PalmReader.ai: A Deep Network Driven Fortune-Teller

CHUNG, Pok Yu LAM, Ching Yin LI, Ho Ming
The Hong Kong University of Science and Technology
Clear Water Bay, Hong Kong
pychungae@connect.ust.hk cylambu@connect.ust.hk hmliaf@connect.ust.hk

Abstract

Palm-reading is the practice of fortune-telling by reading one's palm. While it dates back to centuries ago, it is regarded as a pseudoscience by many due to a lack of scientific evidence, and is generally performed subjectively by humans. However, with modern statistical and deep learning techniques, we propose an automatic palm reading system by learning characteristics and hidden features within datasets. We construct a pipeline with vision transformers, object detection, and large language models to produce palm reading results and predictions from simply an image of the user's palm. Our project also uncovers the statistical connection between palm features and personal life experiences, creating a path for further related work.

1. Introduction

Fortune-telling through palm reading, known as palmistry, has been practiced for centuries. However, such methods lack scientific support and are regarded by many as pseudoscience. There is also a lack of evidence for the connection between palm line features and the predictions stemming from this form of fortune-telling.

In the age of machine learning, large datasets can be leveraged along with machine learning techniques, such as deep neural networks, to uncover hidden characteristics and connections that were previously difficult to observe. Under this predicate, we propose that there may be statistical connections between palm features and palmistry fortune-telling results.

In this project, we propose PalmReader.ai, an application to perform palmistry fortune-telling using an image of a palm. We construct a full pipeline to predict scores for aspects of the user's life, such as character, emotional predispositions, health status, and life path. Our pipeline is composed of the following components.

- **Vision Transformer (ViT)**: A vision transformer serves to extract features to vector embeddings that encode geometric features of palm lines in the input image.
- **Scoring system**: We score different aspects of life by examining the geometric features of the detected palm lines, using a features extracted from the vision transformer.
- **LLM inference**: We produce, from the and scores, a report in natural language that outputs fortune-telling content.

We obtain satisfactory results from experiments with various architectures, with the final mean squared error (MSE) loss at around 0.028. Inference using an LLM provides accurate reports on fortune-telling results based on our framework's predicted scores.

2. Related Work

2.1. Palmistry with Machine Learning

[3] proposes palmistry inference using random forests and linear SVM models. While the problem formulation is similar, we note that this paper classifies a palm line to different fixed categories, that eventually output to predefined descriptions.

Building upon this idea, we perform score prediction for each palm line and aspect of life, and utilize an LLM for natural language inference. This gives more informative and personalized outputs with our method.

2.2. U-Net Context Fusion Module

[5] proposes a deep learning-based solution to efficiently detect and segment palm lines from an image of a palm. This paper proposes a novel component, the Context Fusion Module (CFM), which is integrated into the U-Net as a bottleneck. The CFM captures both global and local contextual features from an image of the palm, using an attention mechanism to capture global and local contextual features from the palm image. Experimental results showed that the proposed method showed large advantages over traditional palm line segmentation methods.

Inspired by this method, we include a feature fusion module, combining features from regional feature extractors and global feature extractors. This allows us to extract

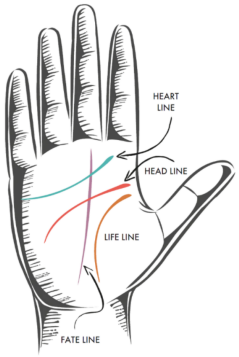

Figure 1. Different palm lines.

information from local to global levels of the image for further analysis and scoring.

3. Problem Formulation

3.1. Four Primary Lines for Palm Line Analysis

Palm line analysis is an ancient practice that interprets the lines on the palm of the hand to glean insights into an individual's personality traits, emotional tendencies, and life experiences. The four primary lines—Heart, Head, Life, and Fate—are central to this practice, each offering unique perspectives into various aspects of human life. All of our following tasks resolving these four lines [4]:

- **Heart Line**: Positioned at the top of the palm, the Heart Line is believed to reflect emotional stability, romantic perspectives, and cardiac health.
- **Head Line**: Running horizontally across the palm, the Head Line represents intellectualism, learning style, and communication abilities.
- **Life Line**: Encircling the thumb, the Life Line is traditionally associated with physical health, general well-being, and major life changes.
- **Fate Line**: Often running vertically from the base of the palm, the Fate Line indicates the degree to which external circumstances influence an individual's life path.

