INK TO LATEX: A TRANSFORMER BASED MATH EXPRESSION LANGUAGE MODEL

Zirui Li, Jianxiong Li, Youjia Qian, Haozhuo Liu

ABSTRACT

This project aims to recognize hand-written math formulas and transform them into their equivalent LaTeXcode representation. The main challenge originates from analyzing the spatial relation of symbols. Traditional Optical Character Recognition (OCR) technology is applied to detect and extract text from images, but characters, especially characters with similar shapes, lead to uncertainty. Neural Networks have been developed. The space to improve, still, is enormous [1]. A transformer-based math language model (TMLM), developed by Ung al et [2], successfully generated output sequences, but the writing styles differ, which leads to relatively low accuracy. The transformer model developed in recent years significantly improved the OCR accuracy [1]. The attention mechanisms enable the model to understand the spatial relationships, especially in hierarchy and priorities.

Based on the previous studies, an improved model was developed. The model is divided into two stages. Firstly, the YOLO v8 model takes images containing handwritten formulas as input and outputs bounding boxes corresponding to each token. After training, its precision is over 99% on the test dataset. Secondly, the YOLO outputs are fed into a transformer model that predicts LaTeX sequence. Our code is publicly available at https://github.com/ZiruiLi-133/HE2L-Net.

Keywords: Handwritten Math Formula Recognition, YOLOv8, Transformer, Machine Vision, NLP

1. INTRODUCTION

The hand-written recognition technology has witnessed an unprecedented boom in the recent decade. Though it is still a challenging area of study, the total accuracy has been improved significantly thanks to the efforts in machine learning models and algorithms. State-of-the-art models, such as the ones based on Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and Long Short-Term Memory networks (LSTMs), have demonstrated high performance in popular competitions and datasets including the International Conference on Document Analysis and Recognition (ICDAR) [3], IAM Handwriting database [4], MNIST database. These models have over 90% accuracy and over 93% precision in recognition of single-line texts in various scripts and languages.

The main challenge comes from the inherently complex spatial relationship between the symbols that beyond the simple one-dimensional sequence of characters. Elements such as superscripts, subscripts, integrals, and matrices require not only the recognition of individual symbols but also an understanding of their spatial interrelations. These elements are often vertically aligned or grouped in non-standard layouts that traditional text recognition models handle poorly.

For instance, the performance of these models tends to degrade significantly when faced with tasks like recognizing whether a numeral is part of a fraction (either as a numerator or denominator) or identifying the scope of a square root symbol that stretches over several terms. The spatial arrangements essential to mathematics—where positioning defines semantic meaning—pose a substantial challenge. This is because standard handwriting recognition systems are primarily trained and tuned for linear text flows, typical of most written

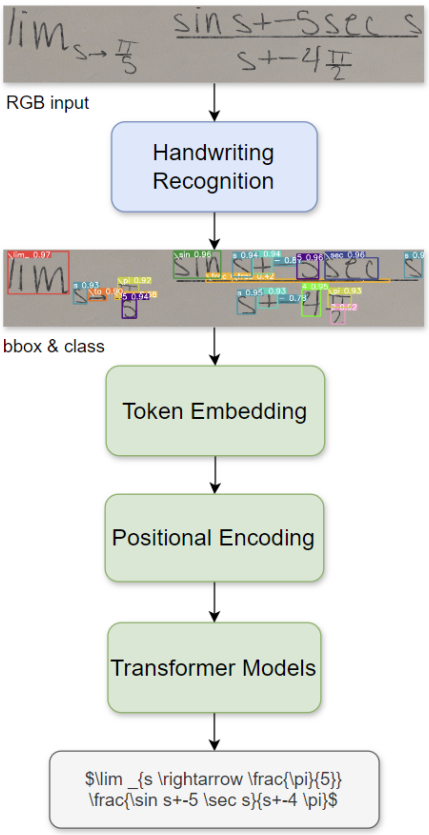

Fig. 1. Network Outline

languages.

Recognizing the limitations of existing handwriting recognition algorithms and the need for a tool to convert HME to LaTeX code, we propose a network with two stages: the handwriting symbol recognition stage and the LaTeX code composing stage, that converts an image input with HME onto a sequence of LaTex code. To summarize, our contribution includes:

1. Trained YOLOv8 on a dataset that includes HME under the topic of limit.
2. Utilizes token embedding and positional encoding to generate valid input for transformer models.
3. Construct a transformer-based math expression language model
4. Trained our model on the bounding box and token IDs.

A visualization of our network outline is shown in Fig 1.

