# MEDICAL TRANSCRIPTION: HANDWRITTEN WORD RECOGNITION MODEL WITH CNN-RNN-CTC

**Si Han Xiong**
# 1008935404
`sihan.xiong@mail.utoronto.ca`

**Zichen Liu**
# 1008789152
`zach.liu@mail.utoronto.ca`

**Hyunjun (Hayden) Choi**
# 1007796173
`hyj.choi@mail.utoronto.ca`

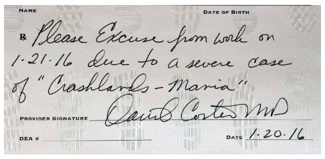
**Seunghyun (Joe) Lee**
# 1006882651
`sehyun.lee@mail.utoronto.ca`

## ABSTRACT

This project aims to develop a deep learning system to recognize handwritten English words from physcian' notes using a Convolutional Recurrent Neural Network (CRNN) architecture. Leveraging the IAM Handwriting Database, we preprocess grayscale word images and train a hybrid model composed of CNN layers for spatial feature extraction, a BiLSTM for sequence modeling, and a Connectionist Temporal Classification (CTC) loss layer for alignment-free transcription. Our model outperforms the rule-based Tesseract OCR baseline, particularly in handling cursive and varied handwriting styles. We evaluate performance using Word Accuracy, Character Error Rate, and Character Accuracy, and observe signs of overfitting, suggesting directions for further regularization and data augmentation. This system highlights both the potential and limitations of applying modern deep learning to critical domains like medical documentation, where interpretability and accuracy are essential. —-Total Pages: 9

## 1 INTRODUCTION

Despite the growing trend of electronic medical documentations by physicians (e.g., general practitioners, doctors, and surgeons), 20% of physicians still use handwritten documentation as seen in Figure 1 (O'Donnell, 2008). While this may not seem like a potential issue, a study concluded that out of 190 operative surgical notes audited for identity details, preoperative diagnoses, operation details, and postoperative instructions, only 42% were completely and entirely legible (ISQua, 2012). This suggests a risk for patient safety given the use of handwritten notes within healthcare to an extent human training may not be able to solve. Deep learning techniques such as Convolutional Neural Networks (CNNs) have seen advancements in image feature extraction—with some models being able to achieve a 99% word accuracy in handwriting recognition (Wiles, 2019). This is a stark improvement compared to the 42% of legibility among humans. Thus, within this project, we will aim to apply deep learning techniques for handwriting transcription within a patient-physician healthcare setting.



NAME DATE OF BIRTH

℞ Please Excuse from work on 1·21·16 due to a severe case of "Crashlands-Maria"

PROVIDER SIGNATURE [illegible] MD

DEA #

DATE 1·20·16

Figure 1: Example Physician Handwriting Documentation.

## 2 BACKGROUND & RELATED WORK

The challenges of deciphering handwritten medical notes have been approached from practitioner training all the way to machine learning techniques. On the practitioner end, some medical professionals have reported enrolling in handwriting training programs for medical professionals with one notable resource being the "Write Now: The Getty-Dubay Program for Handwriting Success" (Getty & Dubay, 2003).

On the other hand, machine learning efforts within CNNs, RNNs, and LSTMs have been largely depended on for their ability to identify text and extract useful information. Recent work on Optical Character Recognition (OCR) published from Jadavpur University proposes a three-layered CNN using a Connectionist Temporal Classification (CTC) loss function to recognize a sequence of alphabetic characters which was then matched with a medical dictionary (Dhar, 2020). According to Stanford University, CTC helps us align inputs and outputs by using an output distribution of all possible outputs, then using this distribution to infer the likely output with a conditional probabilistic loss function (Hannun, 2017). So far, we've mostly seen sole CNN methods with a word accuracy ranging around 60% (Shahade, 2021).

Moreover, research published in the International Research Journal of Modernization Engineering Technology that proposed a model including RNN methods increased word accuracy by 15% compared to the sole CNN networks (Nayak, 2023). As a result, efforts have been now transitioned to hybrid CNN-RNN methods. A study by Beijing Jiaotong University proposed using a Res-Net CNN model with residual architecture coupled with RNN which included the gating mechanism of Bidirectional LSTM–they saw a 5.3% increase of word accuracy from current benchmarks (Ma, 2023). These previous works suggest that a combination of CNN, RNN, and CTCs could be a plausible solution to our medical handwriting transcription problem at hand.