Therefore, one of our primary objectives is to successfully segment these primary palm lines. Achieving precise segmentation is essential before we can undertake further analytical tasks. This foundational step ensures that subsequent analysis will be based on accurate and distinct representations of each line.

3.2. Feature Extraction and Text Decoder

Following the segmentation of the palm lines, a critical subsequent step involves utilizing a text decoder to meticulously analyze the shapes and other significant characteristics that may influence the analysis. This detailed examination is essential before the data is passed to a large language model for comprehensive interpretation and final analysis.

3.3. Quantified Reasoning

To enhance the reasoning process and mitigate any uncertainties, it is essential to implement a scoring system for the features identified in the palm lines. This scoring will assess various attributes, such as indicators of luck, health, and other relevant characteristics. By quantifying these features, we can provide a more structured and objective analysis that informs subsequent interpretations and enhances the overall reliability of the insights derived from the palm analysis.

3.4. Final Analysis

After obtaining scores for various attributes and elaborating on the text descriptions of the palm lines, the final and critical step is to generate a comprehensive, user-friendly analysis. This task will be accomplished using a large language model (LLM), which will synthesize the scores and textual insights into a coherent narrative. The LLM will provide an accessible interpretation of the analysis, enabling users to easily understand the implications of the palm features concerning aspects such as luck, health, and personal insights. By leveraging the capabilities of the LLM, we aim to deliver a polished and informative report that enhances the user experience and facilitates deeper engagement with the results of the palm analysis.

4. Dataset Design

4.1. Data Origin and Characteristics

The data utilized in this study is purely based on images of hands, sourced from the Kaggle dataset titled "Human Palm Images." [1] To ensure a diverse and representative dataset, we selected images that capture an overhead view of hands, which aligns with our objective of scanning users' hands effectively. The dataset is well-balanced, encompassing both left and right hands, as well as involving both male and female images of 400 each. This careful selection aims to mitigate potential biases and ethical concerns associated with dataset representation. By prioritizing diversity in our dataset, we strive to prevent discriminatory practices and ensure that our findings are applicable across different demographics. This approach not only enhances the robustness of our analysis but also upholds ethical standards in research.

4.2. Data Preprocessing

The data preprocessing pipeline for our AI application involves several critical steps to enhance the quality of the input images and prepare them for effective analysis. We perform the following operations on the dataset.

4.2.1. Background Extraction

We isolate the palm from the surrounding background in an image. By effectively removing the background, we ensure that the model focuses exclusively on the palm, which leads to more accurate segmentation. We assume that the images are simple and there is no occlusion of our target objects.

4.2.2. Transformation to Grayscale

Grayscale conversion is the process of transforming a color image into a single-channel format that represents varying shades of gray. This technique simplifies the complexity of the image by reducing it from three color channels (RGB) to one, making it easier to process [2]. In the context of palm segmentation, grayscale images enhance the visibility of edges and contours while eliminating color distractions that may not be relevant to the task. By focusing solely on intensity values, we can better highlight the structural features of the palm, which is essential for accurately identifying and isolating it from the hand.

4.2.3. Black-Hat Morphological Transform

The Black-Hat morphological transform is a key preprocessing step for our palm image analysis, defined as

$$\text{BlackHat}(I) = \text{Closing}(I) - I$$

It isolates darker, smaller image structures—primarily palm lines—by subtracting the original image from its morphological closing. This effectively enhances thin, dark creases while suppressing uniform skin texture.

This operation provides targeted feature amplification before Vision Transformer (ViT) processing. It increases the signal-to-noise ratio of palm lines, allowing the ViT to focus its attention mechanisms on high-level relationships between enhanced features rather than low-level feature extraction. The transform's illumination robustness also aids in input normalization across varying capture conditions.

For implementation, the structuring element size is tuned to match typical palm line widths (10-20 pixels). The output is a grayscale, line-enhanced image suitable for standard ViT preprocessing (resizing, patching, projection), foregrounding the geometric information essential for robust palm representation learning.

