diverse marking styles, which often challenge the performance of traditional OMR methods. Their approach supports advanced mark detection techniques, including Gaussian adaptive thresholding and RANSAC-based [6] statistical methods, to improve the accuracy of mark recognition. By effectively handling a variety of marking styles (e.g., ticks, crosses, partially filled bubbles) and reducing noise and artifacts, the system shows its adaptability to real-world conditions.

This work highlights the potential of software-based OMR systems to provide high accuracy and flexibility without the reliance on expensive hardware. While the study highlights the transformative capabilities of advanced mark detection methods, it also points out limitations, such as the need for

optimization for specific bubble sizes and reduced effectiveness for handwritten fields. Regardless, this research serves as a foundation for advancing software-based OMR, aligning closely with our focus on using deep learning techniques to overcome the remaining challenges in flexibility, scalability, and dataset diversity.

More importantly, Afifi et al. [7] proposed a semi-automated OMR system that addresses the limitations of traditional multiple-choice question (MCQ) grading systems by redefining the task as a machine learning-based classification problem. Unlike classic methods that rely on simple image processing, their approach determines between three classes of answer boxes (confirmed, crossed-out, and empty) through a trained machine learning model. This redefinition helps the system handle diverse marking styles and patterns.

The authors created a dataset having 735 answer sheets from 6 MCQ examinations. This dataset features various templates and diverse marking styles, making it powerful compared to other datasets available in the field. Three classification methods were evaluated: Naive Bayes Classifier (NBC)[8], Bag of Visual Words (BoVW)[9], and Convolutional Neural Networks (CNN) architecture proposed by Krizhevsky et al.[10]. The study also introduced a two-stage classification strategy to reduce errors, by achieving improved performance. The experimental results demonstrated that CNN achieved the highest accuracy in end-to-end classification. However, the two-stage strategy, combining BoVW and CNN, showed better generalization.

While the study achieves remarkable accuracy and flexibility, it acknowledges certain limitations, including the need for improved error correction mechanisms and further generalization to broader datasets. This research demonstrates the potential of implementing machine learning techniques in OMR to overcome real-world challenges, placing closely with our project's aim of using deep learning to improve bubble sheet scanning.

Building on these advancements, our project introduces a deep learning-based OMR system that addresses traditional limitations. By using state-of-the-art models, image-based classification techniques, and multimodal learning; we aim to further enhance adaptability and diversity in bubble sheet scanning.

Although CNN-based segmentation methods, particularly U-Net [11] for our case, are not commonly applied in the domain of OMR, they have demonstrated strong performance in similar pixel-level prediction tasks such as medical imaging [12]. Inspired by this success, we adapted a U-Net-based approach to detect the answer areas in our bubble sheets dataset.

In addition to CNN-based segmentation, we also experimented with transformer-based models to achieve the results of two competitive architectural methods. For this purpose, SegFormer [2], a lightweight yet powerful semantic segmen-

tation framework was selected due to its superior performance and efficiency. Unlike traditional CNNs, SegFormer combines hierarchical attention with multi-scale features, making it practical for tasks like answer area detection.

During the development of our project, we stated image captioning [13] as a new and promising approach to predict the answers from answer areas, since we realized that this method has not been applied in the field of OMR. We observed that combining computer vision with language modeling techniques, specifically using an encoder–decoder architecture, could enable the system to evaluate visual answer patterns as textual outputs. In our implementation, we utilized EfficientNet-B0 as the visual encoder due to its strong performance-to-efficiency ratio [14], followed by a transformer-based decoder to generate answer predictions. This integration of visual and textual processing represents a new perspective in the domain of automated answer sheet analysis.

We also incorporated with Qwen [15], a vision-language model for evaluation of our result that we obtained from the image captioning process. This model specializes in analyzing the image with given textual prompts. It is a newer approach by Alibaba Group, offers a strong benchmark for further researches, and validates the effectiveness of multimodal evaluation in OMR tasks.

3 Design

Our design for this project is a structured pipeline that integrates data collection and preprocessing, segmentation experiments and model selection, collecting outputs as cropped images and padding, applying our image captioning model and experimenting with different setups to improve the prediction, evaluating with multimodal modeling and concluding the results. Our workflow aimed to analyze deep learning techniques with improved setups after each evaluation and achieve promising results with provision of new research aspects to the OMR area. Below, we showed the design's core components and logical workflow.

Figure 1: Design of Our Project

First, we prepare the dataset which includes scanned answer sheet images for our use. Then we apply pre-processing methods, and generated masks.

After deciding each model's definition, we trained the models for segmentation task. After selecting the best performed model, we saved the outputs of bounding boxes and added padding to make the dataset consistent for image captioning stage. After experimenting with image captioning, we evaluate the results accordingly. To have another approach on all the systems that we had, we wanted to add visison-language model for further comparison. After conducted all of the results, we came up with the conclusion and highlighted the challenges and future works for optimization of the project.

4 Methodology

4.1 Preliminary Works

As we said earlier there wasn't a lot of public dataset options for us to use. For this work, we used the dataset from the study by Afifi et al. [7] that we mentioned earlier in the related works section. We gathered 171 scanned exam images from various exam styles using this dataset.

However, these images were unlabeled, so we used VGG Image Anotator (VIA)[16] to name them. The bubble areas have been defined as bounding boxes, and we assign a label of 1. Additionally, we define the remaining portion of the image as the background and labeled 0. Because of the non-rectangular shape of the bubble area sections, this labeling method introduced slight differences even though it offered a good beginning point. Later on, this dataset was split into an 70% training set,15% validation set and a 15% testing set. Then we did the mask generation and saved the binary masks in a related directory. To manage data loading,and batching during model training, ImageMaskGenerator, a custom Keras data generator, was implemented. Both images and masks were resized to 256x256 pixels by the generator, which also normalized pixel values to fall between [0, 1].

