A Multi-Backbone Feature-Concatenation Framework for Handwritten Prescription Word Classification

Swetha V Padmavathi Polisetty, Deepthi Godavarthi*

School of Computer Science and Engineering, VIT-AP University, Amaravati, Andhra Pradesh, India

Received 01 May 2026; received in revised form 05 July 2026; accepted 10 July 2026

DOI: https://doi.org/10.46604/aiti.2026.16426

Abstract

This study proposes a multi-backbone deep learning framework for handwritten prescription word classification to support reliable prescription digitization. EfficientNet-B0, ViT-Base, and Swin-Base operate in parallel to extract complementary local, global, and hierarchical visual features from pre-cropped medicine-name images. The extracted features are concatenated and passed through fully connected and classification layers. Training incorporates mixup regularization, label smoothing, and a warm-up learning schedule. Experiments are conducted on the Doctor's Handwritten Prescription Bangladesh (BD) dataset, containing 4,680 images across 78 medicine-name classes. The framework is evaluated through five repeated runs, ablation studies, class-wise analysis, and comparisons with single- and dual-backbone baselines. It achieves an average accuracy of 88.44 ± 0.99% and a macro F1-score of 0.88 ± 0.01. Holm-corrected paired t-tests show significant improvements over EfficientNet-B0 and ViT-Base, whereas differences from Swin-Base and dual-backbone models are not statistically significant. These results demonstrate the value of combining complementary visual representations for robust prescription-word classification.

Keywords: Handwritten prescription word classification, prescription digitization, deep learning, healthcare automation

1. Introduction

Poor handwriting in medical prescriptions persists as a critical challenge. The issue poses challenges for both the patient and the pharmacist and remains a major concern regarding the convenience [1-2]. In addition, medical language, abbreviations, and variations in handwriting styles render interpretation difficult, especially when using automated software [3]. As medical prescriptions contain crucial information for the safe administration of medication, proper interpretation is vital. Nevertheless, handwritten prescriptions usually display poor legibility and variation [4].

The hazards of illegible prescriptions are pronounced in clinical settings, where errors in reading prescriptions directly affect medication safety and service quality [5-6]. Physicians working under time pressure often use cursive handwriting, abbreviations, and shorthand for prescriptions. While this improves work efficiency, it frequently reduces the readability of prescriptions, thereby increasing the likelihood of errors among pharmacists or patients [2, 7]. Several studies show the danger of prescription reading mistakes, such as dangerous cases of drug-dispensing and preventable errors of patients' safety [8-9]. One frequently cited case in Texas, caused by a dosage reading mistake, led to a patient's death [1].

In the face of such problems, healthcare systems began to employ digitalization techniques, enabled by advances in computer vision and machine learning algorithms [10-11]. Methods for automated handwriting recognition and information

* Corresponding author. E-mail address: deepthi.g@vitap.ac.in

extraction show high potential to increase the accuracy and efficiency of prescription analysis. Specifically, machine learning information extraction algorithms and optical character recognition (OCR) systems help reduce medication errors, enhance patient safety, and deliver healthcare services more efficiently [12]. At the same time, recent advances in deep learning algorithms demonstrate strong potential for visual representation and classification, increasing the likelihood of accurate classification of handwritten prescription words [6, 13-14]. Research results published in 2025 and 2026 have confirmed the continued interest in deep learning-based prescription analysis and highlighted the difficulties in interpreting handwritten prescriptions under noisy and visually ambiguous conditions [15-17].

Despite these advances, a noticeable research gap remains. Most previous techniques rely on OCR-based preprocessing or a single-model architecture that struggles to handle highly cursive handwriting, noisy backgrounds, and visual similarities among medicines. OCR-based methods suffer from error accumulation when dealing with low-quality images, non-standard layouts, and distorted character shapes. It should also be noted that different backbone models attend to different features of visual representations: local textures, global contextual relationships, and hierarchical spatial relations [13, 15-16]. Little attention is paid to investigating whether such diverse types of information can be directly combined by feature concatenation for handwritten prescription word classification. Specifically, the parallel combination of EfficientNet-B0, ViT-Base, and Swin-Base models remains insufficiently explored.

To address this research gap, this study proposes a multi-backbone feature framework for classifying handwritten prescription words. This study adopts EfficientNet-B0, ViT-Base, and Swin-Base, integrating them into a single parallel architecture to extract local, global, and hierarchical visual features from images of handwritten medicine words. Specifically, the local features are extracted using EfficientNet-B0, the global features using ViT-Base, and the hierarchical features using Swin-Base shifted-window attention. The extracted features will be concatenated and then passed through a projection head, which includes a fully connected hidden layer with ReLU activation and dropout. The presented framework aims to improve handwritten prescription word classification and the digitization process using the Bangladesh (BD)—Doctor's Handwritten Prescription dataset. The efficiency of the suggested framework is analyzed through repeated-run comparisons, ablation studies on the fully connected layer size and dropout rate, computational cost analysis, confusion matrix analysis, and performance analysis for each class. Fig. 1 shows an example of a handwritten prescription and its legibility issues.

The remainder of this paper is organized as follows. Section 2 reviews related work. Section 3 presents the methodology and framework design. Section 4 reports the experimental results and analysis. Section 5 discusses the findings and their implications. Section 6 concludes the paper and outlines future research directions.

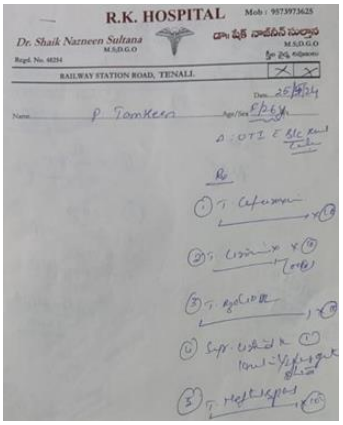


Fig. 1 Sample handwritten prescription

2. Related Work

2.1. OCR and deep learning-based prescription recognition

The recognition of handwritten prescriptions through the integration of OCR and deep learning techniques is widely

investigated, and continues to grow in popularity. In both early and relatively new research, methods such as bidirectional long short-term memory (Bi-LSTM), convolutional neural network (CNN), convolutional recurrent neural network (CRNN), and OCR-assisted frameworks perform tasks such as prescription digitization, medicine name extraction, and handwritten medical word classification. These studies demonstrate that deep learning improves not only classification but also extraction performance [6, 10-11, 13]. Other studies extend this trend further by utilizing localized prescription databases and OCR-assisted extraction pipelines; however, the problems remain the same: complex cursive handwriting, noisy scans, and peculiar prescription structure [5, 12, 14, 17]. Furthermore, recent studies reaffirms the relevance of deep neural network-based approaches for prescription recognition and digital record creation; nevertheless, the robustness of these techniques when operating under real-world prescription conditions remains questionable [15-16].

2.2. System-level and application-oriented prescription technologies

Other researchers explore prescription processing systems as well as application-driven technologies. Studies investigate the automation of prescription reading, handwriting recognition-based methods, scanner-based systems, and integrated application-based services for healthcare [18-19, 23]. In 2024, the application of deep learning, combined with YOLOv5 and an RNN, demonstrated that end-to-end multilingual prescription translation is feasible. This approach also satisfied the practical need for technologies that enable the automatic conversion of prescriptions regardless of writing style or language [23]. However, research using a multimodal deep learning approach—utilizing transformers for document understanding combined with sequence-to-sequence—demonstrated that multimodal systems usually outperform baseline systems based on a single model, such as LayoutLMv2 and DONUT [20]. Moreover, such systems incorporate explainable artificial intelligence (AI) mechanisms to enable clinicians to understand system operation and to gain an additional level of clinical interpretability.

Despite these advancements, many studies focus more on user-friendly transcription assistance or service integration rather than robust word classification in the presence of high cursive variability and inter-class similarity in handwritten medicines. The paper published in 2025 on AI integration with IoT, focusing on handwritten and multilingual prescription interpretation also pointed out that prescription recognition systems must take into consideration both accuracy and deployment practicality [27].

2.3. Digital prescription systems and structural understanding

Additional research addresses the process of generating an e-prescription, which includes, but is not limited to, document classification, structured data extraction, and prescription layout storage. The importance of recognizing both printed and handwritten text, as well as preserving structure, is evident in this area of study [5, 21-22]. Altogether, these findings suggest that an approach to processing handwritten prescriptions should incorporate structure modeling.

2.4. Contextual modeling and advanced deep learning models for handwriting prescription recognition

Recent investigations indicate that superior contextual model and a deeper learning architecture improves the performance of handwritten prescription recognition. Indeed, Bidirectional LSTM models, CRNN pipelines, OCR-integrated frameworks, and structure-oriented prescription recognition demonstrate the benefits of richer contextual modeling over character-based approaches, such as direct character generation [6, 8, 10, 13, 21]. Recent investigations into vision transformer (ViT) architecture show that ViT-based methods can model global contextual dependencies in handwritten words, often outperform conventional convolution-based methods on challenging datasets [24]. A recent Springer conference study explored handwritten prescription recognition using a deep learning framework optimized with stochastic gradient descent and adaptive momentum learning. This work reflects the continued interest in improving learning strategies for prescription-word recognition [26]. A hybrid approach integrating EfficientNet with ViT demonstrates that combining local feature extraction with global contextual awareness improve classification performance compared to using either architecture independently [25].

Nevertheless, evidence indicates that solutions built around a single model remain at risk, especially when handling highly cursive writing, symbol ambiguity, noisy frames, and irregularities in prescription layouts. This difficulty becomes pronounced when visually similar medicine names must be recognized reliably. Studies on the identical 78-class handwritten prescription dataset used in this work verify the findings. For example, a CNN-focused classification approach achieves 69.74% test accuracy, whereas a cascading lightweight hybrid adaptive attention model reaches 87.41%. These performance gap suggests that even architecturally advanced single-model methods struggle to distinguish between visually similar medicine names at the word level. Collectively, these results imply that context-aware representations perform optimally when paired with complementary feature-learning methods rather than when used in isolation [15-17, 28].

2.5. Research gap

As outlined in the Introduction, existing approaches rely on OCR preprocessing, single-model architectures, or workflow-oriented systems, which are not well-suited to robustly classifying visually similar handwritten words. Given this limitation, the present study proposes a multi-backbone feature-concatenation framework. It integrates EfficientNet-B0, ViT-Base, and Swin-Base, thereby enabling the capture of multiple visual cues, such as complementary local, global, and hierarchical features, for more robust prescription word classification, particularly under conditions of high character density.

3. Methodology

3.1. Overview of the proposed framework

This study proposes and investigates a multi-backbone feature-concatenation approach for the classification of handwritten prescription words. The framework consists of EfficientNet-B0, ViT-Base, and Swin-Base, which operate in parallel to generate complementary local, global, and hierarchical visual features. The feature vectors extracted from these three backbones are concatenated and subsequently passed through a fully connected and classification layers. This framework uniquely leverages the synergistic combination of these three visual backbones to enhance the classification of handwritten medical terms. The workflow comprises dataset preparation, label encoding, data augmentation, custom dataset preparation, parallel feature extraction, feature concatenation, classification, model optimization and evaluation. The architectural workflow is illustrated in Fig. 2, while the detailed training and evaluation procedure is presented in Algorithm 1 in Section 3.11.