4.3. Data Annotation

In our study, we utilize a Vision-Language Model (VLM) to perform the initial labeling of palm line images. The VLM leverages its ability to understand and process visual and textual information simultaneously, enabling it to generate accurate labels for the intricate patterns found in palm lines. This automated labeling process serves as a foundational step in our data annotation pipeline, significantly reducing the manual effort required and ensuring consistency across the dataset. By employing a VLM, we capitalize on advanced machine learning techniques to efficiently and effectively annotate our dataset, facilitating subsequent analysis and model training. This approach not only enhances the quality of the initial labels, but also accelerates the overall data preparation process.

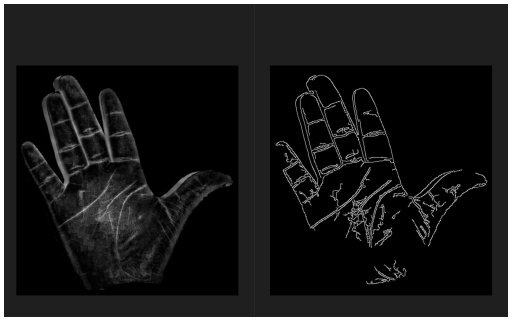

Figure 2. Example of preprocessing steps performed on an input palm image.

```
{
  "image": "FEMALE/IMG_0025.JPG",
  "scores": {
    "luck": 0.9,
    "health": 0.7,
    "romantic": 0.4.
  }
}
```

Figure 3. An example of labels produced by a VLM, in JSON format.

5. Methods

Our system is designed as a novel, multi-modal pipeline that integrates computer vision and natural language processing to translate palm features into insightful descriptions. The architecture moves beyond a simple feature-to-text model by incorporating a rule-based scoring mechanism and a final Large Language Model (LLM) synthesis step for coherent and contextualized output.

5.1. Feature Extraction

Input images are first converted to grayscale and denoised with Gaussian blur before edge extraction. The preprocessing module loads an image, converts it from BGR to grayscale, applies a configurable Gaussian blur, and then computes edges using Canny detection, producing an "edge-only" version that is saved per image.

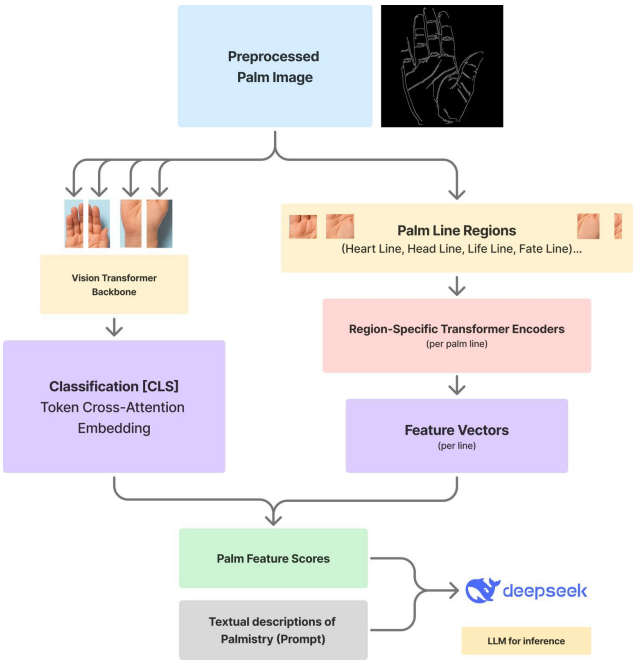

Figure 4. Our pipeline structure.

5.1.1. Preprocessing

The pipeline begins by preprocessing the raw palm image to standardize input and highlight regions of interest. This involves:

- **Color Correction & Normalization**: Adjusting for lighting variations to ensure consistent color and contrast across images.
- **Pale Region Isolation**: A specialized segmentation model (e.g. a lightweight U-Net) is employed to accurately identify and crop the four primary pale regions corresponding to the mounts of the palm (e.g., Mount of Venus, Mount of Jupiter, etc.), as well as the central palmar area containing the primary lines (Heart, Head, Life).
- **Image Patching**: The isolated full palm and/or specific regions are divided into a sequence of fixed-size patches, which are then linearly projected into a lower-dimensional space to create a patch embedding sequence.