## 3 DATA PROCESSING

### 3.1 SOURCES OF DATA

The dataset used within this project consisted of 44,564 images of handwritten English words from 500 authors along with their corresponding labels which was sourced from the IAM Handwriting Database (Marti & Bunke, 1999). The images are unconstrained and were scanned at 300 dpi resolution with 256 gray levels.

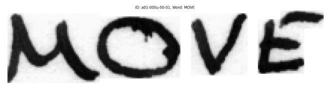


Figure 2: Example of the Handwritten Word 'MOVE' from the Dataset.

Exploratory Data Analysis (EDA) of the training data revealed that there were 5792 unique words, with most common words being "the," "of," and "to". All characters from the labels were extracted to create a character-to-index mapping consisting of 77 unique mappings, which includes both uppercase and lowercase alphanumeric characters, numbers and special characters [1]

Additionally, the word length distribution was approximately normal, with a mean word length of 6.9 characters and a maximum length of 19 characters as seen in Figure 3.

Following the initial EDA, 6259 images in the dataset that had null labels were removed due to the abundance of available data. After removing all null labels, the dataset was split into training, validation, and test sets in an 8:1:1 ratio.

### 3.2 IMAGE PROCESSING

The image transformation process is illustrated step by step in Algorithm 1, with a corresponding visual representation in Figure 4.

[1]The 77 Mappings include: !"&'()*+,-./0123456789:;?ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz

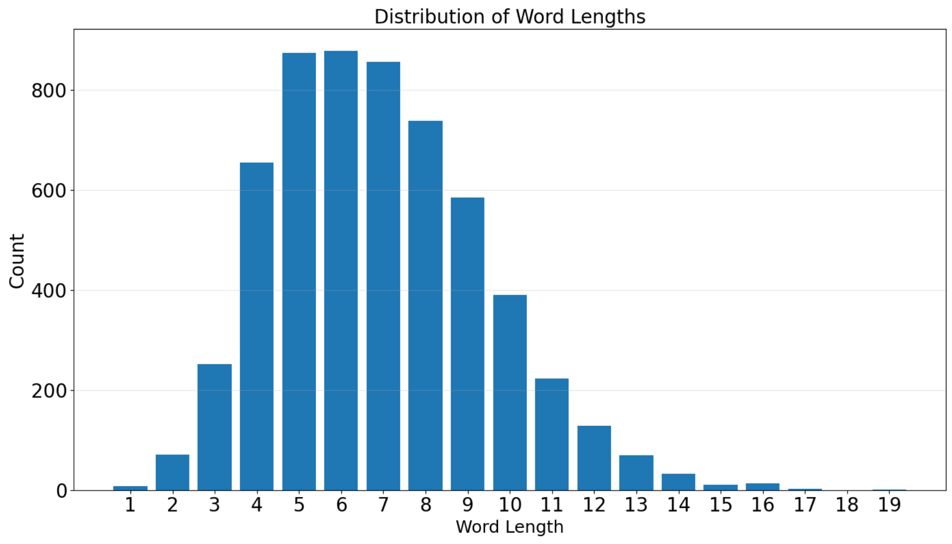


Figure 3: Word Length Distribution in Training Set

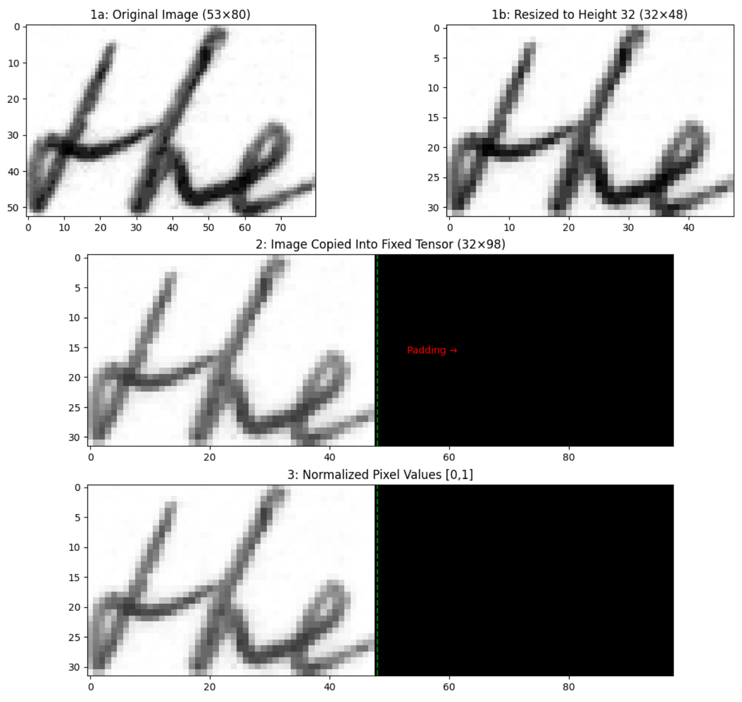

1a: Original Image (53×80)
1b: Resized to Height 32 (32×48)
2: Image Copied Into Fixed Tensor (32×98)
Padding →
3: Normalized Pixel Values [0,1]

Figure 4: Image Processing Steps Visualization.

## 4 Model Architecture

Our final model for medical handwritten text recognition uses a hybrid of the CNN and RNN architecture–also known as a Convolutional Recurrent Neural Network (CRNN) (Keren & Schuller, 2017). As seen in Figure 5, the model input is a grayscale image of fixed height 32 pixels, with variable width depending on the maximum word length of each batch. The Convolutional layers within the model extracted spatial features from word images and passed them onto the recurrent network layers, which then captured temporal dependencies across character sequences. Then, a CTC-based

**Algorithm 1** Process Image for Batch Input with Dynamic Width

```
1: **Input:** Original image $I$ of size $(H, W)$, Maximum batch width $W_{max}$
2: **Output:** Resized and normalized image $I'$ of shape $(1, 32, W_{max})$
3: **Step 1: Resize Image**
4: Set new height: $H' = 32$
5: Calculate new width: $W' = \frac{H'}{H} \times W$
6: Resize $I$ to $(H', W')$ while maintaining aspect ratio
7: **Step 2: Copy Resized Image**
8: Initialize an empty tensor $I'$ of shape $(1, 32, W_{max})$ filled with white pixels and copy resized
   image $I$ into the left of $I'$
9: **Step 3: Normalize Pixel Values**
10: Scale pixel values to range [0, 1]
11: **Return** Processed image $I'$
```

output stage converts the sequential representations into alignment-free transcription of handwritten text.

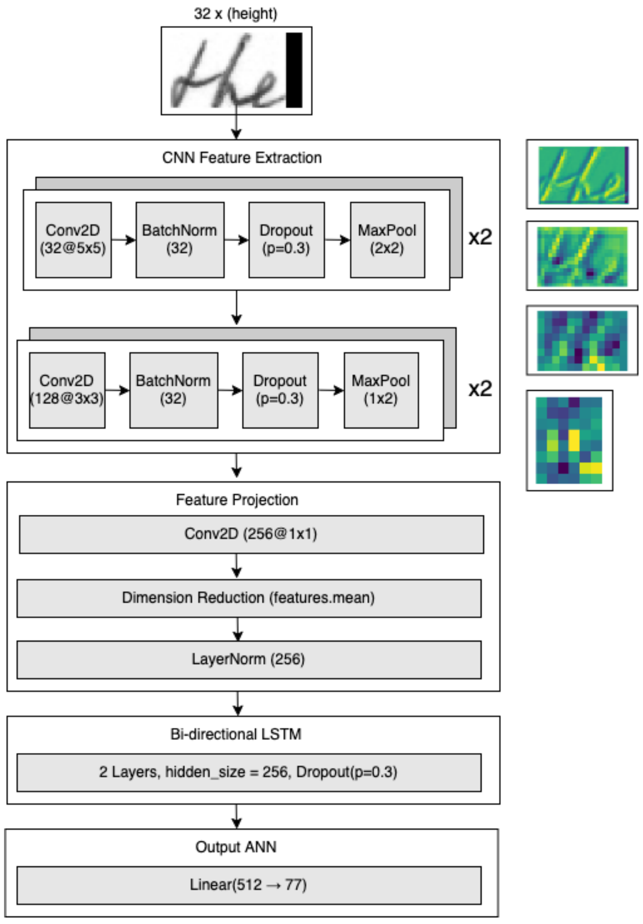


Figure 5: Final Model Visual Representation.