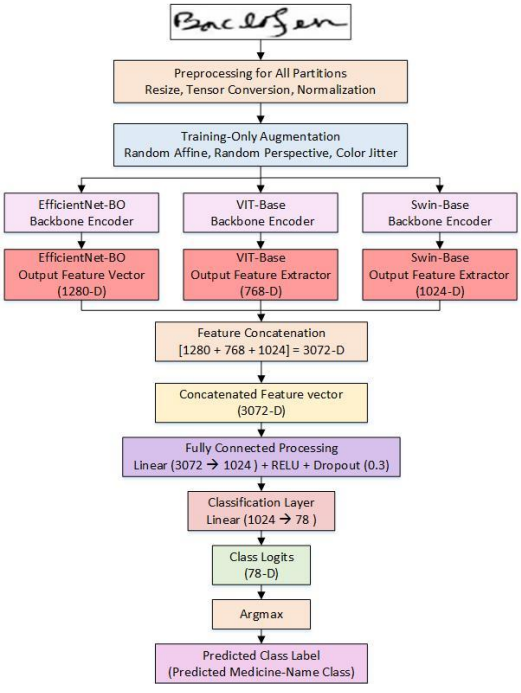


Fig. 2 Overall architecture of the proposed multi-backbone feature-concatenation framework

3.2. Dataset description and label encoding

These experiments are performed using the publicly available Doctor's Handwritten Prescription BD Dataset obtained from Kaggle [29]. The dataset, originally introduced by Mia et al. [14], contains images of handwritten medication names. The dataset comprises 4,680 images across 78 distinct medical classes. The pre-defined dataset partitions are maintained throughout this study. Specifically, the training partition has 3,120 images, while the validation and test partitions have 780 images each.

The dataset exhibits a balanced class distribution, with each of the 78 medicine-name classes containing 40 training images, 10 validation images, and 10 test images, totaling 60 instances per class. The samples consist of isolated, pre-cropped word-level images, each depicting a single handwritten medicine name. Consequently, text detection, word segmentation, and cropping are excluded from the scope of the study. The reported results reflect classification performance under the assumption that the medicine-name crop is available. Table 1 presents the dataset characteristics and partition details.

Table 1 Characteristics of the Doctor's Handwritten Prescription BD dataset

Partition	Number of images	Number of classes	Images per class	Split percentage
Training	3,120	78	40	66.67%
Validation	780	78	10	16.67%
Testing	780	78	10	16.67%
Total	4,680	78	60	100%

Let D denote the dataset collection of handwritten prescription word images and their corresponding class labels. If x_i denotes the i-th image and y_i represents its associated medicine-word label, the dataset is defined as the following set:

$$D = (x_i, y_i)_{i=1}^{N} \tag{1}$$

where i denotes the sample index and N = 4,680 denotes the total number of images. Since the medicine-name labels are initially available as text, they are converted into numerical indices prior to model training. The label space is given by:

$$y_i \in \{0, 1, 2, \ldots, K-1\} \tag{2}$$

where K = 78 denotes the total number of medicine-name classes. A consistent label-to-index mapping is constructed based on the medicine-name classes identified in the training partition. To ensure consistency and prevent data leakage, all labels in the validation and test partitions are verified strictly adhere to this training-based mapping.

3.3. Image preprocessing and augmentation

As the dataset consists of isolated, pre-cropped handwritten medicine-word images, preprocessing begins directly with the supplied word-level crops. Accordingly, prescription-page text detection, word segmentation, and cropping are excluded from the scope of this study. Each image is resized to 224×224 pixels for uniform processing by EfficientNet-B0, ViT-Base, and Swin-Base. Separate transformation pipelines are used for training and evaluation: the training pipeline includes resizing, random affine and perspective transformations, color jittering, tensor conversion, and normalization based on ImageNet statistics, whereas the validation and test pipelines include only resizing, tensor conversion, and normalization. These preprocessing operations described are applied after the medicine-word regions have been cropped in the released dataset. The normalization operation is given by:

$$x' = \frac{x - \mu}{\sigma} \tag{3}$$

where x denotes the input image tensor, x' denotes the normalized image tensor, and μ and σ denote the channel-wise mean and standard deviation of the ImageNet normalization statistics, respectively.

3.4. Custom dataset construction and data loading

A custom dataset class loads handwritten prescription word images and their labels from the annotation files. For each sample, the image filename is retrieved, the image is loaded in RGB format, the textual label is mapped to its numerical index, and the selected transformation pipeline is applied. The dataset then returns the transformed image tensor and its target label. These datasets are wrapped into PyTorch data loaders for mini-batch processing.

In this implemented configuration, the batch size is set to 8. Given that EfficientNet-B0, ViT-Base, and Swin-Base are fine-tuned simultaneously using 224×224 input images, the combined framework demands higher graphics processing unit (GPU) memory compared to individual models. A batch size of 8 enables reliable training of the complete framework within available NVIDIA L4 GPU memory constraints. The batch size is maintained across all baseline and ablation experiments to ensure consistent training conditions. As gradient accumulation is not used, the effective batch size remains 8. Although larger batch sizes are commonly employed for transformer training, the study fine-tuned pretrained ViT-Base and Swin-Base models using AdamW and a learning rate of 3×10^{-5}, rather than training the backbones from random initialization.

3.5. Mixup-based regularization

To improve generalization, mixup regularization is applied during training. Given two training samples (x_i, y_i) and (x_j, y_j), the mixed input is defined as:

$$\tilde{x} = \lambda x_i + (1-\lambda) x_j \tag{4}$$

where x_i and x_j denote two randomly selected training images. y_i and y_j denote their corresponding class labels, $\tilde{x}$ represents the mixed training image, and $\lambda \in [0,1]$ is the mixing coefficient.

Next, the mixing coefficient λ is sampled from a Beta distribution, as expressed by:

$$\lambda \sim \text{Beta}(\alpha, \alpha) \tag{5}$$

where Beta (α,α) denotes the symmetric Beta distribution used to sample λ ,and α controls the strength of the mixup interpolation.

In the implemented framework, α is set to 0.1, as shown below:

$$\alpha = 0.1 \tag{6}$$

The mixup loss is computed as a weighted combination of the losses associated with the two labels, as given by:

$$L_{\text{mixup}} = \lambda L(\hat{y}, y_i) + (1-\lambda) L(\hat{y}, y_j) \tag{7}$$

where L_{mixup} denotes the mixup loss, $L(.)$ represents the classification-loss function, $\hat{y}$ denotes the model-predicted class-probability vector, and y_i and y_j are the class labels of the two mixed training samples.

3.6. Proposed multi-backbone architecture

The proposed architecture captures complementary local, global, and hierarchical information from handwritten prescription images to support robust feature representation. EfficientNet-B0 extracts local texture and shape cues, ViT-Base models global contextual dependencies through patch-based self-attention, and Swin-Base captures hierarchical local-to-global patterns using shifted-window attention. For an input image, parallel feature extraction is defined as:

$$f_e = \phi_e(x), \quad f_v = \phi_v(x), \quad f_s = \phi_s(x) \tag{8}$$

where x denotes the input medicine-word image; ϕ_e, ϕ_v, and ϕ_s denote the feature-extraction functions of EfficientNet-B0, ViT-Base, and Swin-Base, respectively; and f_e , f_v , and f_s represent the corresponding extracted feature vectors.

Here f_e , f_v , and f_s denote the feature vectors extracted by EfficientNet-B0, ViT-Base, and Swin-Base, respectively. In the implemented framework, the original classification heads are removed, leaving each backbone to act as a feature extractor. The extracted feature dimensions are as follows:

$$f_e \in \mathbb{R}^{1280}, \quad f_v \in \mathbb{R}^{768}, \quad f_s \in \mathbb{R}^{1024} \tag{9}$$

where $\mathbb{R}^d$ denotes a d-dimensional real-valued feature space. Thus, EfficientNet-B0, ViT-Base, and Swin-Base produce feature vectors of 1,280, 768, and 1,024 dimensions, respectively.

3.7. Feature concatenation and fusion

The feature vectors from EfficientNet-B0, ViT-Base, and Swin-Base are concatenated to form a unified representation:

$$f = [f_e, f_v, f_s] \tag{10}$$

where f denotes the unified feature vector obtained by concatenating f_e , f_v , and f_s.

The concatenated feature vector lies in a 3,072-dimensional feature space, as defined by:

$$f \in \mathbb{R}^{3072} \tag{11}$$

Here, $\mathbb{R}^{3072}$ denotes a 3,072-dimensional real-valued feature space. Given that EfficientNet-B0, ViT-Base, and Swin-Base produce feature dimensions of 1,280, 768, and 1,024, respectively, the resulting dimensionality is:

$$1280 + 768 + 1024 = 3072 \tag{12}$$

The concatenated vector is then passed through a fully connected hidden layer, ReLU activation, and dropout. The projected hidden representation h is defined as follows:

$$h = \text{ReLU}(W_f f + b_f) \tag{13}$$

where f denotes the 3,072-dimensional concatenated feature vector obtained from Eq. (10), W_f and b_f represent the weight matrix and bias vector of the fully connected projection layer, respectively, h denotes the projected hidden representation, and ReLU(·) denotes the rectified linear unit activation function. To regularize the projected hidden representation and reduce overfitting during training, dropout is subsequently applied to h, as follows:

$$h' = \text{Dropout}(h) \tag{14}$$

where h' denotes the dropout-regularized hidden representation, h denotes the projected hidden representation obtained from Eq. (13), and Dropout(·) denotes the dropout operation. In the primary implemented configuration, a dropout rate of 0.3 is applied to the ReLU-activated fully connected hidden representation h before it is passed to the final classification layer. This dropout operation is used to reduce overfitting and improve model generalization during training.

3.8. Classification layer

The final classifier maps the dropout-regularized hidden representation to the target class space and produces the class logits, as defined by

$$o = W_c h' + b_c \tag{15}$$

where o denotes the vector of class logits, W_c represents the weight matrix of the classification layer, h' denotes the dropout-regularized hidden representation obtained from Eq. (14), and b_c represents the corresponding bias vector.

The predicted class label $\hat{y}$ is obtained by selecting the maximum logit score:

$$\hat{y} = \underset{c}{\text{argmax}}\, o_c \tag{16}$$

where $\hat{y}$ denotes the predicted medicine-name class, c denotes the class index, o_c represents the logit value for class c, and $\underset{c}{\text{argmax}}$ selects the class with the highest logit value.

Thus, fully connected processing and classification operations are summarized as:

$$\text{Linear}(3072 \rightarrow 1024) + \text{ReLU} + \text{Dropout}(0.3) + \text{Linear}(1024 \rightarrow K) \tag{17}$$

where K = 78 denotes the total number of medicine-name classes.