2. RELATED WORK

Converting handwritten math expressions to LaTeX is a challenging problem that researchers have been dealing with. This section aims to provide a comprehensive overview of the related areas and main contributions in this field. Topics such as transformer-based language models, optical character recognition (OCR) techniques tailored for mathematical notation, the application of neural networks in recognizing mathematical expressions, and attention mechanisms in the domain of mathematical recognition will be included.

2.1. Optical Character Recognition for Mathematical Notation

Optical Character Recognition (OCR) has conventionally focused on detecting text in images, but recognizing mathematical notation is more complicated due to diverse symbols and logical structures within expression. Sun et al. [1] developed a transformer-based OCR system to identify text and math symbols, creating a dataset with labeled bounding boxes from arXiv papers. Orj et al. [5] improved OCR accuracy for converting images to LaTeX using active learning. Another model in [6] combines visual feature extraction with language modeling via a transformer-based OCR system, refining its recognition of handwritten and printed text through pre-training on large datasets.

2.2. Transformer-based Math Language Model

Handwritten mathematical expressions can be ambiguous due to the similarity between writing styles. To address this issue, Ung et al.[2]propose a transformer-based math language model (TMLM) to understand the relationships between tokens in a sequence. The model, trained on approximately 70,000 LaTeX sequences from CROHME 2016, achieved dramatic results with a perplexity of 4.42, outperforming traditional previous network-based models. This model was combined with a stochastic context-free grammar-based HME recognition system to improve recognition rates on the CROHME 2016 and CROHME 2019 datasets.

2.3. Attention Mechanisms in Math Recognition

Attention mechanisms, embedded within transformer models, have significantly enhanced the recognition of mathematical notation. The location-guided transformer in [1] improves OCR accuracy by focusing on relevant areas of an image. It allows the model to better understand spatial relationships which play an important role in mathematical expressions. Attention mechanisms facilitate the hierarchical processing of mathematical expressions and enable the model to prioritize key elements and relationships.

3. METHODOLOGY

3.1. Handwriting Recognition

The handwriting recognition stage is the basis for the network. As previously mentioned, the spatial relationship between the characters in the image is crucial for our task. Therefore, traditional models and OCR methods which output a sequence of characters are not suitable for the task. Instead, the task of object detection is more close to the goal.

State-of-the-art object detection models such as EfficientDet and YOLOv5 predicts the bounding box of an object from predefined anchor boxes. Since these boxes are predefined with fixed size, it complex the optimization of hyperparameters. Moreover, if an object's size or aspect ratio falls significantly outside these predefined anchor boxes, it can be harder for the model to accurately detect and bind the object. This is particularly relevant for objects like fraction bars, which can vary widely in length depending on the number of terms they span. Therefore, we use YOLOv8 in our network to take advantage of its anchor-free feature.

3.2. Transformer Architecture

3.2.1. Token Embedding

This step transfers the bounding boxes of every token in each input image into vectors of continuous form so that they can be fed to the encoding to be processed. The number of nodes in the embedding layer was selected to be 512, so the token IDs, which represent the input tokens, were transformed into 512-sized vectors. That is, each input token was mapped to its unique vector through the embedding layer.

To make the model understand the spatial relationship of each character in an image, the relative position of each to-

ken's bounding box in each image was directly combined with the embedding input. A linear layer was created so that the 4-sized bounding box coordinate could be mapped to the 512-sized embedding layer. Through this projection from low to high dimension, the spatial information was fed to the input simultaneously with the related token information. During forwarding, they are directly summed together and then go into the positional encoding layer.

3.2.2. Positional Encoding

Combining the bounding box coordinates directly to the embedding layer with input tokens makes the model understand the characteristics of the tokens and their positional information together. Positional encoding makes the model further understand the order of a series of tokens. It is a crucial step for the transformer because the transformer does not process the order of input intrinsically.

Positional encoding adds another vector to the vector from the input embedding layer. A tensor was generated with step 2 to represent the even embedding dimension positions, and then it was put in a sine function. Another tensor, similarly, was generated and put in a cosine function for odd. A scaling factor was computed based on the 512 dimensions to make the frequency go smoothly. The encoding, then, adds the sine and cosine functions to the encoding vectors.

3.2.3. Encoder & Decoder

The encoder and decoder play a critical part in this transformer model. They take in vector sequences from the embedding layers and generate Latex code sequences as output.

The encoder was constructed in multiple layers using nn library, and they are stacked to compose the transformer encoder. The goal of each encoder layer is to apply self-attention and multi-head attention. The multi-head attention is key to making the model focus on different aspects of the input vector at the same time. It enables the model to understand the context because different parts of the mathematical tokens have various influences on other parts including relative position and functions in the sequence. Self-attention enables the model to focus on some of the symbols that have a huge effect overall, such as "lim", "frac", and operators. Then, the values in the networks were fed forward to the next part of the model.

