CAFA 6 Protein Function Prediction Report

Trung Hau Tran, Trung Hieu Pham, Nam Khanh Pham
VNU University of Engineering and Technology
Hanoi, Vietnam

***Tóm tắt nội dung*—Bài báo cáo này trình bày phương pháp của chúng tôi cho CAFA 6 Protein Function Prediction. Cuộc thi này tập trung vào việc xây dựng mô hình học máy có khả năng dự đoán được chức năng sinh học của protein chỉ dựa trên trình tự axit amin của các protein đó. Dữ liệu gồm các chuỗi protein và trình tự axit amin tương ứng, cùng với các nhãn chức năng sinh học được gán với Gene Ontology (GO).**

Chúng tôi áp dụng phương pháp học máy và học sâu, kết hợp các kỹ thuật trích xuất đặc trưng , lựa chọn mô hình embedding phù hợp và kĩ thuật ensemble mô hình để xây dựng mô hình dự đoán chức năng protein.

Điểm cuối cùng đạt được là 0.358, top 56/460 đội (tính đến ngày 9/8/2025).

I. Giới thiệu

Câu hỏi "Liệu có sự sống nào tồn tại ngoài Trái Đất?" từ lâu đã là một trong những vấn đề lớn nhất của thiên văn học và khoa học vũ trụ. Để tiếp cận câu hỏi này, các nhà khoa học đã triển khai nhiều hướng nghiên cứu, từ việc tìm kiếm tín hiệu vô tuyến, quan sát hành tinh ngoài Hệ Mặt Trời, cho đến phân tích thành phần hóa học của các thiên thể. Trong số đó, phương pháp phân tích quang phổ khí quyển hành tinh nổi lên như một hướng nghiên cứu đầy hứa hẹn, vì có thể cung cấp thông tin trực tiếp về điều kiện vật lý – hóa học liên quan đến khả năng tồn tại sự sống.

Tuy nhiên, phân tích quang phổ của các hành tinh ngoài Hệ Mặt Trời vẫn còn nhiều thách thức, đặc biệt do tín hiệu thu được thường rất yếu và bị nhiễu mạnh từ sao chủ cũng như các yếu tố quan sát. Hiện chưa có một phương pháp tối ưu nào được công nhận rộng rãi, đòi hỏi cộng đồng khoa học phải tìm kiếm các giải pháp mới thông qua cả nghiên cứu lý thuyết lẫn thực nghiệm mô phỏng.

Một trong những cách thúc đẩy tiến bộ trong lĩnh vực này là tổ chức các cuộc thi mô phỏng dữ liệu quan sát, tạo điều kiện cho các nhà nghiên cứu và kỹ sư dữ liệu thử nghiệm, so sánh và cải thiện phương pháp. Cuộc thi Ariel Data Challenge – NeurIPS 2025 (ADC2025) là một ví dụ tiêu biểu, với mục tiêu xây dựng mô hình dự đoán quang phổ của các hành tinh dựa trên dữ liệu quan sát giả lập sát thực tế.

Trong báo cáo này, chúng tôi trình bày các phương pháp đã áp dụng khi tham gia ADC2025, bao gồm mô hình hóa bài toán, xây dựng khung quy trình tiền xử lý dữ liệu, tiếp cận bài toán theo hai hướng: không sử dụng học máy (non-ML) và học máy (ML).

II. Background

A. Phương pháp phân tích quang phổ

Phân tích quang phổ tách ánh sáng thành các bước sóng để xác định thành phần hóa học, nhiệt độ, áp suất và đặc điểm khí quyển của thiên thể. Các phương pháp chính gồm:

- **Hấp thụ** – đo suy giảm cường độ tại các bước sóng đặc trưng.
- **Phát xạ** – ghi nhận bức xạ do vật thể phát ra.
- **Truyền qua** – quan sát ánh sáng sao đi qua khí quyển hành tinh khi quá cảnh.
- **Phản xạ** – phân tích ánh sáng phản xạ để xác định suất phản chiếu và bề mặt.

Tuy nhiên, để từ những dữ liệu nhiễu muốn có được thông tin chính xác của hành tinh, ta vẫn cần giải pháp trong tiền xử lý và phân tích dữ liệu. Trong bối cảnh đó, ADC2025 xuất hiện với mong muốn cùng cộng đồng chung tay thúc đẩy tiến trình nghiên cứu này.

B. CAFA 6 Protein Function Prediction

C. Tập dữ liệu

Bộ dữ liệu trong thử thách Ariel bao gồm các quan sát mô phỏng từ hai thiết bị cảm biến chính của tàu Ariel:

- **FGS1** (Fine Guidance Sensor): cung cấp chuỗi ảnh 32×32 pixel, độ phân giải thời gian 0.1s trong phổ ánh sáng khả kiến (0.60–0.80 µm).
- **AIRS-CH0** (Ariel Infrared Spectrometer): cung cấp chuỗi ảnh 32×356 pixel, độ phân giải thấp hơn nhưng nằm trong phổ hồng ngoại (1.95–3.90 µm).