The convolutional backbone consists of four Conv2D layers as follows:

- Layer 1: `Conv2D(1, 32, kernel_size=5, stride=1, padding=2)`
- Layer 2: `Conv2D(32, 64, kernel_size=5, stride=1, padding=2)`
- Layer 3: `Conv2D(64, 128, kernel_size=3, stride=1, padding=1)`

- Layer 4: `Conv2D(128, 256, kernel_size=3, stride=1, padding=1)`

Each block includes batch normalization, a ReLU activation function, dropout with $p = 0.3$, and max pooling:

- Max pooling after layers 1 and 2: `kernel_size=(2, 2), stride=(2, 2)`
- Max pooling after layers 3 and 4: `kernel_size=(1, 2), stride=(1, 2)`

This architecture includes max pooling, which reduces spatial dimensions to help prevent overfitting. One important feature to note is the use of asymmetrical pooling in the later blocks to preserve vertical information, which is critical for character recognition.

Before passing features to the Recurrent layers, a $1 \times 1$ convolution and layer normalization was applied for feature projection – independently processing features while preserving spatial information and learning cross-channel patterns through 256 weighted mixtures:

- `Conv2D(256, 256, kernel_size=1, stride=1)`
- `LayerNorm((256,))`

The 2D CNN output tensor of shape `(batch_size, channels=256, height, width)` is permuted to `(batch_size, width, channels, height)` and flattened to `(batch_size, width, channels×height)` so it can be fed into 1D Recurrent layers, where the width dimension is treated as the sequence of time steps.

Subsequently, a two-layer Bidirectional Long Short-Term Memory (BiLSTM) network Ghosh et al. (2022) is used to model temporal dependencies (see Figure 5):

- `LSTM(input_size=256, hidden_size=256, num_layers=2, bidirectional=True, batch_first=True)`

This produces an output of size 512 per time step (256 from the forward direction and 256 from the backward direction). By using a bidirectional network, the model is able to capture information from both past and future context.

Finally, the BiLSTM output is passed through a dropout layer and a fully-connected layer, as demonstrated in Figure 5:

- `Dropout(p=0.3)`
- `Linear(512, 77)`

This maps each time step to logits over 77 character classes, alongside with a blank token used for the CTC loss. The output tensor is permuted from `(batch_size, width, num_classes=77)` to `(width, batch_size, num_classes=77)` to match the input requirements of the CTC.

The model is trained using the Connectionist Temporal Classification (CTC) loss, which enables alignment-free training by allowing the model to learn to predict sequences without requiring explicit character-level segmentation (Hannun, 2017). During training, we use the Adam optimizer with an initial learning rate of 0.001:

```
optimizer = torch.optim.Adam(model.parameters(), lr=0.001)
```

The CTC loss is implemented as follows:

```
ctc_loss = CTCLoss(zero_infinity=True)
criterion = ctc_loss(logits, targets, input_lengths, target_lengths)
```

This training setup allows the model to robustly handle variations in handwriting length, spacing, and style.

## 5 BASELINE MODEL

To evaluate our CNN-RNN-CTC model, we chose Tesseract OCR as a baseline (Google, 2025). Tesseract is a widely used, open-source text recognition tool developed by Google that does not rely on deep learning; instead, it uses fixed, rule-based methods to recognize characters. This makes it a simple and appropriate benchmark for comparison. Tesseract can be run immediately without any training, which fits the rubric's requirement for a straightforward, hand-coded baseline. It also works reasonably well on clean, segmented words and is easy to reproduce on any system. Although it struggles with cursive and messy handwriting, this actually highlights the strengths of our model, which is designed to handle more variation and complexity. By comparing our results to Tesseract's, we get a better sense of what deep learning contributes and how much more adaptable it can be in real-world scenarios.

To evaluate Tesseract as a baseline, we ran the following setup:

**Input:** Preprocessed grayscale images of single handwritten words. Each image is resized so that the height is fixed at 32 pixels while the width is adjusted proportionally to preserve the aspect ratio (resulting widths may vary rather than being exactly 128 pixels).

**Command:**

```
tesseract input_image.png output --psm 7
```

The `--psm 7` flag configures Tesseract to treat the image as a single line of text, appropriate for word-level prediction.