The decoder, similarly, was composed of multiple layers and then stacked into the transformer decoder in nn library. Masked multi-head attention was applied so that a given token depends on the previously generated token. Then, multi-head attention was applied to combine the encoder output to highlight each individual position in the decoder relative to all positions of the input vector. Then, the values were fed forward.

3.2.4. Masked multi-head self-attention.

The self-attention mechanism is vital for understanding both spatial and sequential relationships within mathematical expressions. Even though the self-attention mechanisms are not explicitly outlined in our code, they are effectively implemented within the nn.TransformerEncoderLayer and nn.TransformerDecoderLayer from PyTorch. These layers efficiently compute the attention scores, which enables the model to dynamically focus on the most relevant parts of the input data. The computation involves converting input embeddings—which combine token data, bounding box information, and positional encodings—into sets of queries, keys, and values. These elements are then passed through dot products and softmax operations, a method core to the transformer architecture introduced in the groundbreaking paper by Vaswani et al[7]. Additionally, the mechanism incorporates masking within each attention layer, which prevents each token from attending to subsequent tokens in the sequence. This masking is necessary to ensure that the prediction for each token can only depend on known outputs of previous tokens. The resulting contextual understanding from these combined features allows the model to effectively decode complex relationships like hierarchical mathematical structures and symbol connections.

4. EXPERIMENTS

4.1. Dataset

Our primary dataset is the Aida Calculus Math Handwriting Recognition Dataset [8], which consists of 100,000 images. Each image contains a photo of a handwritten calculus math expression (specifically within the topic of limits) written with a dark utensil on plain paper. Each image is accompanied by ground truth math expression in LaTeX as well as bounding boxes, which satisfies our requirement for training YOLOv8 object detection and transformer for composing LaTeX code. We randomly separate the 100,000 images with a ratio of 6:2:2 into training, validation, and test sets.

4.2. Dataset Preparation

To standardize the format of the dataset for this machine vision-oriented project, COCO (Common Objects in Context) format was applied. This format organized data into these three sections: Images info, annotations, and categories. Images info includes the list of images being processed, annotations links to these images and describes all bounding boxes and their corresponding token IDs for each image, and categories map all unique token IDs to their actual Latex token names. The COCO format is than converted to YOLO format for YOLOv8 training.

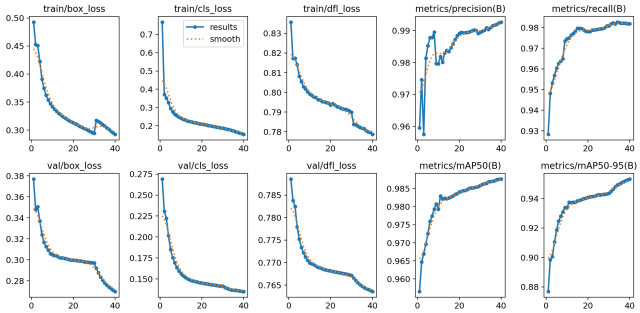

Fig. 2. YOLOv8 Training Losses & Evaluation Metrics

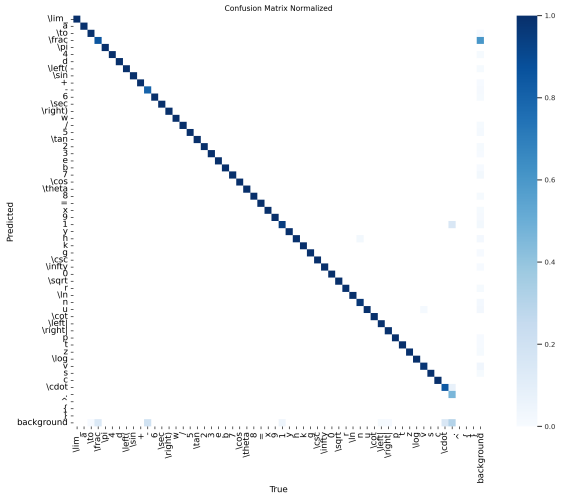

Fig. 3. Normalized Confusion Matrix of YOLOv8

4.3. YOLOv8 Handwriting Recognition Results

We trained YOLOv8 on the training dataset and validated it on the validation dataset for 40 epochs. The key evaluation metrics for the object detection task is Mean Average Precision (mAP), specifically mAP50 and mAP50-95. mAP50 refers to the mean average precision at an (Intersection over Union) IoU threshold of 0.50. This means that a predicted bounding box is considered correct if it has an IoU of 0.50 or more with the ground truth box. mAP50-95 averages the mAP calculated at different IoU thresholds, from 0.50 to 0.95 (inclusive), typically at intervals of 0.05. This means calculating the mAP at IoU thresholds of 0.50, 0.55, 0.60, ..., up to 0.95.