3.9. Model optimization and training strategy

A label-smoothed cross-entropy loss is used to reduce overconfidence and improve generalization. The objective is defined as:

$$L_{\text{CE}} = -\sum_{c=1}^{K} y_c^{\text{smooth}} \log P(y = c \mid x) \tag{18}$$

where L_{CE} denotes the label-smoothed cross-entropy loss, K denotes the total number of medicine-name classes, c denotes the class index, y_c^{smooth} represents the smoothed target value for class c, and $P(y = c \mid x)$ denotes the predicted probability that input image x belongs to class c.

When mixup is enabled, the final objective becomes a weighted combination of two label-smoothed cross-entropy losses. Optimization is performed using AdamW, whose update rule is expressed as:

$$\theta_{t+1} = \theta_t - \eta \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon} - \eta \, \lambda \, \theta_t \tag{19}$$

where θ_t denotes the model parameters at optimization step t , θ_{t+1} represents the updated model parameters, η is the learning rate, λ_{wd} is the weight-decay coefficient, $\hat{m}_t$ and $\hat{v}_t$ are the bias-corrected first- and second-moment estimates, respectively, and ϵ is a small constant used for numerical stability.

In the implemented configuration, the learning rate is set to 3×10^{-5}, the weight decay coefficient is set to 1×10^{-4}, label smoothing is 0.1, mixup alpha is 0.1, batch size is 8, and the maximum number of training epochs is 20. The learning rate is progressively increased over the first five epochs, reaching 25% of the maximum value specified by the schedule. Such a cautious warmup phase is required because all three pretrained backbone branches and the new fully connected and classification layers are simultaneously optimized. Progressively increasing the learning rate reduces the likelihood of sudden changes in model parameters during the initial phase of fine-tuning. The warmup phase is followed by the scheduled learning-rate decay. The five-epoch warmup phase is chosen as a conservative optimization technique, but it is not determined as the best value in the ablation study of the warmup phase. The checkpoint with the highest validation accuracy is retained, and early stopping is applied with a patience of five epochs.

3.10. Training procedure

The training procedure begins with dataset loading, label encoding, and image preprocessing. Each transformed image is passed independently through EfficientNet-B0, ViT-Base, and Swin-Base to obtain three feature representations. These feature vectors are directly concatenated to form a 3,072-dimensional representation. The concatenated representation is then passed through the fully connected layer, ReLU activation, dropout, and the final classification layer. Model parameters are optimized using AdamW with label smoothing, mixup, learning-rate warmup, and early stopping. After training, the checkpoint with the highest validation accuracy is reloaded and evaluated on the independent test set.

3.11. Algorithm of the proposed framework

The detailed training and evaluation procedure of the proposed multi-backbone feature-concatenation framework is presented in Algorithm 1.

Algorithm 1 Training and evaluation procedure of the proposed multi-backbone feature-concatenation framework

Input: training set D_{train}, validation set D_{val}, and test set D_{test}.

Output: best trained model and final test performance.

1. Load prescription word images and annotations from D_{train}, D_{val} and D_{test}.
2. Construct a unified label-to-index mapping.
3. Resize all images to 224 x 224 pixels and apply normalization.
4. Apply augmentation only to the training set.
5. Build datasets and mini-batch data loaders.
6. Initialize pretrained EfficientNet-B0, ViT-Base, and Swin-Base.
7. Remove the original classification heads.
8. For each epoch:
   a. Apply mixup to each training mini-batch.
   b. Extract features from the three backbones.
   c. Concatenate the three extracted feature vectors.
   d. Pass the concatenated representation through the fully connected and classification layers to obtain the class prediction.
   e. Compute the loss and update parameters using AdamW with warmup scheduling.
   f. Evaluate on Dval, save the best checkpoint, and apply early stopping if required.
9. Reload the best model.
10. Evaluate the model on D_{test} without augmentation.

3.12. Evaluation metrics

Classification performance is evaluated using accuracy, precision, recall, and F1-score. Accuracy is defined as:

$$\text{Accuracy} = \frac{\text{Correct Predictions}}{\text{Total Predictions}} \tag{20}$$

where correct predictions denote the number of test images assigned to their actual medicine-name classes, and total predictions denote the total number of evaluated test images.

For each medicine-name class, *TP*, *TN*, *FP*, and *FN* denote true positives, true negatives, false positives, and false negatives, respectively. Precision is defined as:

$$\text{Precision} = \frac{TP}{TP + FP} \tag{21}$$

Recall is defined as:

$$\text{Recall} = \frac{TP}{TP + FN} \tag{22}$$

The F1-score is defined as:

$$\text{F1} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{23}$$

In the implementation, macro-averaged and weighted precision, recall, and F1-score are reported to provide balanced and support-aware evaluation. The proposed framework is further assessed through overall performance comparison, ablation analysis, confusion-matrix-based evaluation, and class-wise error analysis.

3.13. Statistical analysis

All models are evaluated in five independent runs using random seeds 42, 123, 456, 789, and 999. Dataset partitions, preprocessing steps, and experimental settings remain unchanged across runs. Accuracy, macro F1-score, and weighted F1-score are calculated for each run and are reported as mean ± sample standard deviation. A two-sided paired t-test is used to compare the proposed model with each baseline model using the five seed-level test accuracy values. Since multiple baseline

comparisons are performed, the resulting p-values are adjusted using the Holm correction. A Holm-adjusted p-value below 0.05 is considered statistically significant. The use of five repeated runs provides a more stable estimate of run-to-run variability than the previous three-run setting and improves the reliability of the Holm-corrected statistical comparison.

3.14. Computational efficiency measurement

The computational requirements of the baseline and proposed models are evaluated under the same hardware and input conditions. The measurements are performed using an NVIDIA L4 GPU. Each model is evaluated in FP32 precision and evaluation mode using an input tensor of size [1, 3, 224, 224], corresponding to one RGB prescription-word image. Model complexity is examined using the total number of trainable parameters and the number of floating-point operations required for a forward pass. Inference time is measured in milliseconds per image, while peak GPU memory consumption is recorded in megabytes. The same measurement procedure is applied to all single-backbone, dual-backbone, and three-backbone configurations to ensure a consistent comparison.

To provide a preliminary practical mitigation analysis for the high computational cost of the proposed three-backbone framework, a branch-pruned version of the model is also evaluated. In this experiment, the ViT-Base branch is removed from the full EfficientNet-B0 + ViT-Base + Swin-Base framework, resulting in a lighter EfficientNet-B0 + Swin-Base configuration. This experiment is treated as structured branch pruning because an entire high-cost feature-extraction branch is removed from the original architecture. Finally, the full and branch-pruned models are compared using parameter count, inference time, peak GPU memory usage, and five-seed classification accuracy.

3.15. Ablation experiment design

Ablation experiments are conducted to examine the effects of training regularization, fully connected-layer design, and alternative backbone selection. The regularization analysis compares the complete training configuration with variants trained without mixup and without label smoothing. The fully connected layer analysis evaluates hidden dimensions of 512, 1,024, and 2,048; dropout rates of 0.1, 0.3, and 0.5; and direct classification, one-hidden-layer, and two-hidden-layer structures. The backbone-sensitivity analysis replaces individual components of the EfficientNet-B0, ViT-Base, and Swin-Base configurations with ResNet50, DeiT-Base, and ConvNeXt-Tiny.

Within each ablation group, the configurations are evaluated using the same training, validation, and test partitions, as well as the same preprocessing and optimization settings. Each configuration is independently evaluated using random seeds 42, 123, and 456. The results are reported as mean ± sample standard deviation. Because these ablation experiments are used as supporting sensitivity analyses rather than as the primary statistical comparison, they are evaluated over three independent runs, while the main baseline comparison and statistical significance analysis were expanded to five independent runs.

4. Results

4.1. Overall performance of the proposed model

The proposed multi-backbone feature-concatenation framework is evaluated on the independent test set for handwritten prescription word classification. To ensure robustness,the experiment is repeated independently using random seeds 42, 123, 456, 789, and 999 under identical training and evaluation conditions. Across the five runs, the proposed model achieves a mean accuracy of 88.44 ± 0.99% and a mean macro F1-score of 0.88 ± 0.01. The relatively small variation across the runs indicates that the model produces reasonably consistent results under different random initializations.

By combining EfficientNet-B0, ViT-Base, and Swin-Base, the framework incorporates local convolutional features, global visual dependencies, and hierarchical image representations. The concatenated representation, therefore, combines information learned through three distinct visual feature-extraction mechanisms. Fig. 3 presents the training and validation

loss curves for a representative experimental run. The curves show a gradual reduction in loss and stable convergence during training, without a persistent increase in validation loss. Because the curves correspond to a single representative run, they are included only to illustrate the model's optimization behavior. The final performance values are reported as the mean and standard deviation obtained across the five independent runs. The effects of mixup, label smoothing, and the fully connected layer configuration are examined separately in the ablation experiments.

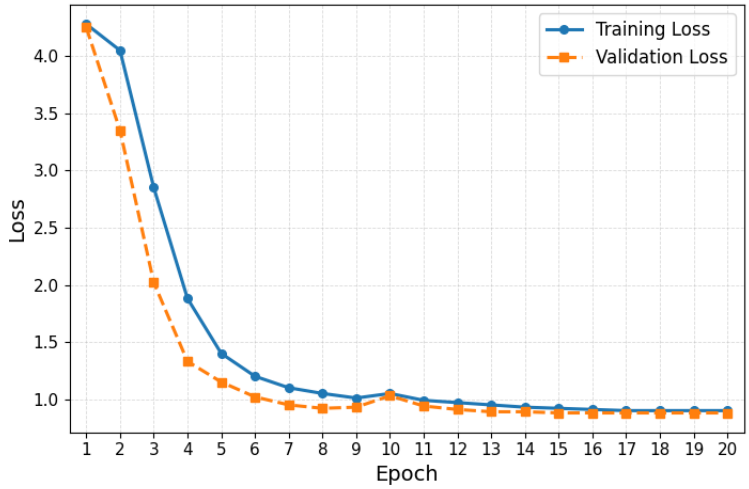


Fig. 3 Training and validation loss curves of the proposed model for a representative experimental run

The reported results demonstrate generalization only to the held-out test partition of the same dataset, as cross-validation and external-dataset evaluation are not performed, and the predefined training, validation, and test partitions supplied with the dataset are retained. Therefore, they do not establish performance across different writers, institutions, or clinical environments. Although the repeated-run results and the training–validation loss curves indicate stable optimization within the present dataset, they cannot replace cross-validation or external validation.

4.2. Comparative analysis with baseline models

To evaluate the proposed framework, comparative experiments are conducted using three single-backbone models and three dual-backbone configurations. Each experiment is independently repeated using random seeds 42, 123, 456, 789, and 999 under the same training, validation, and testing conditions. Table 2 reports the accuracy obtained in each run, along with the mean accuracy, macro F1-score, and weighted F1-score, expressed as mean ± sample standard deviation. The corresponding comparison of mean accuracy values is presented in Fig. 4.

Among the single-backbone models, Swin-Base achieves the highest average performance, with an accuracy of 87.18 ± 0.88% and a macro F1-score of 0.87 ± 0.01. ViT-Base obtains an average accuracy of 85.87 ± 1.21, whereas EfficientNet-B0 produced the lowest standalone result, with an average accuracy of 75.49 ± 1.39%.