Mỗi hành tinh có thể có một hoặc nhiều lần quan sát, được lưu thành các chuỗi thời gian gồm hàng chục nghìn khung hình (FGS1: 135,000 frames; AIRS: 11,250 frames). Các ảnh được lưu dưới dạng mảng phẳng `uint16` và cần được hiệu chỉnh về động dải thực bằng cách sử dụng các tham số gain và offset trong tệp `adc_info.csv`.

Ngoài dữ liệu ảnh, thử thách còn cung cấp:

- **Dữ liệu hiệu chuẩn**: gồm các tệp `dark`, `flat`, `dead`, `linear_corr`, và `read noise` cho từng thiết bị.
- **Siêu dữ liệu vật lý**: thông tin quỹ đạo và đặc điểm vật lý của hệ sao-hành tinh như khối lượng, bán kính, nhiệt độ sao, độ nghiêng quỹ đạo, v.v.
- **Ground truth**: phổ thực (283 điểm phổ).

So với cuộc thi năm 2024, phiên bản 2025 tăng độ chân thực với các yếu tố vật lý như tối rìa sao (limb darkening), có thể nhiều hơn 1 quan sát cho 1 hành tinh.

III. Phương pháp

A. Tiền xử lý dữ liệu

Với số hành tinh n, trong thời gian quan sát t, có s điểm chụp trong không gian và wl bước sóng, một đầu vào của quy trình tiền xử lý sẽ là một tensor có kích thước $n \times t \times s \times wl$.

Qua các bước xử lý bên dưới, giả sử ta lấy a bin theo trục thời gian, lấy trung bình theo trục không gian và b đến c theo trục bước sóng, ta thu được tensor đầu ra có kích thước $n \times \frac{t}{a} \times (c - b + 1)$.

Dưới đây là quy trình tuần tự gồm các bước hiệu chỉnh để xử lý dữ liệu thô chứa nhiều tạp nhiễu vật lý sử dụng 5 tệp dữ liệu hiệu chuẩn:

- dark: ảnh tối, chụp khi không có ánh sáng (đóng shutter), dùng để loại bỏ dòng tối, mỗi giá trị pixel là nhiễu được thêm vào tín hiệu trong 1 giây
- dead: xác định dead pixels, đồng thời cũng được dùng để xác định những hot pixels
- flat: mỗi giá trị pixel thể hiện độ nhạy của điểm ảnh đó trong cảm biến, dùng để chuẩn hoá sự khác biệt độ nhạy giữa các pixel
- linear_corr: đa thức hiệu chỉnh phi tuyến tính của cảm biến
- read: ảnh mô tả nhiễu đọc của cảm biến

1) Đảo ngược chuyển đổi ADC (Analog-to-Digital Converter): Tín hiệu thu dưới dạng số nguyên (unit16) nhờ bộ chuyển đổi ADC. Ta phục hồi giá trị thực (float) của signal bằng công thức:

$$signal = \frac{raw}{gain} + offset \tag{1}$$

Trong đó, raw là giá trị thu được từ các cảm biến, $gain$ là hệ số khuếch đại và $offset$ là độ lệch.

Ví dụ, gain = 0.4369 và offset = -1000 cố định cho ADC2025 (Bạn có thể tìm thấy các giá trị này trong tệp `adc_info.csv`).

2) Điểm chết (dead pixels) và điểm nóng (hot pixels): Trong quá trình xử lý tín hiệu từ cảm biến, hai loại lỗi phổ biến cần được loại bỏ là *điểm chết* (dead pixels) và *điểm nóng* (hot pixels).

- **Điểm chết (dead pixels)** là các pixel không phản ứng với ánh sáng, tức luôn ghi nhận giá trị rất thấp hoặc bằng 0, bất kể tín hiệu thực tế.
- **Điểm nóng (hot pixels)** là các pixel có giá trị bất thường cao, ngay cả trong điều kiện không có ánh sáng, thường do lỗi điện tử hoặc nhiễu nhiệt.

Để phát hiện các điểm nóng, thuật toán `sigma_clip` từ thư viện `astropy` được áp dụng trên dữ liệu ảnh tối (`dark`). Thuật toán này lặp lại việc loại bỏ các pixel có giá trị vượt quá ngưỡng 5σ so với trung bình của các pixel lân cận. Sau một số lần lặp, mặt nạ (mask) của các điểm nóng được tạo ra.