Fig 2 shows the training losses and the evaluation metrics on the validation set in each epoch. The curves show a nice gradual decrease of training loss and a gradual increase of mAP50 and mAP50-95 metrics.

Fig 3 shows the confusion matrix of the YOLOv8 on the test dataset. Table 1 shows the YOLOv8's detection metrics on the test dataset. According to the figure and table, YOLOv8 correctly detects and classifies all of the symbols included in the dataset, even for ambiguous symbols such as short fractions and minus signs.

Table 1. YOLOv8 Result on Test Dataset

mAP50	mAP50-95	Precision	Recall
0.98	0.95	0.99	0.98

4.4. LaTeX code composing

The evaluation metrics employed are Precision, Recall, and F1 score. The F1 Score is used as it combines precision and recall, the F1 score provides a measure to balance both the precision and recall of the model, reflecting how well the model is performing by considering both false predictions and missing symbols. After training our model for 100 epochs,

Table 2. Transformer Result

Precision	Recall	F1
0.20	0.07	0.06

we got the results shown on the table 2. The result is not good enough to produce the desired LaTeX code sequence. Potential reason may include

- The number of epochs is not enough
- The padded tokens are not ignored included when doing prediction and when calculating loss
- Our transformer model need to be improved to better fit this task

5. CONCLUSION AND FUTURE WORK

This paper presented how to transform handwritten mathematical expressions into their corresponding LaTeX code equivalents using advanced machine-learning techniques. The method employs YOLOv8 for handwriting recognition and a custom-designed transformer model for generating accurate LaTeX code. It effectively interprets the spatial relationships among the detected symbols to generate LaTeX code.

There are a few areas that still require improvement. Firstly, the dataset exhibits bias, with some symbols appearing frequently while others appear infrequently. Enriching the dataset source by collecting from various open sources on the internet could help address this issue. This bias may hinder the model's ability to grasp the spatial relationships between symbols fully. Secondly, we should delve into studying and modifying our transformer layers for better optimization.

6. ACKNOWLEDGEMENT

In this project, all four group members, led by Zirui Li, contributed to its success. We extend our gratitude to everyone involved for our collective efforts across various aspects of the work.

In addition, we would like to express our sincere gratitude to Prof. Duan and all the teaching assistants. We thank Prof. Duan for equipping us with the essential tools for conducting this project and for the invaluable insights into machine learning. Additionally, we appreciate the valuable suggestions provided by the teaching assistants throughout the time.

7. REFERENCES

[1] Yu Sun, Dongzhan Zhou, Chen Lin, Conghui He, Wanli Ouyang, and Han-Sen Zhong, "Locr: Location-guided transformer for optical character recognition," 2024.

[2] Huy Quang Ung, Cuong Tuan Nguyen, Hung Tuan Nguyen, Thanh-Nghia Truong, and Masaki Nakagawa, "A transformer-based math language model for handwritten math expression recognition," in *Document Analysis and Recognition – ICDAR 2021 Workshops*, Elisa H. Barney Smith and Umapada Pal, Eds., Cham, 2021, pp. 403–415, Springer International Publishing.

[3] "The 17th international conference on document analysis and recognition," `https://icdar2023.org/`, Accessed: 2024-5-4.

[4] U-V Marti and Horst Bunke, "The iam-database: an english sentence database for offline handwriting recognition," *International Journal on Document Analysis and Recognition*, vol. 5, pp. 39–46, 2002.

[5] Everistus Zeluwa Orji, Ali Haydar, İbrahim Erşan, and Othmar Othmar Mwambe, "Advancing ocr accuracy in image-to-latex conversion—a critical and creative exploration," *Applied Sciences*, vol. 13, no. 22, 2023.

[6] Minghao Li, Tengchao Lv, Lei Cui, Yijuan Lu, Dinei A. F. Florêncio, Cha Zhang, Zhoujun Li, and Furu Wei, "Trocr: Transformer-based optical character recognition with pre-trained models," *CoRR*, vol. abs/2109.10282, 2021.

[7] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, L ukasz Kaiser, and Illia Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds. 2017, vol. 30, Curran Associates, Inc.

[8] Aida by Pearson, "Aida calculus math handwriting recognition dataset," Aug. 2020.