5.1.2. Feature Extraction with Hybrid Transformer

Instead of a single transformer, we propose a Hybrid Transformer Encoder architecture with the following features, as shown in Figure 6.

- **Shared Backbone**: A Vision Transformer (ViT) backbone first processes the entire preprocessed palm image to generate a global context embedding, represented by the [CLS] token.
- **Region-Specific Encoders**: In parallel, the cropped image regions containing specific palm lines (Heart, Head, Life, Fate) are processed by smaller, independent transformer encoders. This allows for focused feature extraction tailored to the unique characteristics of each line.
- **Feature Fusion**: The global [CLS] token embedding and the features from the region-specific encoders are concatenated and passed through a fusion module (e.g., a cross-attention layer or a simple fully connected layer) to create a unified, comprehensive feature representation F_{unified}.

5.2. Feature Processing and Multi-Modal Interpretation

This stage interprets the fused features F_{unified} through two parallel paths.

5.2.1. Text Decoder

A pre-trained text decoder (e.g., a Transformer Decoder or a GPT-2 architecture) takes F_{unified} as input. It is trained to generate structured, factual descriptions in natural language for each palm line in a template-like format. Our template is defined as the following.

- **Input**: F_{unified} (Feature vector for a specific palm).
- **Expected Output Format**: "The heart line is long and curvy. The head line is deep and long. The life line curves completely around the thumb."

5.2.2. Embeddings and Score Prediction

This branch translates the image features into a quantitative and comparable format.

- **Embedding Generation**: The feature vector F_{unified} is projected into a lower-dimensional, normalized embedding space (e.g., using a dedicated Embedding Layer). The core hypothesis is that palms with similar palmistry traits will lie close to each other in this space (high cosine similarity). We train our model using contrastive methods, by minimizing the dot product between embeddings of palms with similar traits, and maximizing that for embeddings of palms with traits that differ more.
- **Rule-Based Scoring Engine**: This is a novel, explicit component. A set of deterministic, human-defined rules based on traditional palmistry (e.g., "if the life line is curved, add 5 points to vitality score") is applied directly to the structured descriptions generated by the Text Decoder branch. This generates a score vector S_{rule} (e.g. [Vitality: 8, Creativity: 5, Logic: 3]).
- **Neural Network Score Predictor**: In parallel, a shallow Neural Network (NN) processes the unified embedding to predict a complementary score vector S_{NN}. The model is trained so that S_{NN} correlates with the rule-based scores S_{rule} while also learning subtle, non-linear patterns from the data.

5.3. LLM Inference for Holistic Synthesis

The final stage leverages a Large Language Model (LLM) as a sophisticated information synthesizer.

- **Input Prompt Construction**: The outputs from previous stages are assembled into a detailed prompt for the LLM. We include structured descriptions from the text decoder, score vectors S_{NN} and S_{rule}, as well as a system prompt framing the task.
- **Natural Language Generation**: The LLM (e.g., a fine-tuned Llama 2 or GPT-3.5 model) processes this rich, multi-modal input to produce the final output. It does not just paraphrase; it reasons about the interplay between different lines and scores, generating a novel, insightful, and human-readable report that includes character analysis and life suggestions. This step adds a layer of interpretation and narrative fluency that is absent in simpler sequence-to-sequence models.

6. Experiments

6.1. Preprocessing

6.1.1. Experimental Setup

Three datasets were constructed from the same 800 labeled palm images (train/validation split 640/160):

1. original images without preprocessing,
2. grayscale+blur preprocessed images, and
3. preprocessed images with edge extraction applied using the Canny-based edge extractor described above.

For all three datasets, the network architecture, optimization settings, and training schedule were kept identical to isolate the effect of the input representation on optimization behavior.

6.1.2. Convergence Comparison

Training and validation losses over 5 epochs were recorded for each dataset and plotted on a single curve for comparison. The original-image setting consistently produced the highest training and validation losses; applying basic preprocessing reduced both losses, while adding edge extraction on top of the preprocessing yielded the lowest losses and the steepest early decrease, indicating faster and more stable convergence. Therefore, subsequent experiments in the paper are conducted using this representation as the default input to the model.