Các điểm nóng và điểm chết sau đó được đưa vào mặt nạ hiệu chuẩn `flat` (ta sẽ nói về `flat` ở phần cuối cùng), gán giá trị `NaN` nhằm loại trừ khỏi quá trình hiệu chỉnh:

3) Non-linearity correction: Các pixel trong cảm biến không luôn phản hồi tuyến tính với cường độ ánh sáng chiếu vào. Khi tín hiệu gần đạt đến mức bão hoà, các pixel sẽ tích điện chậm lại, dẫn đến sai lệch giữa tín hiệu đo được và giá trị thực. Để hiệu chỉnh sai lệch này, một đa thức (thường là bậc hai hoặc cao hơn) được sử dụng với các hệ số được lưu trong biến *linear_corr*. Trong notebook của chúng tôi, chúng tôi áp dụng một đa thức bậc 5 để hiệu chỉnh tín hiệu, như sau:

$$linear\_corr = a \cdot raw^5 + b \cdot raw^4 + c \cdot raw^3 + d \cdot raw^2 + e \cdot raw + f \tag{2}$$

Trong đó, các hệ số được xác định thông qua dữ liệu trong file *linear_corr.parquet*.

Chú ý rằng, trước khi áp dụng hiệu chỉnh, tín hiệu nên được clip về 0 vì dữ liệu sẽ có thể có một số pixel với giá trị âm do bước ADC convert ở phần 1 với offset là giá trị âm, chúng được gây ra bởi nhiễu ngẫu nhiên mà đã được đưa vào tập dữ liệu trong quá trình mô phỏng. Đôi khi điều này xảy ra khi tín hiệu quá yếu. Mặt khác, đa thức mà dữ liệu bài toán cho để hiệu chỉnh ở trên không hoạt động hiệu quả để xử lý các giá trị âm vì chúng dường như đã được fit từ những giá trị dương. Do đó, nếu giữ nguyên các giá trị âm đó thì sẽ gây sai lệch lớn trong quá trình hiệu chỉnh.

4) Dark frames: Khung tối là ảnh phơi sáng được chụp với màn trập đóng, ghi lại nhiễu nhiệt và độ lệch của cảm biến. Chúng được sử dụng để loại bỏ dòng tối khỏi ảnh.

$$signal = signal - dark \cdot \Delta t \tag{3}$$

với Δt là tổng thời gian phơi sáng của ảnh (thông tin về integration time có thể được tìm thấy trong file `axis_info.parquet`).

Khi bạn mở file axis_info.parquet, bạn sẽ thấy cột *integration_time* chứa thời gian phơi sáng của từng khung hình chỉ bao gồm các giá trị 0.1s và 4.5s. Điều này phù hợp với các thông số kỹ thuật của cảm biến FGS1 và AIRS-CH0. Hãy để chúng tôi nói rõ hơn về cách hoạt động của 2 cảm biến này. Cảm biến hoạt động ở chế độ **CDS (Correlated Double Sampling)**, nghĩa là sẽ hoạt động theo các chu kỳ gọi là `integration ramp`. Mỗi integration ramp bắt đầu với detector ở trạng thái ban đầu, tích điện được kích hoạt thời điểm này. Sau 0.1 giây, nó sẽ đọc tín hiệu đầu tiên (NDR0). Quá trình đọc 1 frame mất 0.1s, nhưng giả thuyết bài toán này không sử dụng phương pháp đọc tuần tự mà thay vào đó, tất cả pixel được đọc đồng thời. Sau khi NDR0 được chụp, detector sẽ chuyển sang giai đoạn đợi chờ để thu thập ánh sáng. Khoảng thời gian chờ này chính là cột integration time trong tập dữ liệu. Sau pha chờ này, detector sẽ đọc tín hiệu lần thứ hai (NDR1) và cuối cùng là reset detector để chuẩn bị cho ramp tiếp theo. Quá trình reset này cũng cần 0.1 giây. Bạn có thể kiểm tra điều này bằng cách so sánh dòng thời gian của các khung hình với thời gian tích hợp (trong file `axis_info.parquet`). Để tham khảo, đây là dòng thời gian của hai máy dò:

- **FGS1 (integration time = wait =** $0.1\,\mathrm{s}$**):**

```
|-ground-|-NDR0-|-wait-|-NDR1-|-reset-|
0.0     0.1      0.2    0.3    0.4      0.5
```

- **AIRS (integration time = wait =** $4.5\,\mathrm{s}$**):**

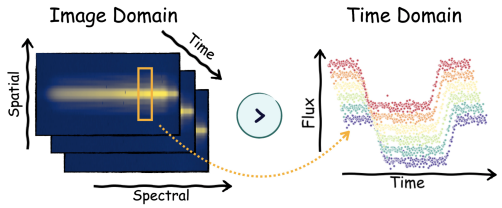
```
|-ground-|-NDR0-|-wait-|-NDR1-|-reset-|
0.0     0.1      0.2    4.7    4.8      4.9
```

Như vậy, ta cần chú ý ở đây rằng: thời gian phơi sáng các frame đầu tiên ở 2 cảm biến sẽ đều là 0.1 giây (do pha ground kéo dài 0.1 giây). Sự khác nhau giữa 2 cảm biến nằm ở frame2. Với giả thuyết các pixel được chụp đồng thời thì tổng thời gian phơi sáng Δt của frame 2 ở FGS1 là 0.2 giây, với AIRS là 4.6 giây.