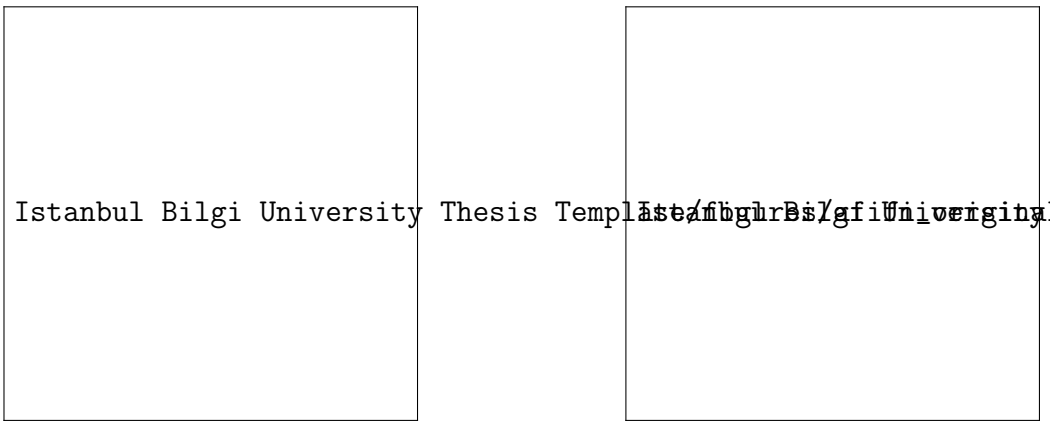


Figure 2: Original image and labeling process.

For the segmentation task, we evaluated both a CNN-based U-Net and a transformer-based SegFormer, each trained for 20 epochs with batch size of 4.The SegFormer model was developed using the "nvidia/segformer-b0" backbone (about 3M parameters), fine-tuned with cross-entropy loss using AdamW (lr=5×10), and a StepLR scheduler (=0.5 every 5 epochs). Our U-Net had two down-sampling conv blocks (32-64 filters), a 128-filter bottleneck, and mirrored up-sampling with skip-connections. It was trained with dice loss using Adam (initial lr=$1\times10^{\mathbf{3}}$) and an exponential decay schedule. The final results were promising:

Metric	**U-Net (CNN)**	**Segformer**
Loss	0.4434	0.0582
Accuracy	0.8155	0.9880
Precision	0.7517	0.9750
Recall	0.4419	0.9793
F1-score	0.5566	0.9771
Dice Coefficient	0.5566	0.9771
IoU	0.3856	0.9553

Table 1: Test performance comparison between U-Net (CNN) and Segformer models

Figure 3: Original Image - Predicted Mask - Ground Truth Mask (U-Net)

Although both models typically generated masks that visually matched the real bubble areas in the images, a significant problem emerged during the labeling process: the ground-truth masks were defined as perfect rectangles, despite the fact that the real bubble areas had tiny cutouts (missing bottom-left section).

The fact that the U-Net model was able to find this geometric change is surprising. Although the global performance measure was worse, the resulting masks were more correct with respect to the actual local bubble size. This is probably because of the convolutional nature of U-Net. Through the use of skip links and hierarchical feature extraction, this architecture emphasizes the local features of the image. Compared to Transformer models,

Figure 4: Original Image - Predicted Mask - Ground Truth Mask (SegFormer)

U-Net is more sensitive to modest, localized changes in the input image, and thus better captures such local changes in the output.

Even though it performed better statistically (e.g., IoU of 0.95 and Dice of 0.97)SegFormer neglected this deviation of structure to generate results exactly close to the shapes of the ground-truth masks. As a Transformer-based architecture, SegFormer fundamentally relies on global context and spatial attention, which might induce global shape consistency tracing over small-scale abnormal regions—particularly when such regions are not uniformly annotated during training or underrepresented. This is not due to a model error, but rather the architecture of the model.

These results obtained from the relatively non-optimal dataset, proved the future of our approach and encouraged us to continue our work. Data quality would be essential for the performance of the models, motivating us to find a more suitable dataset that aligns with our project's requirements.

4.2 Dataset and Preprocessing

Our academic advisor, Özgür Özdemir, provided us with the dataset which includes the sacnned exam papers of his courses. The images are in high-resolution PNG format. The dataset has 47 questions, and 5042 exam papers in total. The layout of the answer regions are different for each question, but it remains consistent inside per question. He also provided us with the json file that contains all the necessary information of each paper, including the path of the paper ("path"), utilizable values ("values"), alignment of the answers whether it is horizontal or vertical ("alignment"), number of answer areas ("nforms"), and answers ("answer").

To standardize the papers and reduce misalignment due to scanning inconsistencies, we utilized the OMR Checker Tool [17] to align all papers by their question types. This alignment ensured that all answer areas in each question were positioned relative to a common frame of reference. In the following, we get the coordinates of the answer areas and saved them in the json file as the "answer_area" variable.

With all the information, we created the ground-truth mask for our segmentation models. For creating our masks, we took our answer-area coordinates from the json file, and then we converted to pixel-level binary masks accordingly. For mask creation, our code red the bounding box coordinates in the format of $[x_1, y_1, x_2, y_2]$ for single or multiple boxes in answer sheets. This conversion was saved by labeling 1 for the bounding boxes and 0 for the background, and saved as .npy files in our directory. All data preparation and model training were performed in Google Colab, with the dataset stored and accessed via Google Drive integration.

Figure 5: Original image, Segmentation mask, Overlay mask

Figure 6: Original image, Segmentation mask, Overlay mask

4.3 Segmentation Approach

To identify the regions of answer bubbles on each sheet, we implemented semantic segmentation. We aimed to achieve it by pixel-level classification, rather than just using bounding box detection. Two different approaches were used through this study: CNN approach (U-Net architecture), Transformer approach (SegFormer). As the input, original image was used. During every training setup, we used the 80the 20using Adam optimizer with

an initial learning rate of 1e-4, and early stop- ping was applied based on validation IoU. After training, the best model was saved under a specified name to be stored in our Google Drive. During the evaluation, the following metrics were used: accuracy, loss, dice, IoU, precision, and recall

4.3.1 CNN Approach

Initially, we trained a CNN-based segmentation model using the U-Net architecture. The model was trained with two different experimental setups for evaluation.