Among the dual-backbone configurations, EfficientNet-B0 + Swin-Base achieves the highest mean accuracy of 88.33 ± 1.66% and a macro F1-score of 0.88 ± 0.02. ViT-Base + Swin-Base obtains an average accuracy of 87.00 ± 1.11%, while EfficientNet-B0 + ViT-Base achieved 86.08 ± 1.49%.

The proposed EfficientNet-B0 + ViT-Base + Swin-Base framework achieves the highest mean accuracy of 88.44 ± 0.99% and the highest macro F1-score of 0.88 ± 0.01 among the evaluated models. It outperforms Swin-Base by 1.26 percentage points and EfficientNet-B0 + Swin-Base by 0.10 percentage points in mean accuracy.

While the proposed framework attains the highest average performance, the observed improvements over the strongest single-backbone and dual-backbone models are relatively small. Therefore, the results are interpreted alongside the paired statistical significance tests reported in the following subsection, rather than relying solely on the numerical differences in

average accuracy. The results indicate that the parallel combination of convolutional, global-attention, and hierarchical visual representations yields complementary information; however, the magnitude and statistical reliability of the improvement depend on the chosen baseline.

Table 2 Comparative performance of the baseline and proposed models over five independent runs

Method	Seed 42 accuracy (%)	Seed 123 accuracy (%)	Seed 456 accuracy (%)	Seed 789 accuracy (%)	Seed 999 accuracy (%)	Mean accuracy (%)	Macro F1	Weighted F1	Runs
EfficientNet-B0	75.38	74.62	73.85	77.44	76.15	75.49 ± 1.39	0.75 ± 0.01	0.75 ± 0.01	5
ViT-Base	84.23	85.77	87.18	86.92	85.26	85.87 ± 1.21	0.85 ± 0.01	0.85 ± 0.01	5
Swin-Base	87.05	88.59	86.15	87.18	86.92	87.18 ± 0.88	0.87 ± 0.01	0.87 ± 0.01	5
EfficientNet-B0 + ViT-Base	86.66	84.74	87.95	84.36	86.67	86.08 ± 1.49	0.86 ± 0.02	0.86 ± 0.02	5
EfficientNet-B0 + Swin-Base	88.97	88.21	85.51	89.36	89.62	88.33 ± 1.66	0.88 ± 0.02	0.88 ± 0.02	5
ViT-Base + Swin-Base	88.72	87.31	86.03	86.92	86.03	87.00 ± 1.11	0.87 ± 0.01	0.87 ± 0.01	5
Proposed model	87.18	88.46	88.97	89.74	87.82	88.44 ± 0.99	0.88 ± 0.01	0.88 ± 0.01	5

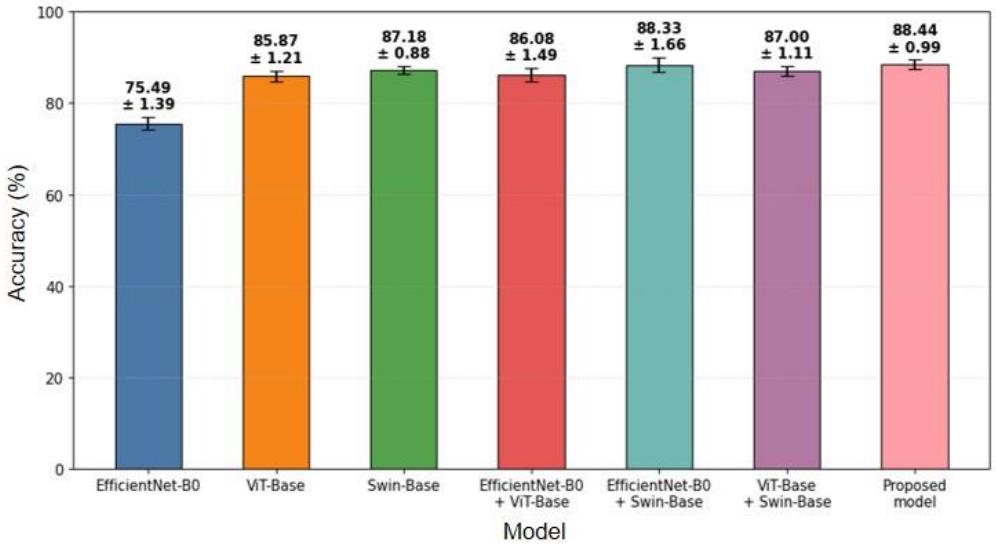


Fig. 4 Performance comparative of baseline and proposed models over five runs. (error bars represent standard deviation)

4.3. Statistical significance analysis

To examine whether the differences between the proposed framework and the baseline models are statistically significant, two-sided paired t-tests are performed using the accuracy values obtained from five independent runs. The results are paired according to the common random seeds 42, 123, 456, 789, and 999. Because the proposed model is compared with six baseline configurations, the resulting p-values are adjusted using the Holm correction to control the family-wise error rate. Statistical significance is assessed at the 0.05 level.

As reported in Table 3, the proposed model achieves a statistically significant improvement over EfficientNet-B0, with a mean accuracy difference of 12.95 percentage points and a Holm-adjusted p-value of 0.001. The comparison with ViT-Base also shows statistical significance, with a mean accuracy difference of 2.56 percentage points and a Holm-adjusted p-value of 0.001. These results indicate that the proposed multi-backbone framework provides statistically supported improvements over the EfficientNet-B0 and ViT-Base single-backbone baselines.

However, the differences between the proposed method and Swin-Base and the three dual-backbone architectures are not statistically significant following the Holm correction. In detail, the proposed architecture outperforms Swin-Base by 1.26 percentage points and EfficientNet-B0 + Swin-Base by only 0.10 percentage points; yet, the corresponding Holm-adjusted p-values are 0.327 and 0.920, respectively. Similarly, the improvements over EfficientNet-B0 + ViT-Base and ViT-Base + Swin-Base do not reach statistically significant after correction. These results suggest that the proposed framework attains the highest mean accuracy among the evaluated models, its improvement over the strongest Swin-based and dual-backbone baselines remains mainly numerical rather than statistically significant.

Table 3 Paired t-test results comparing the proposed model with the baseline models over five independent runs

Comparison	Mean difference (percentage points)	t-statistic	Raw p-value	Holm-adjusted p-value	Significant at 0.05
Proposed model vs. EfficientNet-B0	12.95	19.37	<0.001	<0.001	Yes
Proposed model vs. ViT-Base	2.56	12.65	<0.001	0.001	Yes
Proposed model vs. Swin-Base	1.26	2.06	0.109	0.327	No
Proposed model vs. EfficientNet-B0 + ViT-Base	2.36	2.51	0.066	0.264	No
Proposed model vs. EfficientNet-B0 + Swin-Base	0.10	0.11	0.920	0.920	No
Proposed model vs. ViT-Base + Swin-Base	1.44	1.76	0.153	0.327	No

4.4. Computational efficiency analysis

Computational efficiency of the single-backbone, dual-backbone, and three-backbone models is measured on the same parameters. The results are presented in Table 4, where the number of parameters, floating-point operations per second (FLOPs), inference time per image, and maximum GPU memory usage are specified for all architectures. These numbers are calculated on an NVIDIA L4 GPU using input dimensions of 224 × 224 with FP32 precision in the evaluation regime.

Among the single-backbone models, EfficientNet-B0 exhibits the lowest computational requirements, with 4.11 million parameters and 0.77 giga floating-point operations per second (GFLOPs). ViT-Base and Swin-Base contain 86.45 million and 86.82 million parameters, respectively. Swin-Base requires a longer inference time of 30.325 ms per image compared with ViT-Base, which records 7.033 ms per image under the evaluated hardware conditions.

The computational requirements increase when multiple backbones are executed in parallel. The proposed EfficientNet-B0 + ViT-Base + Swin-Base configuration contains 180.37 million parameters and requires 64.81 GFLOPs. Its inference time is 46.996 ms per image, with a peak GPU memory consumption of 718.89 MB. These represent the highest values among the evaluated configurations because the proposed framework extracts features using three independent backbones.

Although the proposed model records the highest mean classification performance, the efficiency results show that this improvement comes with increased computational cost. Therefore, the proposed framework is better suited to GPU-supported or server-based prescription-processing environments than to direct deployment on resource-constrained devices without further optimization. Techniques such as model compression, knowledge distillation, quantization, and lightweight backbone selection are potential avenues to reduce this requirement.

Table 4 Computational efficiency of the baseline and proposed models

Model	Parameters (M)	FLOPs (G)	Inference time (ms/image)	Peak GPU memory (MB)
EfficientNet-B0	4.11	0.77	9.101	38.59
ViT-Base	86.45	33.73	7.033	338.65
Swin-Base	86.82	30.31	30.325	359.07
EfficientNet-B0 + ViT-Base	92.57	34.50	16.504	368.88
EfficientNet-B0 + Swin-Base	93.19	31.08	39.893	384.63
ViT-Base + Swin-Base	175.05	64.04	37.333	697.33
Proposed model	180.37	64.81	46.996	718.89

To provide a preliminary practical mitigation experiment, a branch-pruned version of the proposed framework is evaluated, as shown in Table 4a. In this experiment, the ViT-Base branch is removed from the full EfficientNet-B0 + ViT-Base + Swin-Base framework, resulting in a lighter EfficientNet-B0 + Swin-Base configuration. This experiment serves as a form of structured branch pruning because an entire high-cost feature-extraction branch is removed.

Table 4a Preliminary branch-pruning efficiency analysis

Model	Architecture	Five-seed accuracy (%)	Parameters (M)	FLOPs (G)	Inference time (ms/image)	Peak GPU memory (MB)
Full proposed model	EfficientNet-B0 + ViT-Base + Swin-Base	88.44 ± 0.99	180.37	64.81	46.996	718.89
Branch-pruned model	EfficientNet-B0 + Swin-Base	88.33± 1.66	93.19	31.08	39.893	384.63

Compared with the full proposed model, the branch-pruned EfficientNet-B0 + Swin-Base configuration reduces the parameter count by 48.33%, FLOPs by 52.04%, inference time by 15.11%, and peak GPU memory by 46.50%. Importantly, the mean accuracy decreased only from 88.44 ± 0.99% to 88.33 ± 1.66%, corresponding to a difference of 0.10 percentage points. This result suggests that removing the ViT-Base branch substantially reduce computational cost while preserving nearly the equivalent classification performance under. Therefore, branch-pruned or lightweight two-backbone variants provide a more practical accuracy–efficiency trade-off for deployment-oriented studies.

4.5. Ablation analysis

The ablation analysis examines the effects of regularization, hidden dimension, dropout rate, fully connected-layer depth, and backbone selection. Within each ablation group, the configurations are evaluated using the same dataset partitions and experimental settings. Each configuration is evaluated across three independent runs, and the results are reported as mean ± sample standard deviation. Because these ablation experiments are used as supporting sensitivity analyses rather than as the primary statistical comparison, they are evaluated over three independent runs, while the main baseline comparison and statistical significance analysis are expanded to five independent runs.