6.2. Vision Transformer for Palm Image Regression

6.2.1. Architecture Overview

The Vision Transformer (ViT) processes palm images for regression of four scores: strength, romantic, luck, and potential. Key components:

- **Backbone**: Pre-trained ViT-Base (12 layers, 768-dim, patch size 16)
- **Input**: 224×224 RGB $\rightarrow$ 196 patches
- **Regression head**: 3-layer MLP with GELU, dropout, and sigmoid output
- **Output**: Four scores in $[0, 1]$ range

```
You are a palmistry expert.
You will assist the user in inferring
fortune-telling results based on their
palm line features.

## Supplementary Information on Palmistry
Palm lines of interest include the *Life
Line*, the *Heart Line*, the *Fate Line*,
and the *Head Line*. These lines
represent respectively, *enthusiasm
and strength*, *romantic life*, *fortune
and luck*, and *smartness and potential*.
You will receive scores predicted based
on palm line features by a Deep Learning
model. You are then to provide fortune-
telling results in natural language,
further instructions will be given in
the next section.

### About Each Palm Line
<supporting information>

## Inference Instructions
You will be provided with scores for
'strength', 'romantic', 'luck', and
'potential'. These scores were previously
predicted using a deep learning model
based on geometric features of the user's
palm lines, and they correspond to the
above points provided to you as
supplementary information on palmistry.

Provide the user with fortune-telling
results based on these scores, and give
explanations to assist the user in
inferring their results. Explain what
those scores mean and how their luck
or fortune looks like. Give advice to
the user on how to face probable events
in their future life.
```

Figure 5. Contextual instructions provided through the system prompt to the LLM.

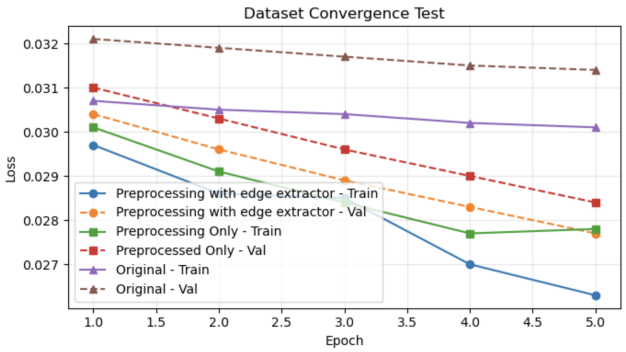

Figure 6. Comparison of convegence speeds when using datasets with different preprocessing steps for training.

6.2.2. Training Configuration

Parameter	Value
Batch size	8
Learning rate	2×10^{-4}
Epochs	80
Warmup	5% (fast)
Optimizer	AdamW (wd=0.01)
Loss	MSE
Scheduler	OneCycleLR

6.2.3. Data Processing

- **Augmentation**: Random crop, flip, rotation ($\pm 15°$), color jitter
- **Normalization**: Mean = 0.5, std = 0.5
- **Labels**: JSON parsing with error handling

6.2.4. Training Results

The model achieved strong performance in palm score regression:

- **Best model at epoch 45** (early convergence)
- **Validation loss**: 0.0186 MSE
- **Validation MAE**: 0.1002 (average absolute error)
- **Practical implication**: Predictions within ± 0.1 of true values

6.2.5. Training Curves

Training performance is visualized through loss curves showing model convergence:

6.2.6. Key Advantages

- **Global context**: Self-attention captures relationships across entire palm
- **Transfer learning**: ImageNet-21k pre-training provided strong initialization
- **Fast convergence**: Best performance reached at epoch 45 of 80
- **High accuracy**: MAE of 0.1002 indicates precise score prediction

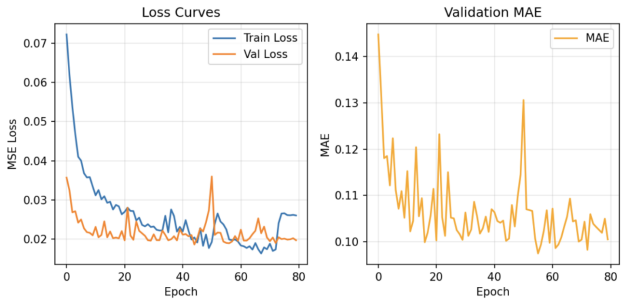

Figure 7. Training curves showing convergence at epoch 45. The model achieved validation loss 0.0186 and MAE 0.1002, indicating effective learning of palm features.