Figure 7: U-Net Architecture [1]

For the first experimental setup, the model was trained with a batch size

of 2 and resized inputs of 416×576 pixels. Training lasted for 50 epochs, with validation occurring after each epoch. And for the second experimental setup, we utilized resized inputs of 832×1176 pixels, since this change affect the training periods, we reduced the epochs to 20 in order to keep the operation non-time consuming. The performance comparison between the two experimental setups is summarized in Table 2.

Şunu dedin:

Metric	**Setup 1**		**Setup 2**	
	Training Score	Validation Score	Training Score	Validation Score
Dice	0.9772	0.9741	0.9767	0.9695
IoU	0.9559	0.9500	0.9550	0.9418
Precision	0.9747	0.9610	0.9811	0.9672
Recall	0.9801	0.9881	0.9807	0.9856
Accuracy	0.9941	0.9932	0.9933	0.9924

Table 2: Performance comparison of U-Net CNN model under different setups

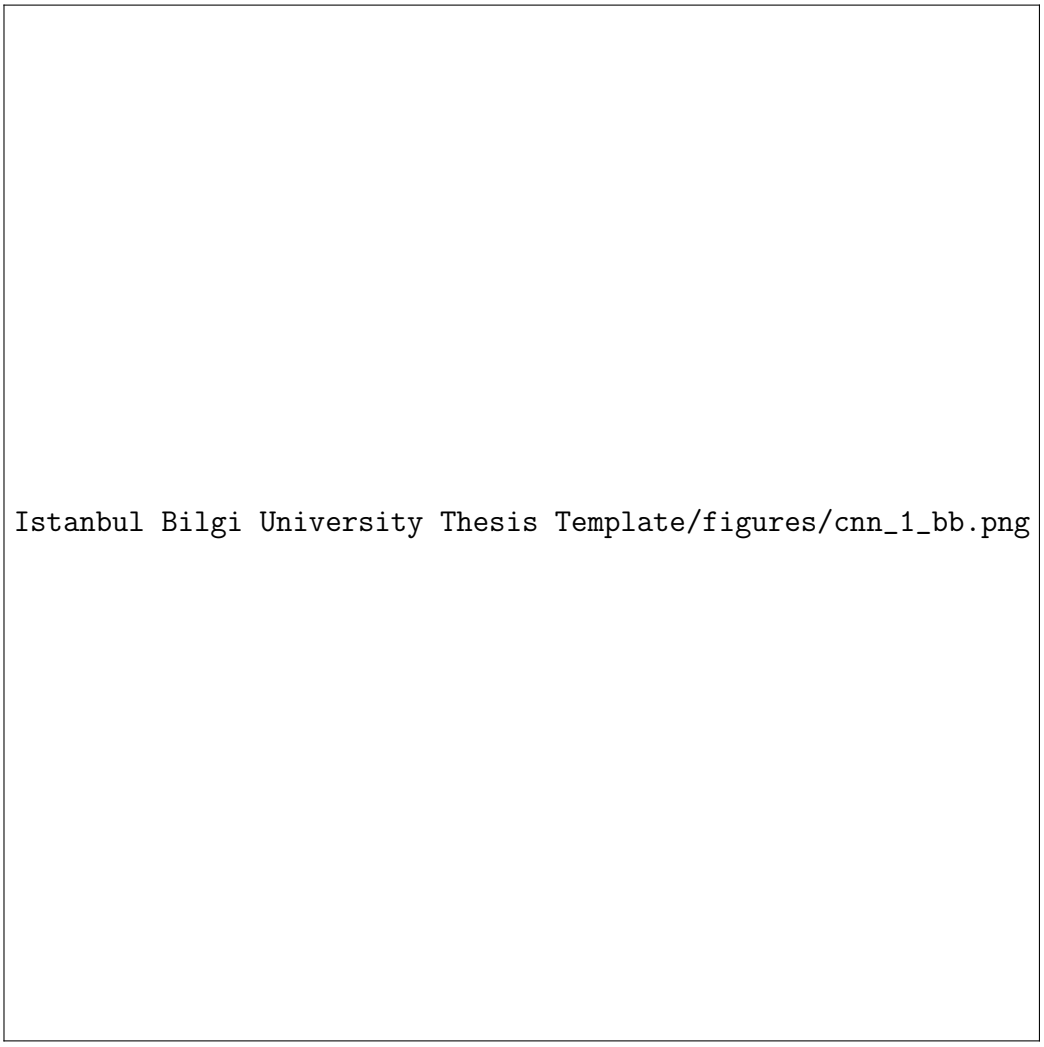


Figure 8: Ground Truth Mask, Predicted Mask, Bounding Box Prediction (Setup 1)