5) Flat field correction: Lý do cho việc hiệu chỉnh flat field là để loại bỏ sự khác biệt về độ nhạy giữa các pixel trong cảm biến. Cảm biến trong máy ảnh không hoàn hảo, có pixel nhạy hơn, có pixel kém nhạy hơn, có thể có bóng mờ, bụi, hay sự không đồng đều trong hệ quang học. Do đó, khi chụp một vùng sáng đồng đều (ví dụ: ánh sáng trắng trải đều), ảnh thu được không hề đồng đều, mà có vùng sáng – vùng tối do sự khác biệt trong cảm biến. Vì vậy, để thu được độ nhạy cảm mỗi pixel, người ta chụp một ảnh với nguồn sáng đồng đều, ảnh thu được gọi là flat field, mỗi pixel trong ảnh này thể hiện độ nhạy sáng của pixel đó. Tín hiệu mà ta thu được sẽ tỉ lệ với độ nhạy của pixel nhân với tín hiệu thực tế từ bản chất ánh sáng đó. Do đó, với một ảnh bất kỳ thu được, để hiệu chỉnh nó để thu được tín hiệu thực từ nguồn sáng, ta phải thực hiện chia cho flat field (độ nhạy sáng của mỗi pixel).

Tín hiệu pixel khác nhau do hiệu suất quang học khác nhau. Cần hiệu chỉnh bằng cách dùng flat mapping để chuẩn hoá:

$$signal_{corrected} = \frac{signal}{flat\_field} \tag{4}$$



Hình 1. Pipeline tiền xử lý dữ liệu

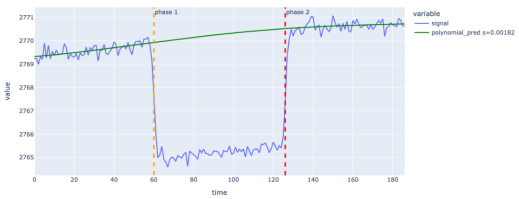
6) Correlated Double Sampling (CDS): Trước khi tính CDS, ta cần làm gọn dữ liệu bằng cách loại bỏ đi các phần không quan trọng của frame, cũng như làm một số thao tác để lấy ra con số biểu diễn chung nhất cho một frame. Sau đó, ta tính CDS bằng cách lấy hiệu ảnh chụp được lúc kết thúc và bắt đầu của mỗi ramp, cụ thể là:

$$CDS = signal_{end} - signal_{start} \tag{5}$$

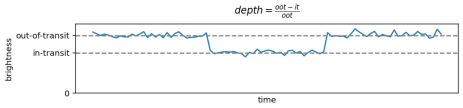
7) Binning: Binning là quá trình kết hợp tín hiệu của nhiều bước thời gian thành một để tiết kiệm không gian bộ nhớ cũng như tính toán. Điều này không làm giảm hiệu quả của model vì ta nhận thấy, các hành tinh di chuyển rất chậm, do đó, các ảnh liên tiếp nhau không có sự khác nhau nhiều nên việc binning sẽ không làm mất đi thông tin quan trọng mà còn giảm được nhiễu nhờ cách tính trung bình trên mỗi bin.

B. Tiếp cận theo hướng không dùng học máy

Trong bối cảnh bài toán này, có một phương pháp non-ML mang lại hiệu quả đáng kinh ngạc so với các phương pháp sử dụng thuật toán Machine Learning, Deep Learning. Chúng tôi tạm gọi nó là heuristic method. Có một số lý do khiến model này hiệu quả. Thứ nhất, chúng tôi nhận thấy khi lấy trung bình tín hiệu trên từng bước thời gian sau khi xử lý tín hiệu, ta được Hình 2 và Hình 3.



Hình 2. Trung bình tín hiệu trên từng bước thời gian



Hình 3. Insight về hiện tượng transit

Có thể quan sát thấy rằng tín hiệu thu được thường có dạng lõm ở phần giữa, phản ánh hiện tượng hành tinh đi qua phía trước ngôi sao chủ (transit), khiến cường độ ánh sáng giảm xuống do bị che khuất. Độ suy giảm này đạt cực đại khi hành tinh nằm tại vị trí trung tâm của ngôi sao trên đường đi của nó.