**Output:** Text predictions are stored in `.txt` files. These predictions are compared against the ground truth using standard metrics such as Character Error Rate (CER) and Word Accuracy.

## 6 QUANTITATIVE RESULTS

Table 1: Evaluation Metrics for Training, Validation, and Test Sets.

| **Set** | **Total Samples** | **Word Accuracy** | **Avg. CER** | **Character Accuracy** |
|---|---|---|---|---|
| *Training* | 30,643 | 0.8588 | 0.0365 | 0.9346 |
| *Validation* | 3,830 | 0.6454 | 0.1387 | 0.8047 |
| *Testing* | 3,830 | 0.6580 | 0.1306 | 0.8199 |

Table 1 presents the quantitative results of the model's performance based on three key metrics: Word Accuracy, Average Character Error Rate (CER), and Character Accuracy. Word Accuracy measures the proportion of correctly predicted words, meaning every character in a word must be correct for it to count as accurate. For example, a validation word accuracy of 0.6454 indicates that out of 3,830 words, 2,473 (64.54%) were predicted correctly. CER represents the proportion of incorrect characters compared to the total number of characters in the truth labels. Character Accuracy, on the other hand, measures the proportion of correctly predicted characters, considering only one-to-one matches between the ground truth and the prediction. While CER and Character Accuracy have an inverse relationship, they do not always sum to 1. This discrepancy arises because CER accounts for insertions and deletions, which can alter the total number of predicted characters, whereas Character Accuracy strictly evaluates correct matches without considering extra or missing characters. As a result, the model may predict more or fewer characters than expected, affecting CER but not directly influencing Character Accuracy.

As seen in Table 1, the performance of the model declined in the testing and validation set compared to the training set. This performance gap suggests that the model may be overfitting to the training data, limiting its ability to generalize effectively to unseen examples.

## 7 QUALITATIVE RESULTS

Additionally, for the Qualitative results as seen in Section 7, we observed that our model performed differently based on the type of handwriting and context of the word. To begin, our model performed well on handwriting fonts similar to sans serif (such as the word “nursemaid” in Figure 5) which are characterized by detached letters and the absence of small decorative strokes often seen within cursive writing (such as the word “premiere” in Figure 5) Creatopy (2021). We hypothesize this challenge to be due to the nature of cursive, which includes multiple letters blended into a single stroke in addition to variable spacing, which might confuse CTC’s blank-label alignment strategy Wang (2021). This hypothesize can be seen where the word ”Premiere” was mistaken to ”Premiue” where the ”er” may look similar to ”u” given the cursive handwriting style. Moreover, we observed that the length of the word didn’t affect the model performance nor did the model performance consider the context of the word. This is due to our proposed model not considering the context and meaning of the word but rather only focusing on character recognition.

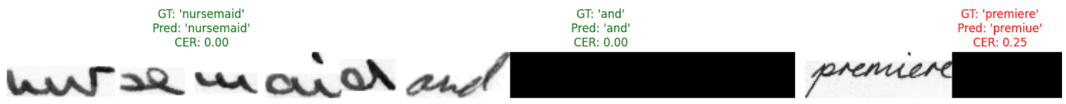


Figure 6: Model Predictions Across Various Handwriting Styles.

## 8 NEW DATA EVALUATION

To ensure that the model performance results during our training are a good representation of the performance on unseen data, we made efforts to fine-tune our model for better generalization. To begin, we included dropout (0.3) to encourage the model to learn redundant representations for better generalization (Science, 2023). In addition, we reduced the number of RNN layers to a total of 2, given that the unseen data we will evaluate often consists of short sequences, such as medication names, brief instructions, and dosage recommendations. As a result, we do not require additional RNN layers to capture extensive sequential data, leading to a simpler model with fewer chances of overfitting when exposed to new data.

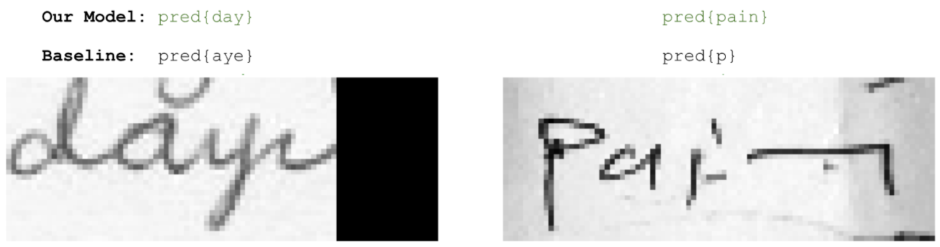


Figure 7: Example Model Predictions on Samples from New Data.