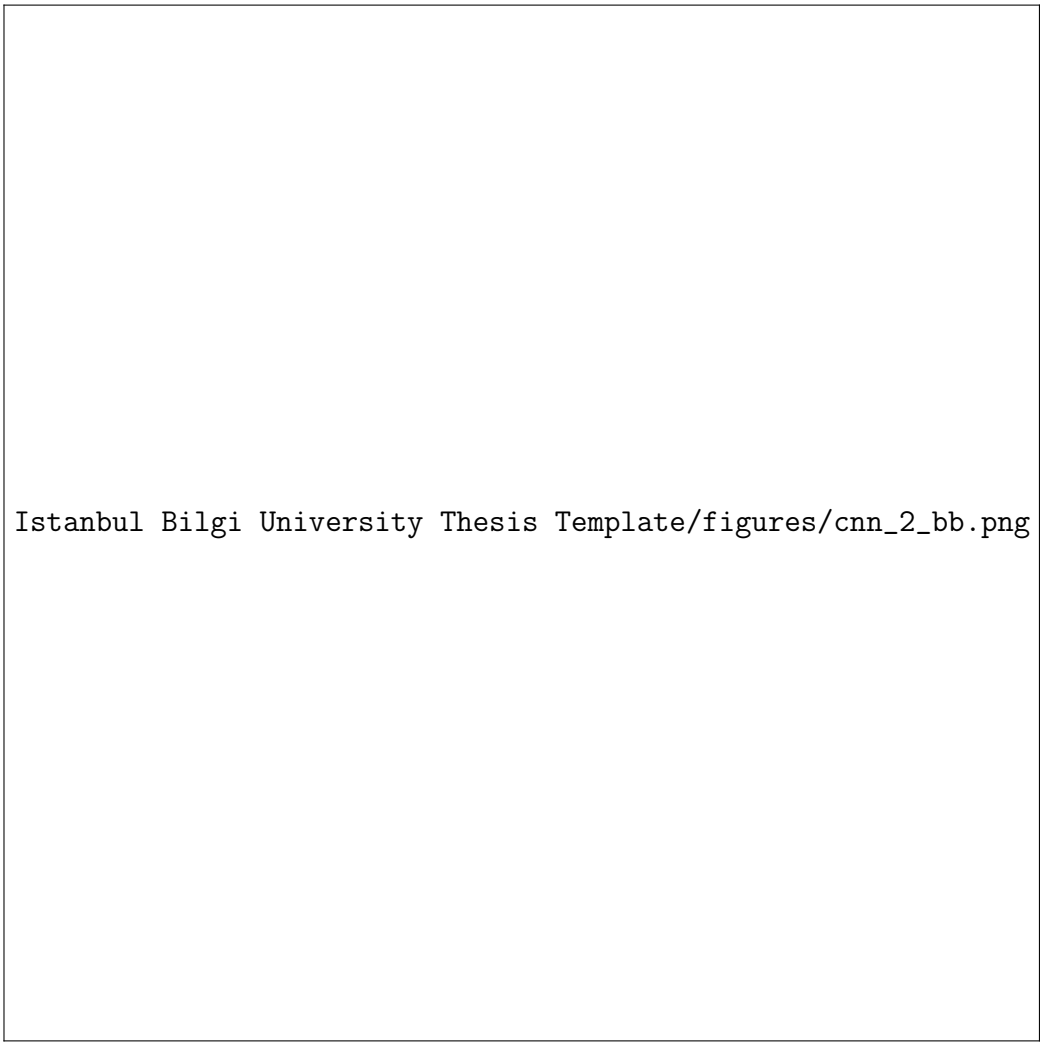
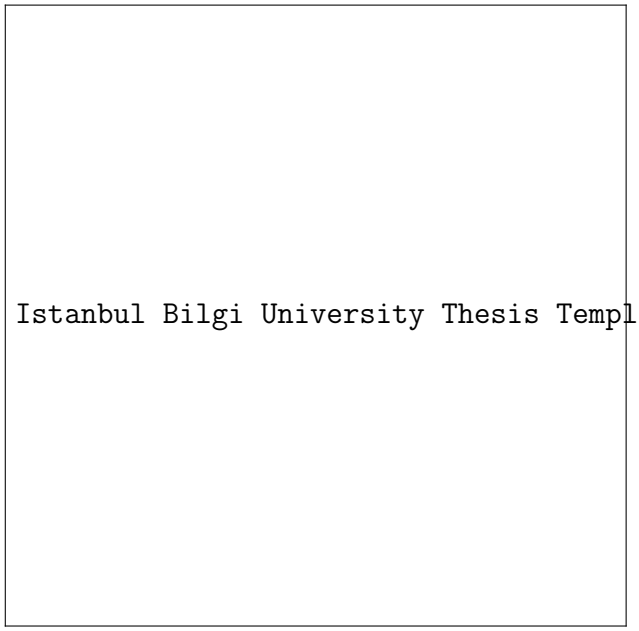
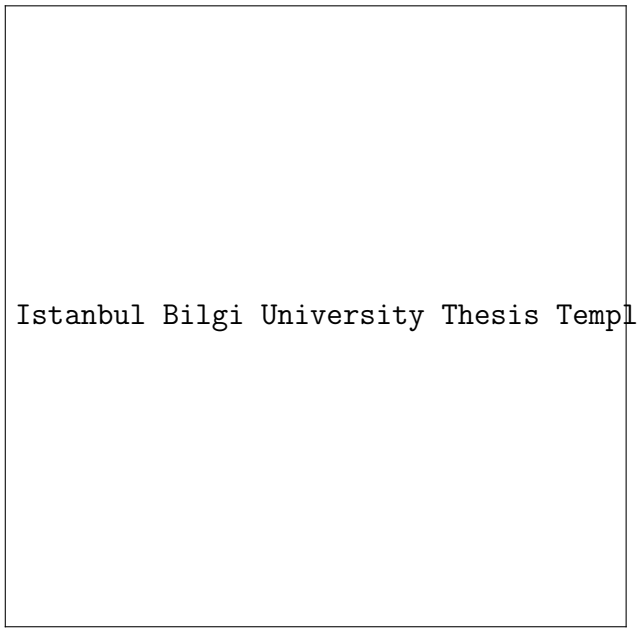


Figure 9: Ground Truth Mask, Predicted Mask, Bounding Box Prediction (Setup 2)

These scores showed that our U-Net architecture gives high performance for both setups. But we realized that even though there were higher training scores for Setup 2, it had lower validation accuracy suggesting overfitting. However, when we visualized the output images with overlay masks (Figure3), we realized that the predicted bounding boxes were slightly broader than the true answer-areas. This issue was consistent across both setups and showed the weaknesses of the model's ability to pinpoint small regions precisely. Because of that, we decided to test a transformer-based segmentation model, specifically SegFormer due to its ability to capture local details and global context more accurately.



(a) Setup 1



(b) Setup 2

Figure 10: Visualization of loss and IoU values throughout the processes for both training and validation.

4.3.2 Transformer Approach

Subsequently, we experimented with a transformer-based model, SegFormer, because of its ability to capture the context through self-attention mechanisms. The model was trained with two different experimental setups for evaluation.