Dựa trên hiện tượng vật lý này, thay vì huấn luyện một mô hình học máy, chúng tôi đề xuất một phương pháp tiếp cận thực nghiệm, trong đó tìm kiếm trực tiếp một hệ số khuếch đại tối ưu cho từng phổ tín hiệu.

Cụ thể, với mỗi phổ, chúng tôi thực hiện phát hiện khoảng thời gian xảy ra transit, bằng cách xác định hai điểm mốc là `phase1` và `phase2`—tương ứng với thời điểm bắt đầu và kết thúc của quá trình transit. Quá trình này được thực hiện thông qua hàm `phase_detector`, dựa trên đạo hàm bậc nhất của tín hiệu. Đạo hàm được sử dụng bởi vì ta thấy phase1 là nơi tín hiệu giảm nhanh nhất, phase2 là điểm tín hiệu tăng nhất nhất. Do đó, ta dùng đạo hàm bậc nhất để tìm điểm có đạo hàm cực tiểu và cực đại trong khoảng có tiềm năng.

Sau khi xác định được đoạn tín hiệu chứa hiện tượng transit, chúng tôi tìm một hệ số khuếch đại s sao cho khi nhân đoạn tín hiệu này với $(1 + s)$ thì toàn bộ tín hiệu trở nên "mượt" nhất có thể khi được xấp xỉ bằng một đa thức. Độ "mượt" này được định lượng bằng các hàm sai số dùng trong hồi quy giữa tín hiệu đã điều chỉnh và đường cong hồi quy. Cụ thể, trước tiên ta sẽ dùng hồi quy đa thức để fit phần out-of-transit với một đường đa thức nào đó. Sau đó, dùng phương pháp tối ưu hóa Nelder-Mead để tìm s sao cho hàm lỗi là nhỏ nhất khi nhân phần in-transit với (1+s). Sau nhiều thử nghiệm, chúng tôi nhận thấy rằng sử dụng đa thức bậc ba với hồi quy (degree $= 3$) sẽ cho kết quả ổn định và chính xác nhất.

Để rõ ràng hơn, ta giả sử đoạn tín hiệu trong khoảng `[phase1:phase2]` tương ứng với pha transit có giá trị trung bình là it (in-transit), và phần còn lại có giá trị trung bình là oot (out-of-transit). Khi áp dụng hệ số khuếch đại s vào vùng transit, ta có:

$$\text{Adjusted in-transit signal} = it \cdot (1 + s) \tag{6}$$

Giả định rằng việc điều chỉnh này làm cho mức tín hiệu trong vùng transit tiệm cận với mức tín hiệu ngoài transit, tức là:

$$it \cdot (1 + s) = oot \tag{7}$$

Từ đó, ta suy ra:

$$s = \frac{oot - it}{it} \tag{8}$$

Mặt khác, transit depth (mục tiêu cần dự đoán) được định nghĩa là:

$$\text{transit depth} = \frac{oot - it}{oot} \tag{9}$$

Ta thay s vào biểu thức trên:

$$\text{transit depth} = \frac{s \cdot it}{oot} = \frac{s \cdot it}{it(1+s)} = \frac{s}{1+s} = 1 - \frac{1}{1+s} \tag{10}$$

Vậy:

$$\boxed{\text{transit depth} = 1 - \frac{1}{1+s}} \tag{11}$$

Công thức này cho thấy mối quan hệ chặt chẽ giữa hệ số khuếch đại s và transit depth (mục tiêu cần dự đoán), giúp diễn giải rõ ràng giá trị đầu ra của mô hình non-ML theo cách có ý nghĩa vật lý.

C. Tiếp cận theo hướng sử dụng học máy

Chúng tôi đã thử một số phương pháp học máy khác nhau, gồm: K-Nearest Neighbors (KNN), Gaussian Process Regressor, phương pháp denoise bằng Auto Encoder. Tuy nhiên, kết quả thu được không khả quan, do dữ liệu có quá nhiều nhiễu và không đủ thông tin để mô hình hóa chính xác.

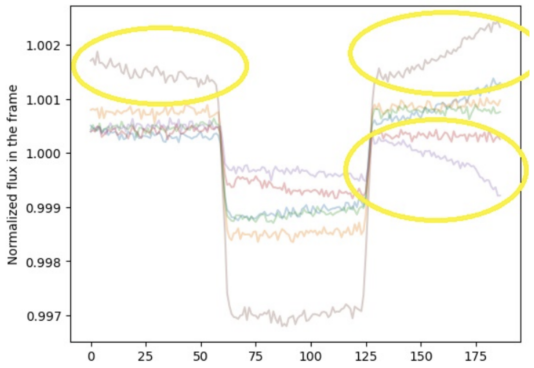
IV. Thực nghiệm

A. Tối ưu các hệ số mô hình non-ML

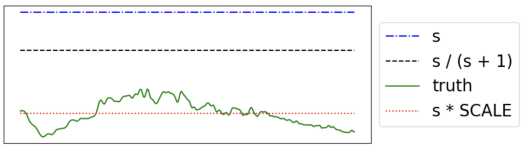
Ý tưởng của phương pháp non-ML đơn giản nhưng để thực nghiệm hiệu quả thì cần phải chọn các tham số phù hợp dựa vào các quan sát trên dữ liệu.