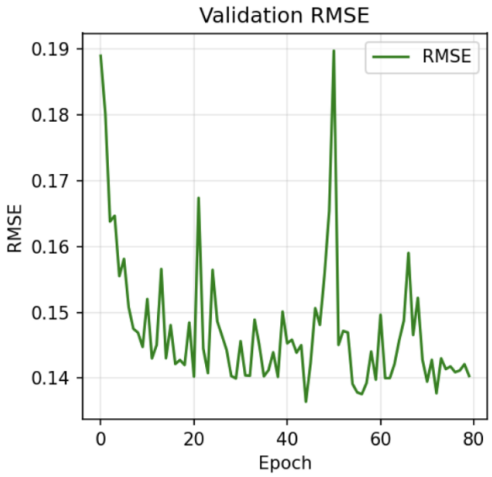

Figure 8. Root Mean Square Error (RMSE) curves during training. The model's RMSE decreased steadily till epoch 45.

6.3. Alternative Framework Using CNNs

We also explore an alternative implementation of our framework using different CNN architectures in place of our vision transformer. The architectures used and results are listed as follows.

6.3.1. Training Configuration

For all CNN architectures trained and tested, the parameters are the same and are listed in the following table.

Parameter	Value
Batch size	8
Learning rate	1×10^{-4}
Dropout rate	0.2 or 0.3
Epochs	20
Optimizer	AdamW (wd=0.01)
Loss	MSE
Scheduler	ReduceLROnPlateau

6.3.2. Multi-scale Feature Extraction CNN

This architecture performs multiple convolutions to extract features at different levels, then passes the features through an attention mechanism to focus on important palm line regions, and finally fuses the feature outputs to regress to 4 scores. The detailed flow is as follows.

- Initial convolution: 1 channel to 32 channels, kernel size 3, padding 1
- Mid-level features: 32 channels to 64 channels, kernel size 3, padding 1
- High-level features: 64 channels to 128 channels, kernel size 3, padding 1
- Attention (from high-level features): 2D Conv from 256 channels to 1 channel, kernel size 1, padding 0
- Global features (from attended features): 128 channels to 256 channls, kernel size 3, padding 1
- Further upscaling (from high-level features): 128 channels to 256 channls, kernel size 3, padding 1
- Feature fusion: combines further upscaling features and global features
- Regression to 4 scores: linear layers
- ReLU layer to clip values between 0 and 1

This architecture gave an average MSE loss of 0.0168 on evaluation using 800 images. The final training loss was 0.0096, and the best validation loss was 0.0231.

6.3.3. Transfer Learning CNN

This architecture uses a pretrained backbone, either EfficientNet or ResNet, to extract features. The extracted features are then flattened and passed through linear layers, and finally a regression layer to output to 4 scores. A sigmoid activation is used before outputting the scores to clip scores between 0 and 1.

This architecture gave an average MSE loss of 0.0075 on evaluation using 800 images, the best among all CNN architectures tested. The final training loss was 0.0063, and the best validation loss was 0.0251.

Since our dataset is rather small and only consists of 1600 images, we find transfer learning to be the most efficient and provides the best results.

6.3.4. Region-specialized CNN

This architecture uses a shared backbone to first extract features, then passes the features through 4 parallel branches, each attending to a different palm region, and finally concatenates the features from the branches to regress to 4 scores. The detailed flow is as follows.

- Shared backbone: 2D convolutionse to 32 then 64 and 128 channels, kernel size 3, padding 1
- Parallel branches:
  - Attention in each branch: 2D convolution from 128 to 64 then 32 channels, kernel size 1, padding 0
  - Region extractor in each branch: 2D convolution from 128 to 64 channels, kernel size 3, padding 1

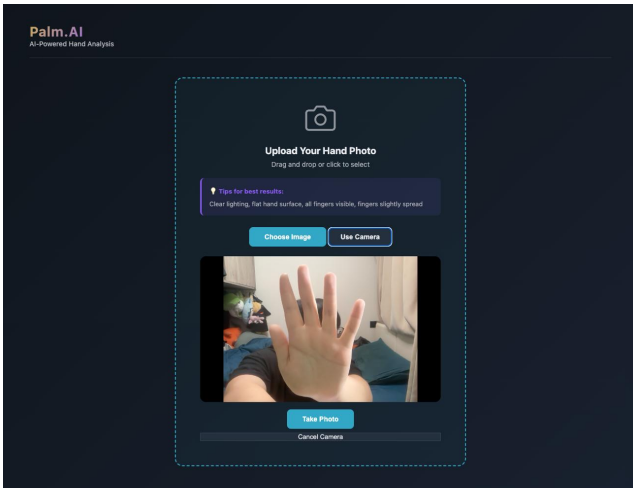

Figure 9. User interface prompting for palm scanning.