4.5.1. Regularization ablation

Table 5 compares the complete training configuration with variants trained without mixup and without label smoothing. Removing label smoothing reduces the mean accuracy from 88.21% to 87.82%.The macro F1-score also decreased slightly, although both values round to 0.88 at two decimal places. This result indicates that label smoothing provides a modest benefit under the evaluated conditions.

In contrast, the configuration trained without mixup achieves a slightly higher mean accuracy of 88.42% and a macro F1-score of 0.88. However, it also showed greater variation across the three runs. Therefore, mix-up does not consistently improve mean performance under the evaluated conditions.

Table 5 Regularization ablation results over three independent runs

Configuration	Accuracy (%)	Macro-F1	Runs
Complete configuration	88.21 ± 0.92	0.88 ± 0.01	3
Without mixup	88.42 ± 1.31	0.88 ± 0.02	3
Without label smoothing	87.82 ± 1.05	0.88 ± 0.01	3

4.5.2. Hidden dimension and dropout analysis

Table 6 presents the effect of changing the hidden dimension and dropout rate of the fully connected layer. Reducing the hidden dimension from 1,024 to 512 decreases the mean accuracy to 87.22% and the macro F1-score to 0.87. Conversely, increasing the hidden dimension to 2,048 improves the mean accuracy to 88.76% and the macro F1-score to 0.89.

Among the tested dropout settings, the 1,024-dimensional configuration with a dropout rate of 0.1 attains the highest mean accuracy of 88.89% and a macro F1-score of 0.89. Increasing the dropout rate to 0.5 reduces both measures. These findings show that the fully connected layer configuration influences the effectiveness of the classification process for the concatenated feature representation.

Table 6 Effect of hidden dimension and dropout rate

Fully connected-Layer configuration	Accuracy (%)	Macro-F1	Runs
512 dimensions, dropout 0.3	87.22 ± 0.45	0.87 ± 0.00	3
1,024 dimensions, dropout 0.3	88.21 ± 0.92	0.88 ± 0.01	3
2,048 dimensions, dropout 0.3	88.76 ± 0.07	0.89 ± 0.00	3
1,024 dimensions, dropout 0.1	88.89 ± 0.75	0.89 ± 0.01	3
1,024 dimensions, dropout 0.5	87.99 ± 0.82	0.88 ± 0.01	3

4.5.3. Fully connected-layer depth analysis

The variants of the fully connected layer depth are independently retrained as part of a separate controlled ablation experiment. Therefore, the results in Table 7 are used to compare the depth configurations within this experiment and do not replace the primary proposed model result reported in Table 2.

The direct-classification configuration contains one fully connected layer that maps the 3,072-dimensional concatenated vector directly to the 78 output classes. The one-hidden-layer configuration contains a 1,024-dimensional hidden layer followed by the classification layer. The two-hidden-layer configuration incorporates 2,048 and 1,024-dimensional hidden layers preceding the classification layer.

In this depth-based experiment, the one-hidden-layer configuration achieves the highest performance, with an accuracy of 89.57% and a macro F1-score of 0.89. The direct-classification configuration produces lower average performance, whereas adding a second hidden layer does not provide further improvement. These findings indicate that, among the evaluated depth configurations, a single hidden layer provides the most effective processing of the concatenated feature vector.

Table 7 Effect of fully connected-layer depth

Configuration	Structure	Accuracy (%)	Macro F1	Runs
Direct classifier	3,072 → 78	89.02 ± 1.22	0.89 ± 0.01	3
One hidden layer + classifier	3,072 → 1024 → 78	89.57 ± 0.63	0.89 ± 0.01	3
Two hidden layers + classifier	3,072 → 2,048 → 1,024 → 78	88.97 ± 0.22	0.89 ± 0.00	3

4.5.4. Alternative-backbone sensitivity analysis

Table 8 shows the results obtained by substituting each backbone from the initial setting. The accuracy drops to 86.45% and F1-score to 0.86 when EfficientNet-B0 is substituted by ResNet50. This indicates that the ResNet50 branch does not yield performance improvements under the current training settings.

Replacing ViT-Base with DeiT-Base produces a mean accuracy of 89.02%, although the variation across runs is comparatively high. Configurations incorporating ConvNeXt-Tiny-based produce the strongest numerical results in the sensitivity analysis. Specifically, ConvNext-Tiny + ViT-Base + Swin-Base achieves a mean accuracy of 89.83% and a macro F1-score of 0.90, while EfficientNet-B0 + ViT-Base + ConvNeXt-Tiny attains 89.70% and 0.90, respectively.

The original EfficientNet-B0, ViT-Base, and Swin-Base combination is selected prior to the sensitivity experiments to represent three complementary visual learning mechanisms. EfficientNet-B0 provides convolutional local texture and stroke information, ViT-Base models global patch-level relationships, and Swin-Base captures hierarchical spatial patterns through shifted-window attention. The original configuration achieves the highest performance in the primary comparison against predefined single- and dual-backbone baselines. Nevertheless, the sensitivity analysis identifies ConvNeXt-Tiny-based alternatives with higher numerical performance. Therefore, the original combination is viewed as an a priori heterogeneous design rather than as the empirically optimal combination among all tested alternatives.

Although the ConvNeXt-Tiny-based configurations achieve higher numerical performance, the primary proposed framework remains unchanged. This is because the EfficientNet-B0 + ViT-Base + Swin-Base configuration is defined before the sensitivity analysis and before the final interpretation of the test results. Revising the main proposed model after observing the sensitivity-analysis outcomes could introduce post hoc model selection bias. Therefore, the original configuration is retained as the predefined reference for the primary comparative and statistical analyses. The ConvNeXt-Tiny-based results are instead interpreted as evidence that the multi-backbone feature-concatenation framework is extensible, and these combinations warrant further investigation as candidate architectures in future controlled studies.

Table 8 Alternative-backbone sensitivity analysis

Architecture	Accuracy (%)	Macro-F1	Runs
EfficientNet-B0 + ViT-Base + Swin-Base	88.21 ± 0.92	0.88 ± 0.01	3
ResNet50 + ViT-Base + Swin-Base	86.45 ± 0.27	0.86 ± 0.00	3
ConvNext-Tiny + ViT-Base + Swin-Base	89.83 ± 0.45	0.90 ± 0.01	3
EfficientNet-B0 + DeiT-Base + Swin-Base	89.02 ± 1.55	0.89 ± 0.02	3
EfficientNet-B0 + ViT-Base + ConvNext-Tiny	89.70 ± 1.23	0.90 ± 0.01	3

4.6. Confusion matrix and class-wise performance analysis

To examine the classification behavior in greater detail, a confusion matrix and class-wise analyses are performed on the independent test set. The results presented in this subsection correspond to the representative test run used to generate the class-wise F1-score distribution.

As shown in Fig. 5, most values are concentrated along the main diagonal, indicating that the majority of test samples were correctly classified. Nevertheless, several off-diagonal errors persisted among medicine names with similar handwritten structures. The most frequent confusion occurred between Montex and Montair, with four Montex samples predicted as Montair. Other notable confusion pairs included Esoral predicted as Flugal, Ketozol predicted as Ketoral, Montene predicted as Montair, and Sergel predicted as Maxpro, each exhibiting three errors. Furthermore, pairs such as Flugal predicted as Progut,

Rivotril predicted as Provair, Esonix predicted as Exium, Fenadin predicted as Fexo, and Dinafex predicted as Disopan each accounted for two misclassifications. These patterns suggest that similar word lengths, connected strokes, compressed handwriting, and indistinct character boundaries contributed to the observed classification errors.

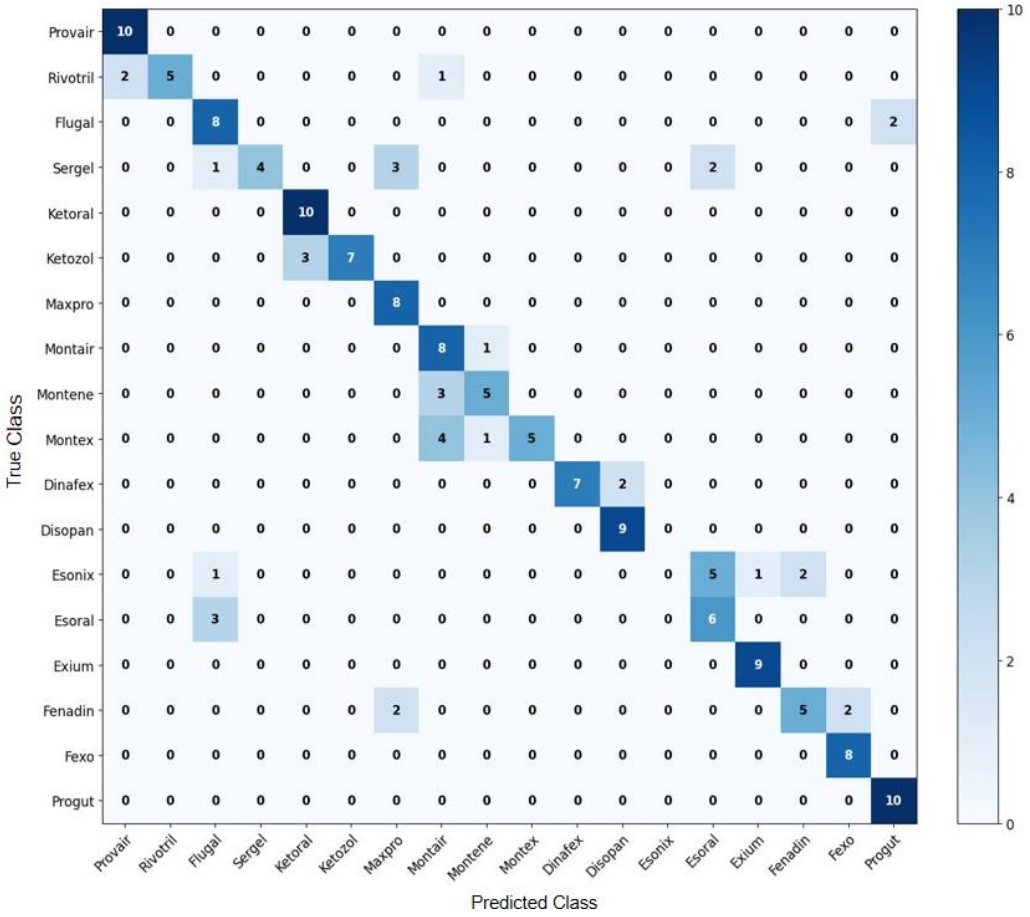


Fig. 5 Confusion matrix for the medicine-name classes involved in the most frequent misclassifications

Fig. 6 shows substantial variation in performance across the 78 medicine-name classes. Several classes achieve perfect or near-perfect F1-scores, whereas a smaller group remains comparatively difficult. To examine this variation more closely, Table 9 reports precision, recall, F1-score, and test support for the five highest-performing and five lowest-performing classes from the same representative test run.