Như đã đề cập ở trên, chúng tôi nhân đoạn tín hiệu transit với $1 + s$ và kỳ vọng đoạn tín hiệu thu được sẽ mượt như khi không có hành tinh nào đi qua sao chủ. Tín hiệu này sẽ được xấp xỉ bằng một đa thức. Việc chọn bậc của đa thức này ảnh hưởng lớn đến độ chính xác của mô hình. Nhìn vào dữ liệu như hình 4, ta thấy rằng sau khi nhân phần lõm ở giữa với $1 + s$ thì tín hiệu không phải lúc nào cũng là 1 đường tuyến tính. Hình dạng phổ biến là bậc 1, 2, 3. Chúng tôi thực nghiệm với các giá trị bậc đa thức khác nhau cho kết quả như bảng IV-A.

Bậc đa thức	1	2	3	4	5	10
Điểm	0.314	0.315	0.322	0.310	0.304	0.115



Hình 4. Hình dạng phổ biến của flux theo thời gian trong tập dữ liệu



Hình 5. Mối quan hệ giữa s, mục tiêu cần dự đoán

Sau khi fit tín hiệu với một đa thức bậc 3, ta tìm s tối ưu (hiệu trị tuyệt đối nhỏ nhất), chúng tôi dự đoán transit depth dựa vào s.

Nhận thấy rằng, nếu lấy giá trị dự đoán theo $\frac{s}{1+s}$ cho ra kết quả thấp hơn so với lấy $s * SCALE$. Hơn nữa, khi ta tính giá trị $SCALE = truth.mean/s$ với mỗi điểm dữ liệu, thì giá trị này sẽ gần bằng nhau cho tất cả các hành tinh. Chạy thực nghiệm cho thấy, sử dụng $SCALE = 0.9396$ cho điểm cao nhất.

Dựa vào công thức tính điểm, để tối ưu điểm không chỉ cần dự đoán chính xác mà còn cần dự đoán độ không chắc chắn phù hợp. Giả định rằng, giá trị dự đoán của mô hình khớp với trung bình của quang phổ cần dự đoán, như vậy

$$\sigma^* = argmax_\sigma \sum_{i=1}^{283} \left(log(\sigma^2) + \frac{(truth[i] - \mu_{truth})^2}{\sigma^2} \right) \tag{12}$$

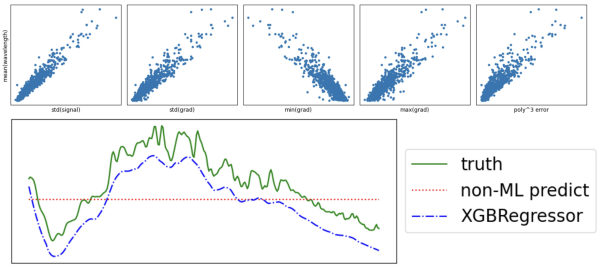
Dễ thấy, hàm mục tiêu là tổng 2 hàm lồi,

$$(\sigma^*)^2 = \sum_{i=1}^{283} (truth[i] - \mu_{truth})^2 = (\sigma_{truth})^2 \tag{13}$$

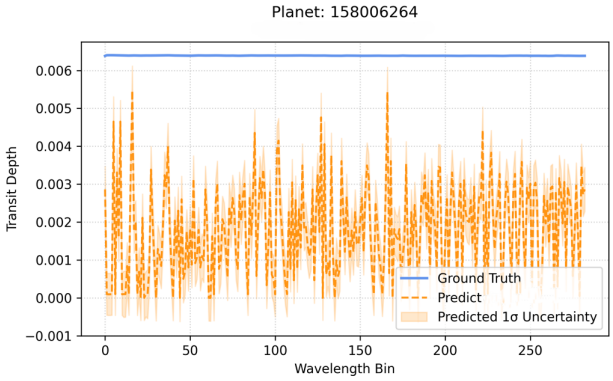
Nhận thấy rằng, giá trị σ tối ưu cho từng hành tinh là khác nhau và có liên quan đến tín hiệu theo trục thời gian. Chúng tôi sử dụng mô hình phụ LinearRegression với đầu vào là tín hiệu, đầu ra là σ_{truth}, và thực hiện huấn luyện mô hình này trên tập dữ liệu huấn luyện. Kết quả tăng 0.004 so với việc sử dụng giá trị σ cố định cho tất cả các hành tinh.