Figure 11: SegFormer Architecture [2]

For the first experimental setup, the model was trained with a batch size of 8 and resized inputs of 224×224 pixels. Training lasted for 30 epochs, with validation occurring after each epoch. And for the second experimental setup, we utilized resized inputs of 512×512 pixels, since this change affect the training periods, we reduced the epochs to 10 this time. Transformers are

relatively larger models than CNNs, causing us to reduce the epochs value more to keep the training stable. The performance comparison between the two experimental setups is summarized in Table 3.

Metric	**Setup 1**		**Setup 2**	
	Training Score	Validation Score	Training Score	Validation Score
Dice	0.9997	0.9996	0.9972	0.9982
IoU	0.9994	0.9991	0.9944	0.9965
Precision	0.9997	0.9994	0.9973	0.9985
Recall	0.9997	0.9998	0.9971	0.9980
Accuracy	0.9999	0.9999	0.9993	0.9996

Table 3: Performance comparison of transformer based SegFormer model under different setups

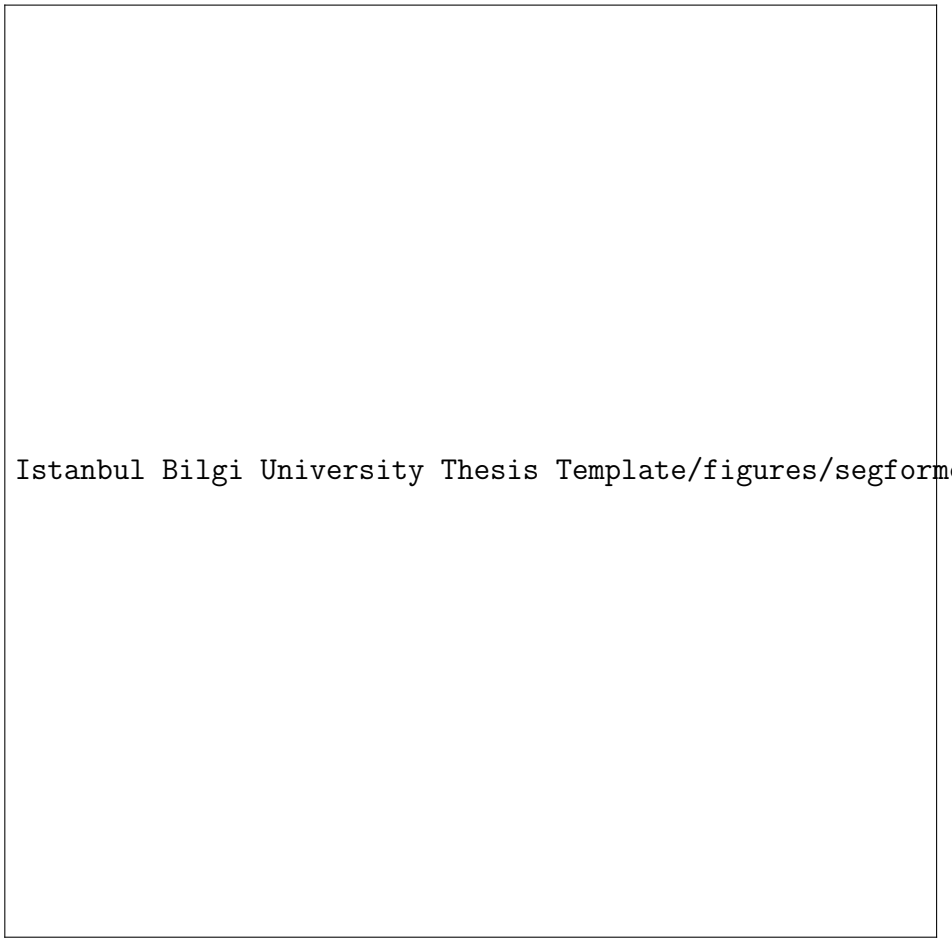


Figure 12: Ground Truth Mask, Predicted Mask, Bounding Box Prediction (Setup 1)