Classes such as Dexilofen, Baclon, Bacmax, Azyth, and Metro achieved precision, recall, and F1-score values of 1.00, indicating that all ten test samples from each of these classes were correctly classified. In contrast, Trilock achieved the lowest F1-score of 0.33. Its recall of 0.20 indicates that only two of its ten test samples were correctly identified. Renova and Ritch also obtained recall values of 0.40, showing that several true samples from these classes were assigned to other medicine-name labels.

Montair demonstrated a distinct error pattern. Its recall was relatively high at 0.70; however, its precision was comparatively low at 0.33. This result suggests that while the model correctly identified many of the true Montair examples, it also frequently misclassified samples from other classes as Montair.

The lower-performing classes were not rare categories. The dataset was balanced, with 40 training images, 10 validation images, and 10 test images for every medicine-name class. Therefore, the observed variation in F1-score cannot be attributed to class frequency. Instead, the reduced performance was likely associated with visual similarity, cursive joining, compressed character formation, overlapping strokes, and unclear boundaries between handwritten letters.

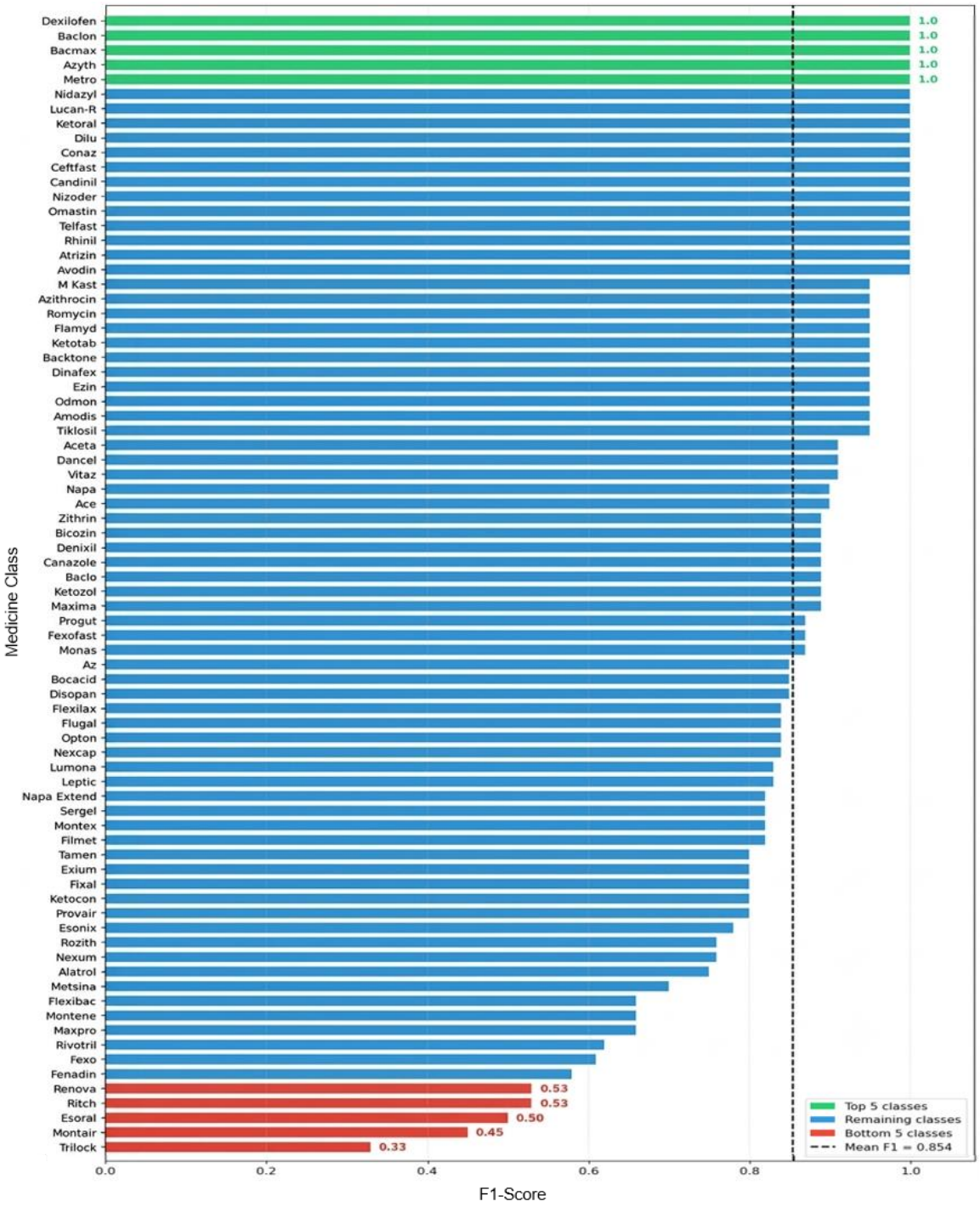


Fig. 6 Per-class F1-score distribution across the 78 medicine-name classes
(The dashed line represents the mean class-wise F1-score of 0.854)

Table 9 Per-class performance of the top five and bottom five medicine-name classes

Performance group	Medicine-name class	Precision	Recall	F1-score	Test support
Top 5	Dexilofen	1.00	1.00	1.00	10
Top 5	Baclon	1.00	1.00	1.00	10
Top 5	Bacmax	1.00	1.00	1.00	10
Top 5	Azyth	1.00	1.00	1.00	10
Top 5	Metro	1.00	1.00	1.00	10
Bottom 5	Renova	0.80	0.40	0.53	10
Bottom 5	Ritch	0.80	0.40	0.53	10
Bottom 5	Esoral	0.43	0.60	0.50	10
Bottom 5	Montair	0.33	0.70	0.45	10
Bottom 5	Trilock	1.00	0.20	0.33	10

Fig. 7 shows that the misclassifications were not randomly distributed but repeatedly occurred between specific medicine-name pairs. The most frequent mapping was Montex → Montair, occurring in four of the ten Montex test samples. Montene → Montair occurred three times, indicating that samples from more than one visually similar class were misclassified as Montair. Other repeated mappings included Esoral → Flugal, Ketozol → Ketoral, and Sergel → Maxpro, each occurring three times. Additionally, pairs such as Flugal → Progut, Rivotril → Provair, Esonix → Exium, Fenadin → Fexo, and Dinafex → Disopan each occurred twice.

Common failure modes identified through these recurring confusion pairs include the general word length and form, joined cursive strokes, compression of the middle characters, indistinct letter boundaries, and similarities in the initial or final characters. For instance, Montex, Montene, and Montair share highly similar structural patterns, which complicates their discrimination under conditions of handwriting compression or stroke connectivity. Similarly, Ketozol and Ketoral exhibit high visual similarity due to shared prefixes and comparable word lengths.

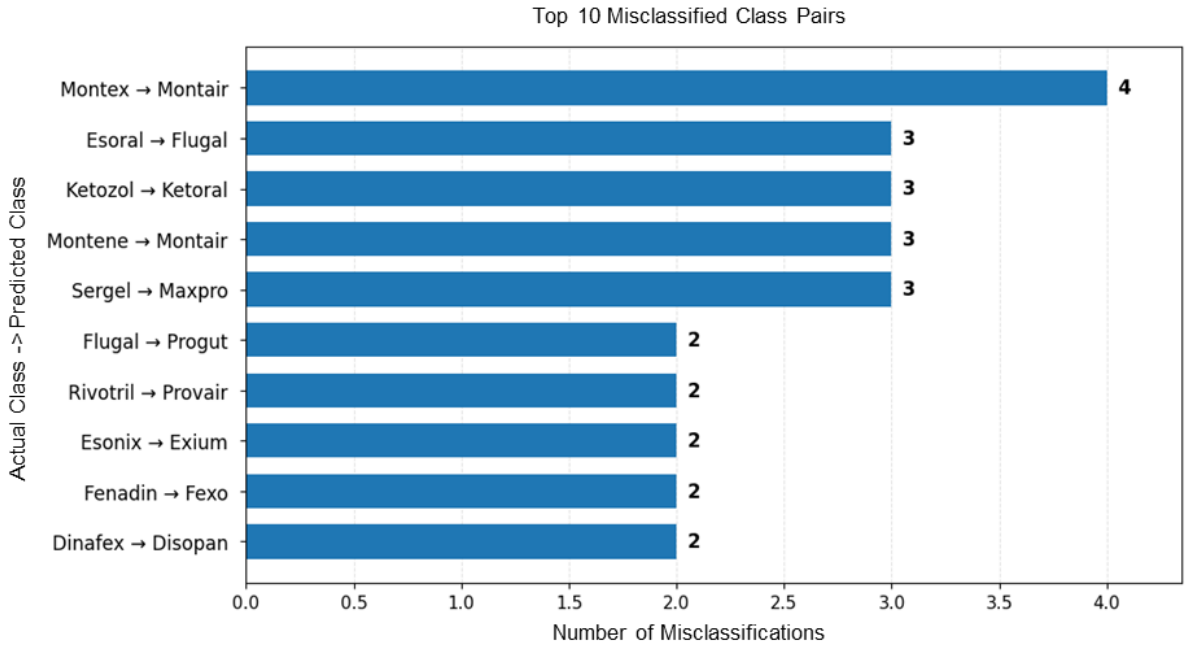


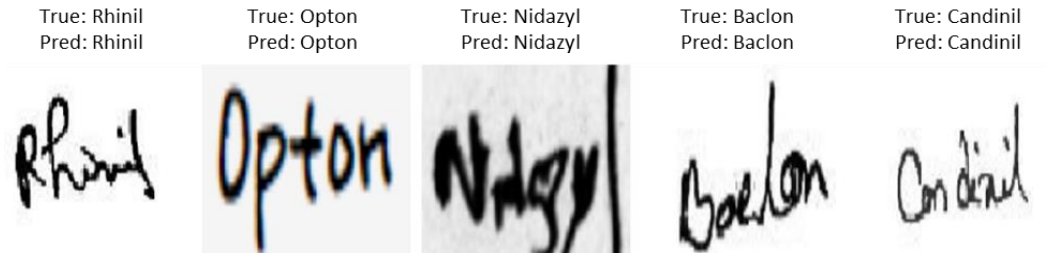
Fig. 7 Ten most frequent misclassified medicine-name class pairs in the representative test run

4.7. Qualitative classification analysis

To illustrate the classification behavior of the proposed model, representative images of correctly and incorrectly classified prescription words are selected from the test set. Fig. 8 displays five samples of correctly classified words are shown at the top and five samples of misclassified instances at the bottom.

The correctly identified instances include Rhinil, Opton, Nidazol, Baclon, and Candinil. The instances demonstrate that the model identifies words by leveraging features such as stroke width, character spacing, slant, and cursive writing patterns. The similarity between the actual and predicted labels indicates that the model learns robust visual representations from the medicines names.