B. Kết quả

Phương pháp non-ML dự đoán quang phổ và LinearRegression dự đoán độ không chắc chắn cho điểm trên tập valid là 0.75, điểm trên tập test là 0.326. Đạt hạng 56/460 đội (tính đến ngày 9/8/2025).



Hình 6. Kết quả của mô hình Regressor so với non ML



Hình 7. Các bước sóng chưa qua làm mịn

Với các phương pháp ML cho dự đoán quang phổ,

Chúng tôi thử với mô hình Regressor, lấy thuộc tính là giá trị flux và một số thuộc tính của gradient của flux có liên quan mạnh đến độ sâu transit như hình 6

với điểm trên tập valid là 0.734, điểm trên tập test là 0.289, nguyên nhân có thể là do sự khác biệt trong phân phối dữ liệu giữa các tập và chọn thuộc tính chưa phù hợp.

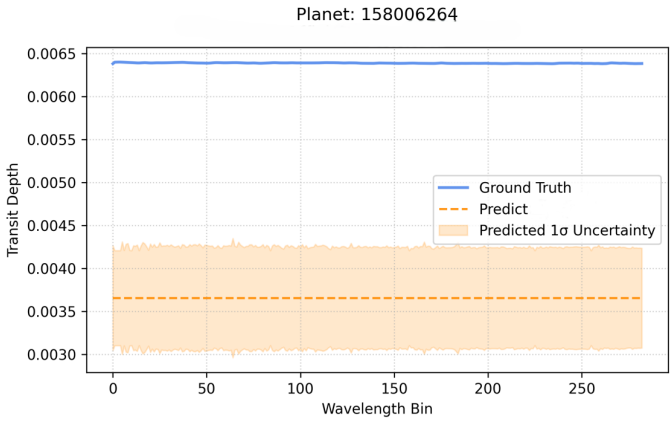
Sự khác biệt về phân phối dữ liệu càng thể hiện rõ hơn khi chúng tôi thử với KNN, với khoảng cách trung bình các điểm dữ liệu trong tập train (với k = 2) là 50. Chúng tôi kết hợp KNN và phương pháp non-ML, với điểm dữ liệu trong tập test có khoảng cách với các điểm trong trong KNN dưới 40, chúng tôi nội suy từ tập train, những điểm lớn hơn thì chúng tôi sử dụng phương pháp non-ML. Kết quả test là 0.322, có hơn phương pháp non-ML nhưng độ chênh lệch quá nhỏ.

Chúng tôi cũng thử làm mịn kết quả dự đoán của 283 bước sóng bằng Gaussian Process (GP) và AutoEncoder (AE).

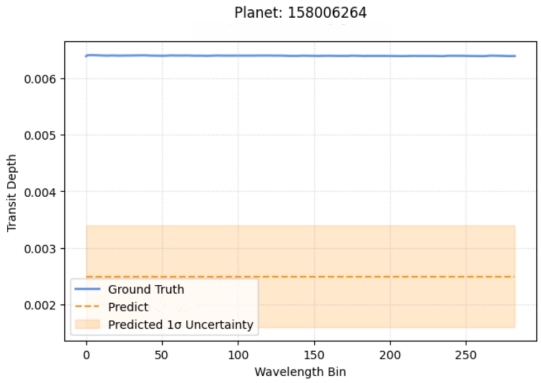
Như trong hình 7, các bước sóng chưa làm mịn có nhiều nhiễu và không đồng nhất. Để cải thiện, chúng tôi đã áp dụng Gaussian Process (GP) và AutoEncoder (AE) để làm mịn, khử nhiễu các bước sóng này. Kết quả là các bước sóng trở nên đồng nhất hơn, như trong hình 8.

Đồng thời, GP cũng sinh ra các ước lượng không chắc chắn cho các bước sóng, cho phép chúng tôi đánh giá độ tin cậy σ của các dự đoán. So với bước sóng được dự đoán bằng phương pháp non-ML , như trong hình 9, các bước sóng này đã sát với ground truth hơn, 0.00356 so với 0.0024.

Một ý tưởng khác để làm mịn bước sóng là sử dụng PCA tìm ra những component chính của các bước sóng này. Kết quả