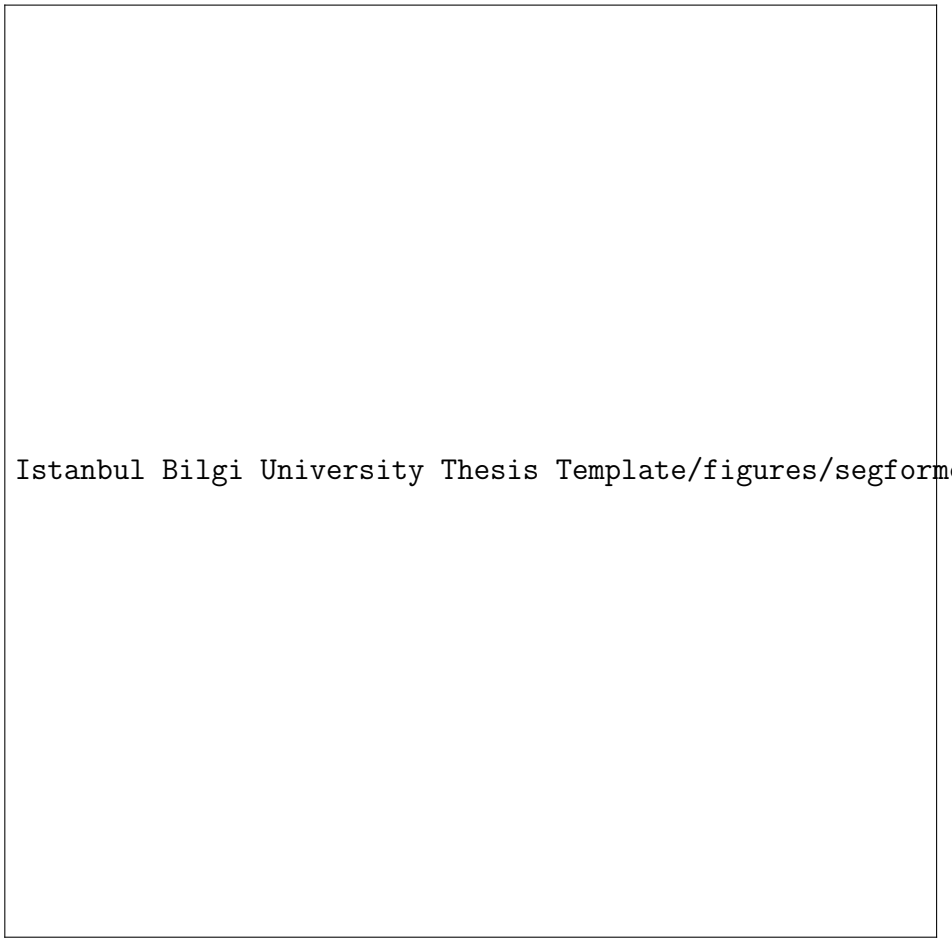
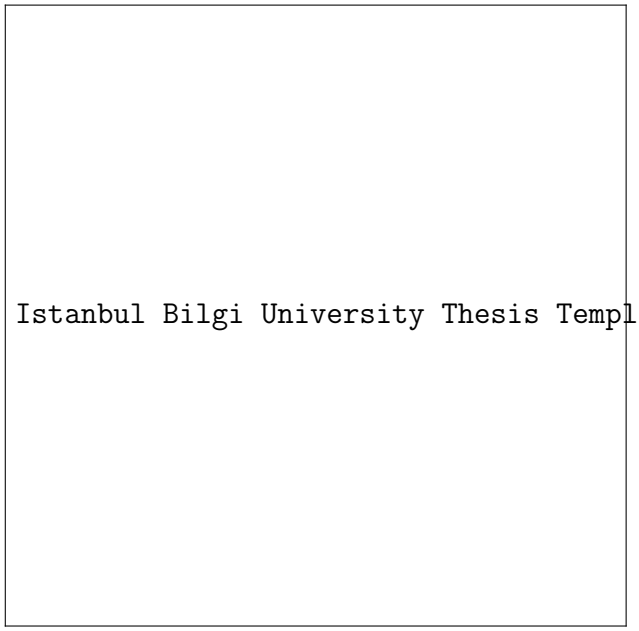
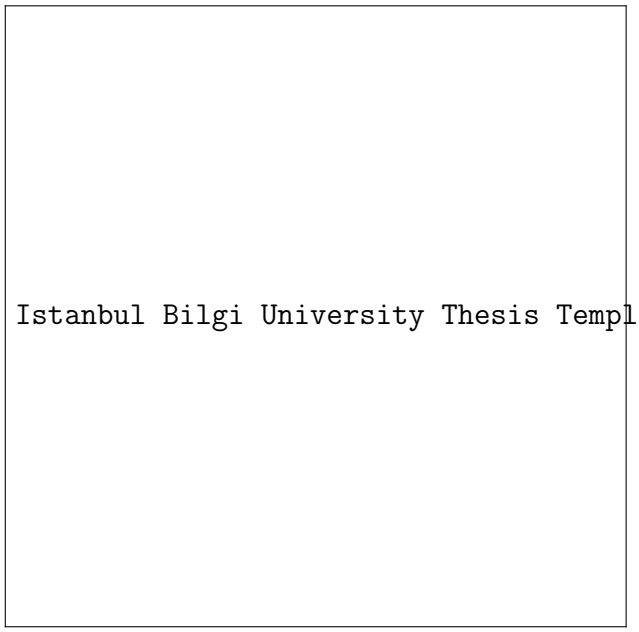


Figure 13: Ground Truth Mask, Predicted Mask, Bounding Box Prediction (Setup 2)



(a) Setup 1



(b) Setup 2

Figure 14: Visualization of loss and IoU values throughout the processes for both training and validation.

Table 3 showed that SegFormer experiments gave better results than CNN

scores. In addition, it captured bounding box areas more accurately, which confirmed our reasoning for using the transformer-based approach. Although Setup 1 gave better results than Setup 2, we chose to proceed with Setup 2 outputs because of the better pixel quality, since it is critical requirement for our next experimental step, image captioning.

4.4 Cropping and Image Normalization

After training and comparing the models, the best-performing version (SegFormer 512×512) was saved and used to generate bounding boxes around the detected answer regions. These bounding boxes were used to extract cropped images that contain only the answer areas.

In order to standardize the input for the image captioning stage, all cropped images were padded to a fixed size. This step ensured compatibility with the input requirements of the image captioning model. The cropped-padded images were then used as input for the answer prediction model based on image captioning.

Figure 15: Cropped image example

4.5 Answer Prediction via Image Captioning

At the last step of our pipeline, we analyzed image captioning models to achieve answer prediction based on the cropped-padded images constructed from the SegFormer stage. The purpose of the model was to predict the answers (e.g., "ABDCB", "011101", "bbbaaa", " ") from the image in textual format by learning the visual patterns of the marked and unmarked bubbles.

Automating bubble-sheet scoring needs exact sequence prediction from visual input, which is difficult to do because answers vary in length and may contain letters, numbers, or symbols.

4.5.1 Dataset and Vocabulary

Initially, we filtered out vertically formatted scans due to their substantial differences in aspect ratio and bubble layout, as well as their limited number. Including both vertical and horizontal formats would have introduced high visual variability without sufficient examples from either type, likely making training unstable. After this initial filtering, we retained 3,635 horizontally aligned bubble-scan entries. From these, we further excluded samples linked to question types that could not be read, ultimately yielding 2,814 usable entries. There were one to four cropped bubble sections in each of these entries, yielding 4,324 training samples, 536 validation samples, and 594 test samples with a distribution of 80% / 10% / 10%. We did not use any data augmentation techniques at this point, such as flipping, rotation, or artificial noise, saving these for later research to enhance generalization even more.