Table 2: Evaluation Metrics for Comparing Baseline and Model.

| Set | Total Samples | Word Accuracy | Character Error Rate |
|---|---|---|---|
| *Baseline* | 10 | 0.0000 | 0.9118 |
| *Model* | 10 | 0.6000 | 0.2188 |

To expose our model on new data, we populated an evaluation dataset of 10 images (words) with similar characteristics as handwritten notes within a healthcare application. This included different kinds of handwriting (eg. cursive, non-cursive), different resolutions of writing utensil on different backgrounds similar to variations in physician notes, in addition of including common vocabulary seen in physician's notes such as "pain" and medication usage frequencies (eg. "days") as seen in Figure 7. All data was processed the same way as training and validation data as detailed in Algorithm 1 to reduce variability within results. Additional evaluation data that were considered included short similar abbreviations (eg. "mg" vs. "mcg") as often seen within medical notes to ensure our model performs well given application context.

The results of evaluating our model and our baseline on the evaluation dataset can be observed in Table 2. Given the valuation dataset including characteristics of physician notes of diverse handwriting types and medical jargon, our rule-based baseline model performed poorly due to these variations. On the other hand, our model was able to perform better on these characteristics given our fine-tuning. Moving forward, when we want to deploy our model within healthcare applications, we will ensure that we prioritize exposing our model on medical specific jargon and abbreviations.

## 9 Discussion

Given our Quantitative results as seen in Section 6, our model had a training performance similar to existing CNN-RNN-CTC models on handwriting with a "Word Accuracy" of 83.5%-90% Hemanth (2021). Additionally, our model outperformed the baseline model as seen in the Evaluation Section 5[2] On the other hand, Word Accuracy on the test set performed around 10-15% below current CNN-RNN-CTC models mentioned above. We hypothesize this could be due to existing models training with an additional dataset called RIMES containing 12,000+ pages of handwriting from volunteers TEKLIA (2021). This additional data used by other models during training can lead to better generalization in the inference stage compared to our proposed model.

Another factor worth noting is the effect of word length. As shown in Figure 3, the average word length in the IAM dataset was approximately 6.9 characters, with most words falling within a 5–8 character range. We initially hypothesized that longer words might result in higher error rates due to alignment challenges in the CTC decoding process. However, our results did not show significant performance drops based on word length alone. This is likely due to the relatively short and consistent sequence lengths within our dataset, which fall well within the modeling capacity of the BiLSTM layers used in our architecture. Thus, while word length did not emerge as a limiting factor, the diversity in handwriting style—especially cursive writing and stylistically merged letters—remained the most prominent contributor to misclassification, as seen in Figure 6.

Furthermore, our evaluation on new, domain-relevant handwritten data (Figure 7) emphasized the model's limitations when encountering unfamiliar handwriting styles or domain-specific terminology such as "mcg" or "q.d.". While the model still outperformed Tesseract OCR, it struggled with abbreviations and specialized terms that were not well represented in the IAM dataset. This suggests that despite our model's visual feature extraction capabilities, it lacks contextual understanding or semantic grounding—a limitation also seen in other CTC-based systems (Wang, 2021; Ghosh et al., 2022).

### What We Learned

Through this project, we developed a deeper understanding of what it takes to build OCR systems that work beyond controlled environments. Technically, we became more confident in designing and debugging hybrid deep learning models, particularly those combining CNNs for spatial feature extraction with RNNs for sequence modeling. Implementing CTC loss forced us to think more carefully about how sequence alignment works in models that don't rely on explicit segmentation, and why this approach is necessary for transcribing unconstrained handwriting.