- Feature concatenation: linear layer from 64×4 dimensions to 256 dimensions
- Regression to scores: linear layers from 256 dimensions to 128 then 4 dimensions
- ReLU layer to clip values between 0 and 1

This architecture gave an average MSE loss of 0.0304 on evaluation using 800 images. The final training loss was 0.0191, and the best validation loss was 0.0237.

We hypothesize that each palm region has different semantic meanings, hence each gives one score per aspect of life. Thus in this architecture, we use parallel convolution operations on each region to extract features and translate those features to scores.

6.3.5. Summary of CNN Training

Architecture	Avg. Loss	Train Loss	Val. Loss
Multi-scale	0.0168	0.0096	0.0231
Transfer Learning	0.0075	0.0063	0.0251
Region-specialized	0.0304	0.0191	0.0237

6.4. User Interface

We provide a graphical user interface for seamless use of our pipeline, through a web app. The user first scans their palm using an on-device camera, and will receive predictions in natural language with an LLM.

7. Conclusion

Palmreader.ai establishes a multi-modal pipeline that performs fortune-telling through palm reading, incorporating Vision Transformers (ViT), a Neural Network for scoring, as well as a Large Language Model (LLM) for final inference. We have provided a first step in using a data-driven approach to draw superstitious practices towards science.

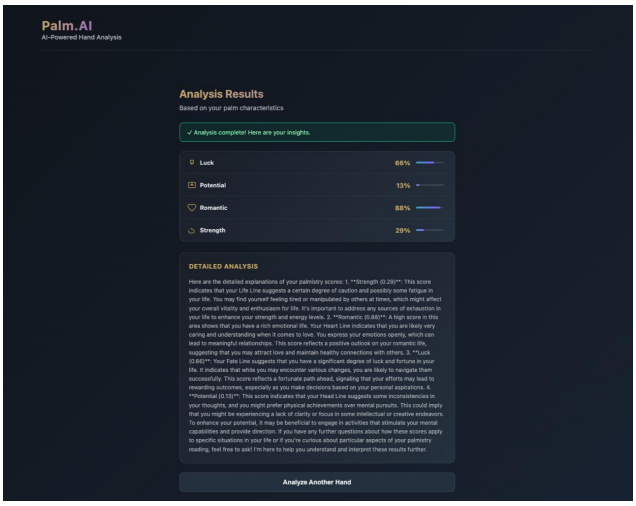

Figure 10. Prediction by LLM.

We expect to produce a robust model that can effectively generate accurate palm-reading results.

Further work may optimize the pipeline by employing more powerful pretrained text decoders. Additionally, external data related to palmistry may be used for Retrieval-Augmented Generation (RAG) with the LLM in the final inference step, to provide context whenever required as additional support.

8. Appendix

Our implemention is linked at this Github repository. Additionally, this is a demo of the final app usage.

References

[1] Feyisayo Amujoyegbe. Human palm images. 2022. 2

[2] Daniel Louise M Parulan, Jon Neil P Borcelis, and Noel B Linsangan. Palm lines recognition using dynamic image segmentation. In *2024 IEEE International Conference on Automatic Control and Intelligent Systems (I2CACIS)*, pages 238–243. IEEE, 2024. 3

[3] Shweta Patil. Palmistry-informed feature extraction and analysis using machine learning, 2025. 1

[4] Big Brothers Big Sisters. Palm reading. *unknown*, 2018. 2

[5] Toan Pham Van, Son Trung Nguyen, Linh Bao Doan, Ngoc N. Tran, and Ta Minh Thanh. Efficient palm-line segmentation with u-net context fusion module. page 23–28, 2020. 1