For output creation, we developed the vocabulary by working at the character level. All unique characters—letters, numbers, and symbols—extracted from the answer labels in the training data are included in the tokens. Furthermore, we incorporate five unique tokens: to align sequences of equal length, to indicate the beginning of a sequence, to indicate its conclusion, for any characters that are not in the vocabulary (although all observed characters are covered in practice), and to indicate a completely unmarked or blank response. A final vocabulary size of 36 is reached by giving a unique index to each unique character and special token.

4.5.2 Model Architecture and Training

Our captioning approach achieves a balance between expressive strength and efficiency by combining a lightweight Transformer decoder with a pre-trained EfficientNet-B0 encoder. A fully connected layer that transforms each image into a 512-dimensional embedding takes the place of the encoder's classification head after it has been set with ImageNet weights. A decoder consisting of three stacked TransformerDecoder layers, each with eight self-attention heads over 512-dimensional token embeddings, uses this global image representation as its only "memory" input. We use greedy decoding with a teacher-forcing ratio of 0.5 during training, and we update parameters using the AdamW optimizer (learning rate = 1e-4, weight decay = 1e-4) and a

Cosine Annealing scheduler through 20 epochs using cross-entropy loss (ignoring padding tokens). . The model is trained on a Tesla T4 GPU using the AdamW optimizer and a Cosine Annealing learning rate scheduler for 20 epochs. A batch size of 8 and a fixed random seed (42) are used for reproducibility.

4.5.3 Evaulation Metrices

Metric	**Score**
Token-level accuracy	0.7185
Token-level F1	0.4814
Exact-match sequence accuracy	0.5240
BLEU-1	0.5031
BLEU-2	0.0007
BLEU-3:	0.0001
Average Character Error Rate (CER)	0.2113

Table 4: Validation performance over 20 epochs

Figure 16: Training vs. Validation Loss and Key Metrics over Epochs

Figure 17: Token Accuracy and F1 over Epochs

Figure 18: Sequence Exact Match and Avarage CER over Epoch

The table and graphs above show the model's performance after 20 training epochs. The training loss stabilized at 0.5978, while the validation loss hit 0.7074, suggesting that, while both have consistently decreased there is still a notable gap, indicating some generalization difficulties.With a token-level accuracy of 71.85%, the model is able to identify most individual characters in the bubble crops. Nevertheless, the token-level macro-F1 score of 0.4814 suggests that issues of class imbalance and confusion remain, especially regarding visually similar characters.With a sequence exact-match accuracy of 52.40%, over half of the predicted sequences are entirely accurate.Based on

the average Character Error Rate (CER) of 0.2113, approximately 21% of the characters in the predictions are either incorrect or misordered.

Although there are some noticeable fluctuations, particularly at epochs 3, 5, and 12, which may indicate temporary overfitting or noise at the batch level, the training and validation loss plots show a consistent downward trajectory, with the training loss declining steadily and the validation loss generally mirroring this trend. The token accuracy and F1 score plots show clear progress: token accuracy slowly increases and peaks at roughly 72%, while F1 rises gradually but remains lower throughout, displaying ongoing class-level confusion. The exact-match graph shows how sequence-level precision improves with each epoch, eventually reaching slightly more than 50%. The CER graph shows a steady decline, reaching a low of 0.21 by epoch 20, indicating reduced character-level errors even when full sequence correctness is not reached.

Overall, the metrics indicate that the model is capable of accurately predicting most individual characters and is progressively better at understanding complete sequences. Nevertheless, its higher-order precision is still constrained, as evidenced by BLEU-2 to BLEU-4 scores that are close to zero.This highlights the difficulty of putting characters in the right order, especially in longer or more complex sequences. Furthermore, the macro-F1 and CER findings indicate that the model continues to struggle with some character classes, particularly in instances involving occlusion or visual ambiguity.

4.5.4 Results, Analysis, Limitations Discussion

Figure 19: Example Predictions

These examples demonstrate both the model's strengths and limitations. In simpler patterns like "000001" or alternating numeric sequences like "010110," the model performs well. The model appears to handle blank responses effectively, as seen by examples when no marked answers were accurately understood as empty predictions. This shows a high sensitivity to the absence of visual input. In more complicated examples, such as "57842041" vs. "7544201," the model misses characters and scrambles the sequence,

most likely due to visual occlusion caused by full bubbles and confusion between tightly packed symbols. The "ABABDA" to "ABABAD" example demonstrates the model's inclination to swap characters in repeating patterns, revealing flaws in fine-grained sequence alignment.

The measures demonstrate consistent increases in token-level accuracy and lower character mistake rates, implying better character-level predictions. However, sequence exact-match accuracy plateaus around 50%, and validation loss remains unstable, indicating difficulties in reconstructing whole sequences, particularly in ambiguous or visually full examples. Misordered or missing letters are common in lengthier inputs or among visually identical symbols, indicating weaknesses in spatial discrimination and contextual awareness. Higher-resolution features, localized attention, and explicit modeling of positional dependencies might assist solve these challenges. The low macro-F1 indicates a class imbalance, which might be addressed by weighted losses or focused sampling. While no augmentation was used, future research might benefit from synthetic variants to improve robustness.

To summarize, the model accurately catches the majority of individual character signals and correctly handles blank or basic response patterns. However, precise full-sequence reconstruction remains difficult, especially under occlusion and when characters are visually identical.

5 Discussion

In our segmentation experiments, both CNN-based (U-Net) and transformer-based (SegFormer) models were evaluated. While the CNN setups achieved good metric scores, they frequently lacked producing bounding boxes precisely and captured more background than necessary. On the other hand, SegFormer provided higher metric scores and more precise boundary predictions, aligning closely with the actual answer areas. This justified our decision to continue the image captioning phase with SegFormer outputs.

To maintain consistent input size for the image captioning model, all cropped answer areas were padded to fixed dimensions. This standardization was crucial for enabling batch processing. After experimenting with both black and white paddings, we decided to work with the white one for consistency in the background coloring of both answer areas and empty areas. Although padding created some empty space around the bubbles, it ensured alignment consistency across dataset, which was the main point of this process.