Working with the IAM dataset also made us more aware of the importance of preprocessing—something that's often underestimated. Handling variable word lengths and inconsistent image shapes through dynamic resizing and padding (Section 3.2) was a key part of making our model architecture viable. We also experienced the trade-off between generalization and overfitting: while our model performed well on training data, the noticeable drop in validation and test accuracy (Table 1) reminded us that strong training performance is not enough. Regularization techniques like

[2]Training, Validation, and Testing evaluation metrics will not be compared for our baseline given Tesseract's requirement for manual input and labeling.

dropout and pooling helped, but we realized that overfitting can still persist when training data lacks enough stylistic or domain diversity.

One of the most valuable takeaways, however, was that accuracy is not always the best proxy for usefulness—especially in high-stakes settings like healthcare. While the model handled general handwriting reasonably well, its performance degraded on handwritten medical terms, abbreviations, and cursive styles not well-represented in the training set (Section 7, Figure 7). This underscored the practical gap between academic performance and real-world readiness. We came to understand that a system intended for clinical use needs more than just a high test score: it needs to perform reliably across edge cases, domain-specific inputs, and ambiguous handwriting styles.

### Future Directions

To enhance our model's performance and applicability in medical handwriting recognition, we propose the following directions:

1. **Domain-Specific Data Collection or Augmentation:** Our current training set (IAM) lacks the specialized vocabulary, abbreviations, and unique handwriting styles seen in clinical documentation. To bridge this domain gap, we suggest collecting handwritten medical notes or using augmentation techniques tailored to the healthcare context. Prior work shows that domain-specific datasets and augmentations—such as elastic distortion, rotation, or artificial cursive generation—can significantly improve OCR performance in specialized settings (Bahl et al., 2021).
2. **Context-Aware Decoding:** Our decoder currently relies solely on CTC without leveraging contextual information. Future iterations should integrate a character-level language model trained on clinical corpora or constrain predictions with a medical lexicon. These enhancements can guide decoding toward more plausible outputs, especially when visual input is ambiguous (Ulhas & Ramanathan, 2022).
3. **Scaling Beyond Word-Level Transcription:** While our model currently transcribes isolated words, extending it to handle full lines or entire notes would improve its usefulness. Document-level transcription enables context-driven disambiguation between similar-looking words and helps preserve the semantic structure of medical text (Zhang & Liu, 2023). Such expansion would also better reflect real-world physician notes.
4. **Error Characterization and Targeted Improvements:** During qualitative evaluation (Figure 6), we noticed consistent failure patterns with cursive handwriting, closely merged characters, and uncommon abbreviations. A more formal error analysis pipeline—using confusion matrices or clustered error types—would allow us to categorize these errors and guide augmentation or architecture-level fixes (Xu & Patel, 2024).

Overall, this project demonstrated that building a model that works is only part of the challenge. Ensuring that it generalizes well, aligns with real-world needs, and performs consistently across data variations is just as important—especially in applications like healthcare where misinterpretation can carry real consequences.

## 10 Ethical Considerations

To begin, incorrect outputs –especially in the context of medications, dosages, or patient instructions– could lead to serious detriments. For example, studies have shown that traditional handwritten prescriptions which were misread may have contribute to adverse drug events (Goundrey-Smith, 2013). Thus, an automated machine learning-based system (like this one) used without human validation could amplify these risks. This demonstrates the need for human supervision in clinical settings and the importance of integrating such systems with verification checks rather than solely relying on them.

The model also has technical limitations that lead to ethical concerns in a chained effect. Although the system currently recognizes individual word segments, its capabilities cannot be extended to situations with full lines or blocks of text. This segmentation can cause loss of context, which is crucial in medical notes. A fragmented understanding of a sentence could lead to misinterpretation of the original intention of a handwritten note.

## REFERENCES

Shreya Bahl, Vidit Agrawal, and Aastha Arora. Handwriting recognition using deep learning with effective data augmentation techniques. *ResearchGate*, 2021. URL `https://www.researchgate.net/publication/354244821`.

Creatopy. What is sans serif? a guide to font styles, 2021. URL `https://www.creatopy.com/blog/what-is-sans-serif/`.

Dibyajyoti Dhar. Hp docpres: A method for classifying printed and handwritten texts in doctor's prescription. *ResearchGate*, 2020. URL `https://www.researchgate.net/publication/345830022_HP_DocPres_a_method_for_classifying_printed_and_handwritten_texts_in_doctor's_prescription`.

Barbara Getty and Inga Dubay. *Write Now: The Getty-Dubay Program for Handwriting Success*. Handwriting Success, 2003. URL `https://handwritingsuccess.com/handwriting-for-adults`.