Hình 8. Các bước sóng đã qua làm mịn

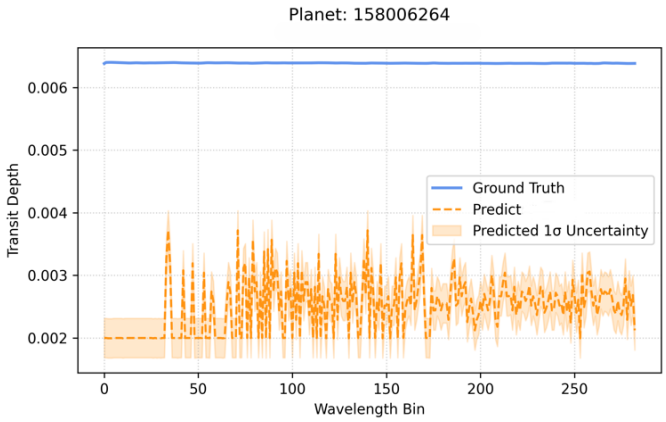


Hình 9. Các bước sóng dự đoán của mô hình non-ML

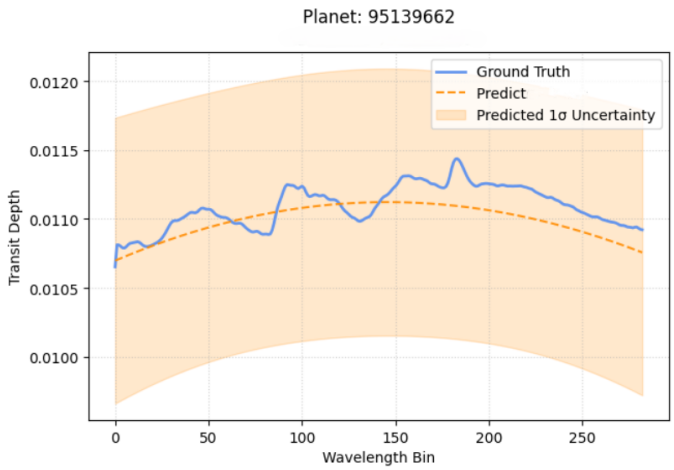
làm mượt bằng PCA cho thấy những tiềm năng của phương pháp này trong việc cải thiện độ chính xác của dự đoán. Bằng việc sử dụng 5 component chính, chúng tôi đã đạt được những cải thiện đáng kể trong việc giảm thiểu nhiễu và tăng cường độ chính xác cho các bước sóng dự đoán.

Dù việc sử dụng PCA đã mang lại những cải thiện đáng kể trên valid, nhưng vẫn còn nhiều thách thức trong việc xử lý dữ liệu nhiễu và không đồng nhất. PCA với 5 component đem lại kết quả thấp trên test set so với valid set có thể do các tập trong test set phân tán hơn, dẫn đến việc mô hình không thể tổng quát tốt hơn cho các bước sóng này.

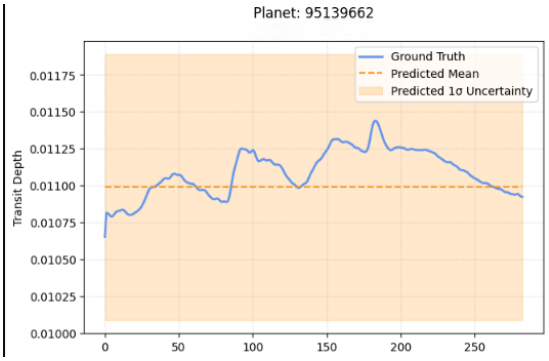
So với sử dụng mean của toàn bộ quang phổ như non-ML, việc dự đoán từng bước sóng cải thiện việc bắt được cấu trúc của các bước sóng quang phổ phức tạp , như hình 11. So với việc dự đoán từng bước sóng, phương pháp non-ML chỉ cho ra được một giá trị trung bình cho toàn bộ quang phổ, như hình 12. Tuy nhiên, việc dự đoán từng bước sóng cũng làm tăng độ nhiễu và độ không đồng nhất của các bước sóng này, dẫn đến việc khó khăn trong việc tổng quát khiến những phương pháp chúng tôi thử nghiệm không đạt được kết quả tốt hơn so với non-ML.



Hình 10. Các bước sóng đã qua làm mượt bằng PCA



Hình 11. Dự đoán của mô hình ML bám sát được hình dạng của ground truth



Hình 12. Dự đoán của mô hình non-ML không bám sát được hình dạng của ground truth

V. Kết luận

Trong bài báo này, chúng tôi đã trình bày phương pháp non-ML để dự đoán quang phổ và linear regression để dự đoán độ không chắc chắn. Phương pháp này sử dụng các kỹ thuật xử lý tín hiệu truyền thống và cho kết quả tốt hơn các phương pháp ML (KNN, GP, AE) chúng tôi đã thử.

Tuy vậy, các phương pháp học máy hiện đang dùng cho kết quả tốt hơn trên tập valid. Do một vài nguyên nhân khiến điểm test chưa tốt. Chúng tôi sẽ tiếp tục nghiên cứu để cải thiện hơn nữa các phương pháp này trong tương lai.

Phụ lục

A. *Tỉ lệ đóng góp*

Thành viên	Phan Bá Thọ	Nguyễn Quốc Huy	Trần Tuấn Anh
Tỉ lệ đóng góp	34%	33%	33%

Bảng I
Tỉ lệ đóng góp của các thành viên trong nhóm