The wide variety of answer formats and the removal of vertically formatted questions reduced dataset size and diversity, limiting the model's generalization—especially in sequence ordering. While token-level accuracy was decent, exact matches were mainly for shorter, clearer patterns. Visual issues like faint or obscured letters caused character errors. Although some repetition problems were fixed, future improvements depend on expanding the dataset with more balanced and complex samples to enhance robustness and sequence accuracy.

To further evaluate our approach of answer predictions with image captioning, we experimented with using a vision-language model, Qwen-VL for analyzing multimodal modeling capabilities. This model generates answers from the provided original answer sheet images and prompts. We aimed to test whether using such advanced models could outperform our pipeline. However, the results were far from the expectations. The model struggled to examine the images provided and responded with non-capabilities of the model. This outcome underlines the lack of domain-specific tuning and data availability for vision-language models in the OMR field. It also highlights the fact that this area remains largely unexplored and requires further research.

Despite the challenges, this study applies the foundation for a novel direction in OMR systems. The proposed pipeline demonstrates the potential for reading stylized marked answers without relying on traditional rule-based methods. With improved datasets and training, this method could handle various sheet layouts, orientations, and answer formats, making it applicable

to real-world exam processing systems.

6 Conclusion

This study demonstrated a novel OMR pipeline that combines segmentation with image captioning based answer predicition. The segmentation experiments handled with both CNN (U-Net) and transformer-based (SegFormer) models. Cropped answer areas were padded to a fixed size before being fed to the captioning model. CNN models did broader detection for bounding boxes, whereas SegFromer models did more accurate predictions, which resulted being adopted in the final pipeline.

For captioning, two experimental setups were tested. After the filteration of the visually empty bubbles in the dataset, Setup 2 performed higher accuracy. However, the overall difficulties of the dataset and empty answers resulted with prevention of perfectly consistent predictions. Some limitations like only using the horizontally aligned answer sheets happened because the available dataset was already challenging for the models and therefore none of the models were practical for OMR datasets in the first place. Consequently, the model repeated the first character in predictions or produced patterned outputs.

In conclusion, this study offers a new pipeline that has never been applied before. It also provides a perspective of marked area detection and character-level identification. Since this is just a starting point for a new research area, we would like to say that there are some important points that will carry this project further. Future works should focus on:

- Extending the captioning model to vertically aligned answer sheets and mixed layouts.
- Improving captioning scores through beam search, error-tolerant decoding, or additional fine-tuning.
- Collecting and annotating larger and more balanced datasets to enhance generalization.

With these improvements, the proposed pipeline has the potential to become a practical solution for real-world OMR applications while using the latest innovations.

References

[1] A. Abdollahi, B. Pradhan, and A. M. Alamri, "An ensemble architecture of deep convolutional segnet and unet networks for building semantic segmentation from high-resolution aerial images," *Geocarto International*, vol. 37, no. 12, pp. 3355–3370, 2022.

[2] E. Xie, W. Wang, Z. Yu, A. Anandkumar, J. M. Alvarez, and P. Luo, "Segformer: Simple and efficient design for semantic segmentation with transformers," in *Advances in Neural Information Processing Systems*, vol. 34, pp. 12077–12090, 2021.

[3] E. M. de Elias, P. M. Tasinaffo, and R. Hirata, "Optical mark recognition: Advances, difficulties, and limitations," *SN Computer Science*, vol. 2, no. 367, 2021.

[4] A. Talib, N. Ahmad, and W. Tahar, "Omr form inspection by web camera using shape-based matching approach," *International Journal of Engineering Research*, vol. 3, no. 4, pp. 29–35, 2015.

[5] S. C. Loke, K. A. Kasmiran, and S. A. Haron, "A new method of mark detection for software-based optical mark recognition," *PLoS ONE*, vol. 13, no. 11, p. e0206420, 2018.

[6] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," in *Readings in Computer Vision*, pp. 726–740, Elsevier, 1987.

[7] M. Afifi and K. F. Hussain, "The achievement of higher flexibility in multiple-choice-based tests using image classification techniques," *IJDAR*, vol. 22, pp. 127–142, 2019.

[8] D. D. Lewis, "Naive (bayes) at forty: the independence assumption in information retrieval," in *European Conference on Machine Learning*, pp. 4–15, 1998.

[9] G. Csurka, C. Dance, L. Fan, J. Willamowski, and C. Bray, "Visual categorization with bags of keypoints," in *Workshop on Statistical Learning in Computer Vision, ECCV*, pp. 1–2, 2004.

[10] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, pp. 1097–1105, 2012.

[11] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*, pp. 234–241, Springer, 2015.

[12] B. Kayalibay, G. Jensen, and P. van der Smagt, "Cnn-based segmentation of medical imaging data," *arXiv preprint arXiv:1701.03056*, 2017.

[13] M. Z. Hossain, F. Sohel, M. F. Shiratuddin, and H. Laga, "A comprehensive survey of deep learning for image captioning," *ACM Computing Surveys (CsUR)*, vol. 51, no. 6, pp. 1–36, 2019.

[14] D. H. Fudholi, Y. Windiatmoko, N. Afrianto, P. E. Susanto, M. Suyuti, A. F. Hidayatullah, and R. Rahmadi, "Image captioning with attention for smart local tourism using efficientnet," in *IOP Conference Series: Materials Science and Engineering*, vol. 1077, p. 012038, IOP Publishing, 2021.

[15] J. Bai, S. Bai, Y. Chu, Z. Cui, K. Dang, X. Deng, Y. Fan, W. Ge, Y. Han, F. Huang, *et al.*, "Qwen technical report," *arXiv preprint arXiv:2309.16609*, 2023.

[16] A. Dutta, A. Gupta, and A. Zissermann, "Vgg image annotator (via)." `http://www.robots.ox.ac.uk/~vgg/software/via/`, 2016.

[17] U. Singh, "Omrchecker: Tool for optical mark recognition alignment." `https://github.com/Udayraj123/OMRChecker`, 2021. Accessed: June 2025.