Misclassified instances provide further insights into the challenges associated with prescription recognition. The model's predictions include: Nexum for Opton, Flugal for Lumona, Zithrin for Flexibac, Esoral for Sergel, and Progut for Flugal. These instances exhibit challenging features such as compressed characters, connected strokes, irregular spacing, or poor letter-boundary formation. This means that drug names from other classes share visual similarities that complicate accurate classification. Consequently, these qualitative observations support the confusion matrix, class-wise, and failure pattern analysis results.



(a) Correctly classified samples

(b) Misclassified samples

Fig. 8 Representative classification results on the independent test set

5. Discussion

These results show that the multi-backbone feature-concatenation approach achieves the highest mean performance among the predefined single- and dual-backbone baselines. This suggests that local texture features, global context-dependent features, and hierarchical features provide complementary information for classifying handwritten prescription words. The framework's performance is largely attributed to the effective integration of these diverse features via direct concatenation.

The comparative analysis establishes that no individual backbone performs as well as three-branch model. Even though Swin-Base produces the best results when used in isolation, its performance is not statistically better than the proposed model. In addition, all dual backbone configurations remain inferior to the complete model in terms of average performance. The paired t-test with Holm correction reveals statistically significant improvements over EfficientNet-B0 and ViT-Base based on five independent runs. However, no statistically significant improvement is observed compared with Swin-Base or the three dual-backbone configurations. Hence, the proposed model achieves the best mean performance among the primary baselines across five repeated runs, but its improvement over the strongest Swin-Base and dual-backbone baselines remains mainly numerical rather than statistically significant.

The observed differences among dual-backbone models indicate that combining more than two backbones does not necessarily enhance classification performance. Key considerations for developing multi-backbone models include feature compatibility, the design of the fully connected layers, and the optimization process. The regularization ablation shows somewhat contradictory findings. Elimination of label smoothing leads to a slight decrease in mean accuracy and macro F1-score. In contrast, the model without mixup exhibits slightly better mean accuracy but greater variability across three trials.

The additional ablation study provides further insight into the choice of architecture. Performance depends on the hidden dimension, dropout value, and depth of the fully connected layers. Among the hidden-dimension and dropout configurations evaluated in Table 6, the model with a hidden dimension of 1,024 and a dropout rate of 0.1 achieved the highest mean accuracy of 88.89% and a macro F1-score of 0.89. In the depth study, the model with a single hidden layer yields the optimal mean accuracy of 89.57% and macro F1-score of 0.89. Conversely, direct classification performs suboptimally, and the addition of a second hidden layer fails to provide further improvements. These results show that moderate fully connected processing outperformes both direct classification and deep architectures.

The analysis reveals that classification performance hinges on the chosen feature extractors. Using ResNet50 instead of EfficientNet-B0 leads to diminished mean accuracy and macro F1-score; however, ConvNeXt-Tiny-based architectures generate superior results. For instance, the combination of ConvNeXt-Tiny + ViT-Base + Swin-Base had a mean accuracy of 89.83% and macro F1-score of 0.90. The combination of EfficientNet-B0 + ViT-Base + Swin-Base is selected to incorporate local, global, and hierarchical feature representations. Therefore, it serves as a predetermined heterogeneous reference design rather than the empirically optimal backbone combination. The main model maintains its original structure following the ConvNeXt-Tiny sensitivity results, as subsequent revisions threaten to introduce post-hoc model selection bias. Instead, these findings underscore the extensibility of the proposed feature-concatenation framework, and ConvNeXt-Tiny-based configuration emerge as compelling candidate architectures for future controlled studies.

The computational efficiency evaluation unveils a trade-off between performance and resource requirements. The three-backbone architecture comprises 180.37 M parameters and demands 64.81 GFLOPs of operations for a input image size 224 x 224. Under evaluation conditions with an NVIDIA L4 GPU and FP32, the model processes images at 46.996 ms and reaches a peak GPU memory usage of 718.89 MB. These metrics surpass those of the single- and two-backbone architectures, as three feature extractors operate in parallel. Hence, while the proposed architecture attains superior classification performance, its computational requirements impede practical deployment. This model is better suited to GPU-supported or server-based environments rather than constrained edge devices. Possible research directions to reduce computational overhead include pruning, quantization, knowledge distillation, and the integration of lightweight backbones.

A preliminary branch-pruning experiment investigates a practical mitigation strategy for the high computational cost. In this experiment, the ViT-Base branch is removed, resulting in an EfficientNet-B0 + Swin-Base configuration. This structured branch-pruned model slashes the parameters count from 180.37 M to 93.19 M, corresponding to a 48.33% reduction. It decreases FLOPs from 64.81 G to 31.08 G, inference time from 46.996 ms/image to 39.893 ms/image, and peak GPU memory from 718.89 MB to 384.63 MB. The five-seed accuracy shows minimal variation, shifting from 88.44 ± 0.99% for the full model to 88.33 ± 1.66% for the pruned version. This indicates that structured branch pruning offers a viable initial optimization direction by curbing computational demands while maintaining stable classification performance. Nevertheless, complementary technique—such as knowledge distillation, quantization-aware training, post-training quantization, and lightweight backbone—remain necessary to facilitate before real-time or resource-constrained clinical deployment.

The batch size and warm-up scheme align with the computing and optimization needs of the proposed framework. A batch size of 8 is adopted, as the simultaneous fine-tuning of EfficientNet-B0, ViT-Base, and Swin-Base imposes a significantly higher memory demand than training individual backbones. Since gradient accumulation is omitted, the effective batch size is 8. Data augmentation is applied only to the training partition, as described in the preprocessing section. Even though transformer models typically thrive with larger batches, both the ViT-Base and Swin-Base branches are fine-tuned with the AdamW optimizer at a relatively low learning rate of 3×10^{-5}. A linear warm-up spans the first five epochs— accounting for 25% of the maximal 20 epochs—to facilitate a stable transition into the target learning rate while simultaneously training the integrated fully connected and classification layers. While these settings ensure a stable training process, they remain conservative and lack exhaustive empirical optimization regarding the batch size and warm-up duration.

Cross-validation and external testing datasets are omitted, as this study maintains the predefined train, validation, and test splits of the dataset. Therefore, the results presented reflect only the model performance on the held-out test split of the same dataset, not on prescription images from other writers, institutions, scanners, image-capture conditions, prescription formats, or clinical environments. Techniques such as pretraining, data augmentation, label smoothing, mixup, dropout, early stopping, validation-based checkpointing, and multi-run testing help mitigate and measure the extent of overfitting. However, these procedures do not replace writer-independent cross-validation or external-dataset validation. The absence of external validation

constitutes a significant limitation, given that the appearance of handwritten prescriptions varies substantially across regions, clinicians, writing styles, imaging devices, and prescription templates. Future work should therefore evaluate the framework on independent prescription-word datasets and full-page prescription images collected from different sources. Even a small externally labeled validation sample would provide useful preliminary evidence of out-of-dataset reliability, but such a dataset is not available in the present study.

The class-wise analysis yields large differences among medicine name categories. Categories such as Ace, Aceta, Amodis, Atrizin, and Backtone achieve perfect precision, recall, and F1-score during the representative test run, while Flexibac, Renova, Sergel, Montair, and Montene had the lowest F1-scores. The suboptimal performance of these categories does not stem from relative rarity, as each category maintains an identical distribution across training, validation, and test. Instead, these results derive primarily from visual resemblance and ambiguous handwriting. Low recall values for Flexibac, Renova, Sergel, and Montene indicate that the model frequently misclassifies true instances into other categories. The high recall and low precision of Montair are attributable to the misclassification of visually similar instances—particularly Montex and Montene—as Montair.

The failure pattern analysis reveals that errors arise consistently from recurring class-to-class mappings rather than random errors. The most common recurring mapping occurs between Montex and Montair appearing in 40% of the Montex test run. Similarly, Montene and Montair exhibit recurring confusion, confirming that samples from visually similar classes are confused with Montair. Other notable pairs include Esoral–Flugal, Ketozol–Ketoral, and Sergel–Maxpro, each mapped three times. Additional pairs, such as Flugal–Progut, Rivotril–Provair, Esonix–Exium, Fenadin–Fexo, and Dinafex–Disopan, which recur twice. These mappings originate primarily from morphological similarities, such as shared word lengths, connected characters, compression of middle letters, unclear borders between some letters, and common first or last letters. For instance, Montex, Montene, and Montair share a structure that proves difficult to distinguishable in compressed, cursive handwriting ,while Ketozol and Ketoral possess similar prefixes and similar length.

To address these visually confusable classes in future work, a confusion-aware training strategy serves as the primary focus.. First, frequently confused class pairs—such as Montex–Montair, Montene–Montair, Ketozol–Ketoral, and Esoral–Flugal—are identified from the validation confusion matrix. These pairs can then facilitate the construct of hard-negative mini-batches, where visually similar but different medicine-name classes are sampled together during training. Second, supervised contrastive learning enhances the classification objective, pulling embeddings of identical medicine-name classes closer while pushing embeddings of visually similar but distinct classes farther apart. Third, class-aware augmentation targets difficult classes by introducing controlled stroke compression, slant variation, blur, spacing changes, and contrast variation. Finally, during deployment, the system top-k predictions with calibrated confidence scores to the pharmacist, enabling the manual verification of visually confusable cases rather than their automatic acceptance.

Qualitative prediction analysis provides deeper insight into the model's performance. The correctly predicted instances confirm that the proposed model accurately classifies prescription words regardless of variability in stroke weight, slanting, letter spacing, and letter formation. On the other hand, incorrect predictions are made for cases involving cursive and tightly spaced letters, as well as in medicine names that share similar structural shapes. This finding is consistent with the confusion matrix results and those from individual classes, suggesting that such errors stem from visual confusability.

Clinical implementation of the proposed approach necessitates a human-in-the-loop workflow rather than an automated medication recognition process. All medication predictions require verification by a professional pharmacist or clinician before dispensing or recording them in an electronic medication record. Such a requirement is especially necessary when dealing with low-confidence predictions and also those medicines whose handwriting resembles other medications. The actual implementation of the suggested approach might involve showing the user the highest-ranked label, along with some

alternative labels and their confidence scores. Predictions that fail to meet a predefined confidence threshold should be automatic exclusion and mandate manual review. Un-calibrated softmax probabilities are not considered clinically reliable confidence measures.

Define a single acceptable error rate for all medication safety scenarios is not feasible, since the acceptable risk varies according to the drug type, dosage implications, clinical context, and the consequences of an incorrect prediction. It is necessary to develop appropriate operational criteria—in consultation with pharmacists and doctors—based on sensitivity, false acceptance rate, confidence calibration, and manual review load. In cases involving safety-critical medications, more stringent criteria and professional confirmation are required.

This current research presents a limitation regarding the data input format. Specifically, the framework utilizes pre-cropped images of medicine words rather than entire prescription pages. Consequently, the presented results reflect word classification performance under the assumption that the medicine name is correctly identified and cropped. Text detection, word segmentation, and cropping are excluded from this study. A full-fledged prescription digitization system requires these steps before word classification, and any mistakes made in this stage influence final classification performance.