Soumitra Ghosh, Asif Ekbal, and Pushpak Bhattacharyya. Chapter 2 - natural language processing and sentiment analysis: Perspectives from computational intelligence. In *Artificial Intelligence for Internet of Things*, pp. 105–123. Elsevier, 2022. doi: 10.1016/B978-0-323-90535-0.00007-0. URL `https://www.sciencedirect.com/science/article/pii/B9780323905350000070`.

Google. Tesseract ocr. `https://tesseract-ocr.github.io/`, 2025. Accessed: 2025-03-31.

Stephen Goundrey-Smith. *Information Technology in Pharmacy: An Integrated Approach*. Springer, London, 2013. ISBN 978-1-4471-2780-2. doi: 10.1007/978-1-4471-2780-2. URL `https://link.springer.com/book/10.1007/978-1-4471-2780-2`.

Awni Hannun. Sequence modeling with ctc. *Distill*, 2017. doi: 10.23915/distill.00008. https://distill.pub/2017/ctc.

G.R. Hemanth. Handwritten text recognition using cnn-rnn-ctc. *ICTACT Journal on Soft Computing*, 12(1):2457–2463, 2021. URL `https://ictactjournals.in/paper/IJSC_Vol_12_Iss_1_Paper_1_2457_2463.pdf`.

ISQua. Poor handwriting: Does it lead to medical error? *ISQua Blog*, 2012. URL `https://isqua.org/resources-blog/blog/poor-handwriting-does-it-lead-to-medical-error.html`.

Gil Keren and Björn Schuller. Convolutional rnn: an enhanced model for extracting features from sequential data. *arXiv preprint arXiv:1602.05875*, 2017.

Jun Ma. Hybrid cnn-rnn model for handwritten text recognition. In *Proceedings of International Conference on Pattern Recognition*, 2023. URL `https://www.scitepress.org/Papers/2023/128010/128010.pdf`.

U. Marti and H. Bunke. A full english sentence database for off-line handwriting recognition. In *Proceedings of the 5th International Conference on Document Analysis and Recognition*, pp. 705–708, 1999.

Neha Nayak. Enhancing ocr accuracy with rnns. *IRJMETS*, 2023. URL `https://www.irjmets.com/uploadedfiles/paper//issue_4_april_2023/37208/final/fin_irjmets1683528914.pdf`.

Heather C O'Donnell. Physicians' attitudes towards copy and pasting in electronic note writing. *National Library of Medicine*, 2008. URL `https://pmc.ncbi.nlm.nih.gov/articles/PMC2607489/`.

Towards Data Science. Dropout in neural networks. *Towards Data Science*, 2023. URL `https://towardsdatascience.com/dropout-in-neural-networks-47a162d621d9/`.

Makarand Shahade. Evaluation of cnn performance in ocr. *Journal of Digital Information Management*, 2021. URL https://www.dline.info/fpaper/jdim/v21i4/jdimv21i4_3.pdf.

TEKLIA. Rimes database: A large dataset for handwriting recognition, 2021. URL https://teklia.com/research/rimes-database/.

Prashanth Ulhas and Krishna Ramanathan. Towards context-aware language models for ocr. In *International Conference on Document Analysis and Recognition*, 2022. URL https://www.semanticscholar.org/paper/af1aa05f8e45de050cd1c519ae72cfb77229a268.

Tianwei Wang. Connectionist temporal classification for handwriting recognition. *arXiv*, 2021. URL https://arxiv.org/pdf/2106.05920.

Rachel Wiles. Have we solved the problem of handwriting recognition? *Medium*, 2019. URL https://medium.com/data-science/https-medium-com-rachelwiles-have-we-solved-the-problem-of-handwriting-recognition

Lin Xu and Rohan Patel. Characterizing errors in handwriting recognition for medical applications. *arXiv preprint arXiv:2411.17332*, 2024. URL https://arxiv.org/abs/2411.17332.

Wei Zhang and Ming Liu. Line-level handwriting recognition for context-rich medical notes. *Journal of Intelligent & Fuzzy Systems*, 45(2):1235–1247, 2023. URL https://content.iospress.com/articles/journal-of-intelligent-and-fuzzy-systems/ifs237135.