The suggested framework contributes to prescription digitization by aiding in the recognition of medication names from cropped word images. Nevertheless, the testing of the framework relies solely on a balanced, word-level dataset. Hence, further validation requires a much broader and more diversified set of data, including full-page prescriptions, independent external validation samples, and clinical settings. In conclusion, the multi-backbone feature concatenation approach appears promise for handwritten prescription word recognition, though additional validation remains necessary.

6. Conclusions

This study developed a multi-backbone feature-concatenation framework using EfficientNet-B0, ViT-Base, and Swin-Base for handwritten prescription-word classification. The main conclusions are summarized as follows:

(1) The proposed framework achieved a mean accuracy of 88.44 ± 0.99% and a mean macro F1-score of 0.88 ± 0.01 over five independent runs, obtaining the highest mean performance among the predefined single- and dual-backbone baselines.

(2) The statistical analysis showed significant improvements over EfficientNet-B0 and ViT-Base, whereas the differences from Swin-Base and the dual-backbone configurations were primarily numerical rather than statistically significant.

(3) The ablation studies confirm that both network architecture and training strategies influence classification performance, providing useful guidance for future model design.

(4) The class-wise and qualitative analyses showed that the main errors occurred between visually similar medicine names with ambiguous, compressed, or connected handwritten characters.

(5) The efficiency analysis indicates that branch-pruned two-backbone variants can maintain comparable classification performance while reducing computational requirements, providing a promising direction for future lightweight deployment.

(6) The present study was limited to pre-cropped medicine-word images and evaluation on a held-out split of the same dataset. Therefore, the results should not be interpreted as evidence of performance across different writers, institutions, imaging conditions, or complete prescription pages.

Future research should investigate writer-independent and external validation, full-page prescription processing, confusion-aware learning, calibrated top-k predictions, lightweight multi-backbone architectures, and pharmacist-assisted verification.

The proposed framework demonstrates the potential of complementary multi-backbone feature learning for handwritten prescription-word classification. However, independent external validation and human verification remain necessary before the framework can be considered for clinical deployment.

Data Availability

The dataset used in this study is publicly available through Kaggle under the title “Doctor's Handwritten Prescription BD Dataset” [29]. The dataset consists of pre-cropped handwritten medicine-name images with predefined training, validation, and test partitions. No new dataset was created as part of this study. Experiments were conducted using the publicly available dataset without altering the class labels or dataset splits. The revised manuscript provides the currently accessible Kaggle URL and the date of access.

Conflicts of Interest

The authors declare no conflict of interest.

References

[1] L. J. Fajardo, N. J. Sorillo, J. Garlit, C. D. Tomines, M. B. Abisado, J. M. R. Imperial, et al., “Doctor's Cursive Handwriting Recognition System Using Deep Learning,” Proceedings of 2019 IEEE 11th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment, and Management (HNICEM), Laoag, Philippines, 2019.

[2] G. Pavithiran, S. Padmanabhan, N. Divya, A. V, I. J. P, and B. Chandar, “Doctor's Handwritten Prescription Recognition System In Multi-Language Using Deep Learning,” Proceedings of 2022 International Conference on Power, Energy, Control and Transmission Systems (ICPECTS), Chennai, India, 2022.

[3] O. Yakovchuk and M. Vasin, “Increasing the Accuracy of Handwriting Text Recognition in Medical Prescriptions with Generative Artificial Intelligence,” Technology Audit and Production Reserves, vol. 4, no. 2, article no. 72, pp. 18-21, 2023.

[4] A. Thavayee and T. Vijayakumar, "Analyzing Medical Manuscripts through Neural Networks and Handwriting Recognition Algorithms," Journal of Emerging Technologies and Innovative Research, vol. 11, no. 3, pp. f52–f57, 2024.

[5] A. Datta, M. M. Hasan, N. Mahmud, B. J. Ferdosi, R. Islam, Z. Rahman, et al., “Preserving Medical Information from Doctor's Prescription Ensuring Relation Among the Terminology,” Computers in Biology and Medicine, vol. 188, article no. 109812, 2025.

[6] S. Tabassum, N. Abedin, M. M. Rahman, M. M. Rahman, M. T. Ahmed, R. Islam, et al., “An Online Cursive Handwritten Medical Words Recognition System for Busy Doctors in Developing Countries for Ensuring Efficient Healthcare Service Delivery,” Scientific Reports, vol. 12, no. 1, article no. 3601, 2022.

[7] A. Y. Nimbalkar, K. P. Ubale, S. P. Patil, and V. H. Bhutnal, “MediCrypt: Survey on Automated Recognition of Handwritten Medical Prescriptions for Enhanced Healthcare Efficiency,” Proceedings of the International Conference on Multi-Strategy Learning Environment, pp. 413–430, 2024.

[8] R. Achkar, K. Ghayad, R. Haidar, S. Saleh, and R. Al Hajj, “Medical Handwritten Prescription Recognition Using CRNN,” Proceedings of 2019 International Conference on Computer, Information and Telecommunication Systems (CITS), Beijing, China, 2019.

[9] S. Tabassum, R. Takahashi, M. M. Rahman, Y. Imamura, L. Sixian, and M. M. Rahman,, “Recognition of Doctors' Cursive Handwritten Medical Words by Using Bidirectional LSTM and SRP Data Augmentation,” Proceedings of 2021 IEEE Technology & Engineering Management Conference-Europe (TEMSCON-EUR), Dubrovnik, Croatia, 2021.

[10] E. Hassan, H. Tarek, M. Hazem, S. Bahnacy, L. Shaheen, and W. H. Elashmwai, “Medical Prescription Recognition Using Machine Learning,” Proceedings of 2021 IEEE 11th Annual Computing and Communication Workshop and Conference (CCWC), NV, USA, 2021.

[11] N. Palani and N. Sampath, “Detecting and Extracting Information of Medicines from a Medical Prescription Using Deep Learning and Computer Vision,” Proceedings of 2022 International Conference on Knowledge Engineering and Communication Systems (ICKES) , Chickballapur, India, 2022.

[12] S. Samson, C. Tejaswini, G. Rishikesh, and M. Ramu, “Analysis of Scanned Medical Prescription Using Machine Learning,” International Journal of Innovative Science and Research Technology, vol. 9, no. 1, pp. 396-399, 2024.

[13] A. Razdan, A. Raghavan, M. Panandikar, A. Pol, K. Hedaoo, and R. Patil, “Recognition of Handwritten Medical Prescription Using CNN Bi-LSTM With Lexicon Search,” Proceedings of 2023 14th International Conference on Computing Communication and Networking Technologies (ICCCNT) , Delhi, India, 2023.

[14] A. R. Mia, M. A.-A.-S. Chowdhury, A. Al Mamun, A. M. Ruddra, and N. T. Tanny, “A Deep Neural Network Approach with Pioneering Local Dataset to Recognize Doctor's Handwritten Prescription in Bangladesh,” Proceedings of 2024 International Conference on Advances in Computing, Communication, Electrical, and Smart Systems (iCACCESS) , Dhaka, Bangladesh, 2024.

[15] M. N. Ismael, “A Deep Learning Approach for Multi-Class Classification of Handwritten Prescription Images Using CNN and Grad-CAM Visualization,” International Journal of Engineering in Computer Science, vol. 7, no. 2, pp. 215–225, 2025.

[16] G. R. Rekha, S. Siddesha, and V. N. M. Aradhya, “Treatment Regimen Segmentation from Handwritten Medical Prescriptions Using Advanced Neural Network,” Artificial Intelligence and Applications, in press, 2026.

[17] B. Biswas, B. K. Sarkar, A. Mustafi, and A. Mitra, “A Novel Approach of Doctors' Handwritten Medical Prescription Recognition Using A Deep Neural Network and Generating Digital Records of The Patients,” Journal of Dynamics and Games, pp. 1-36, in press, 2026.

[18] A. Maiti, A. Podder, C. Dutta, and D. Saha, “Improved RNN-based System for Deciphering Doctors' Handwritten Prescriptions,” American Journal of Advanced Computing, vol. 2, no. 2, pp. 1-5, 2023.

[19] T. Joshi, S. Das, S. Ghosh, C. Jha, and V. D. Shinde, “Web Application for Interpretation of Doctor's Handwritten Prescription and Suggesting the Best Price Offer Over Various E-Commerce Websites Using AI,” International Journal of Scientific Research in Computer Science, Engineering and Information Technology, vol. 9, no. 2, pp. 1-5, 2023.

[20] A. Al Mobassir, M. R. Hasan, M. A. Sadia Borni, M. R. Ahmed, and R. A. Prodhan, “Multimodal Deep Learning for Handwritten Prescription Recognition with Explainable AI,” Proceedings of 2026 IEEE 2nd International Conference on Quantum Photonics, Artificial Intelligence & Networking (QPAIN), Chittagong, Bangladesh, 2026.

[21] D. Dhar, A. Garain, P. K. Singh, and R. Sarkar, “HP_DocPres: A Method for Classifying Printed and Handwritten Texts in Doctor's Prescription,” Multimedia Tools and Applications, vol. 80, pp. 9779-9812, 2021.

[22] M. Agrawal, V. Mishra, H. S. Jogani, P. Kashyap, and R. G. Mishra, “A Review of Artificial Intelligence in Medical Prescription Analysis,” Proceedings of 2023 3rd International Conference on Pervasive Computing and Social Networking (ICPCSN), Salem, India, 2023.

[23] S. Vidhya, M. Yukendhira, and K. Sidharth, “MediScan Handwritten Prescription Translator Using Deep Learning,” Innovations and Advances in Cognitive Systems, vol 16, pp. 187-200, 2024.

[24] Y. Li, D. Chen, T. Tang, and X. Shen, “HTR-VT: Handwritten Text Recognition with Vision Transformer,” Pattern Recognit., vol. 158, article no. 110967, 2025.

[25] A. Taneja and S. Sharma, “Integrating EfficientNet and Vision Transformers for Enhanced Medical Image Classification: A Hybrid Approach,” Proceedings of the International Conference on Data Analytics & Management (ICDAM 2024), 2025.

[26] E. Anbazhagan and E. Sophiya, "A Novel Handwritten Prescription Recognition with Stochastic Gradient Descent Using Adaptive Momentum Learning,"Proceedings of International Conference on Recent Innovations in Computing (ICRIC 2023), Lecture Notes in Electrical Engineering, Springer, Singapore.

[27] A. Z. Tariq, F. Ahmed, A. Khandoker, and M. Rahman, “An AI-IoT Framework for Handwritten and Multilingual Prescription Interpretation with Timely Medication Reminder Support,” Proceedings of 2025 IEEE 49th Annual Computers, Software, and Applications Conference (COMPSAC), Toronto, ON, Canada, 2025.

[28] S. R. Hossain, S. Ahmed, S. Sofder, and R. Khan, "Doctor's Handwritten Prescription Recognition Using Cascading Lightweight Hybrid Adaptive Attention (CLHAA)," Proceedings of IEEE Conference Publication, 2025.

[29] M. Mamun, “Doctor's Handwritten Prescription BD Dataset,” Kaggle, 2024. [Online]. Available: https://www.kaggle.com/datasets/mamun1113/doctors-handwritten-prescription-bd-dataset. Accessed: July 4, 2026.