Meta-learning in movement prediction problem of aperiodic time-series data

Bao-Long Nguyen,[1,*] Kenta Hongo,[2,†] Ryo Maezono,[1,‡] and Tom Ichibha[1,§]
[1] *School of Information Science, JAIST, Ishikawa, Japan*
[2] *Research Center for Advanced Computing Infrastructure, JAIST, Ishikawa, Japan*
(Dated: December 21, 2024)

Abstract. Dự đoán dữ liệu chuỗi thời gian phi chu kỳ (ví dụ: giá cổ phiếu, tỷ giá ngoại hối, giá Bitcoin,...) là một tác vụ khó khăn đối với các mô hình học máy bởi vì loại dữ liệu này có phương sai cao và không cố định; không thể hiện chu kỳ rõ ràng nên khó rút trích đặc trưng; không chỉ phụ thuộc vào giá trị quá khứ mà còn phụ thuộc các tác động bên ngoài như tình hình kinh tế, chính trị, khiến chúng bất ổn và không có chu kỳ. Để khắc phục các thách thức nêu trên, chúng tôi sử dụng Meta-learning để huấn luyện kết hợp mạng `LSTM` và `CNN`, từ đó rút trích và tổng hợp hiệu quả các đặc trưng ẩn của dữ liệu theo thời gian. Các thử nghiệm trên dữ liệu phi chu kỳ (dữ liệu ngoại hối của 60 cặp tiền tệ trong 24 năm từ 2000 đến 2024) cho thấy phương pháp đề xuất hoạt động tốt và có độ chính xác cao hơn 5%-11% so với `NHITS` - mô hình SOTA 2023 trên dữ liệu chuỗi thời gian, trong nhiệm vụ dự đoán xu hướng (tăng hoặc giảm) của ngày giao dịch tiếp theo. Trong khi đó, kết quả thu được trên dữ liệu chu kỳ (Nhiệt độ máy biến áp điện, Thời tiết) đạt xấp xỉ `NHITS`.

I. INTRODUCTION

Dự đoán dữ liệu chuỗi thời gian phi chu kỳ nói chung hay dự đoán tỷ giá ngoại hối (foreign exchange, FX) nói riêng từ lâu đã là vấn đề đáng quan tâm của nhiều nghiên cứu [13, 17, 22]. Hai kỹ thuật chính được sử dụng trong aperiodic time-series prediction là fundamental analysis and technical analysis [3]. Trong khi fundamental analysis thiên về phân tích các yếu tố tác động từ bên ngoài và khó có thể capture từ các biến thiên giá trị trong quá khứ như chính sách, chiến lược kinh tế của công ty, quốc gia để dự đoán tương lai; technical analysis dựa hoàn toàn vào lịch sử biến động giá trị để phân tích xu hướng tương lai.

Theo [22], việc dự đoán trên aperiodic time-series data gặp một vài thách thức cố hữu. Trong đó có thể kể tới: (1) - Variance của loại dữ liệu này biến thiên mạnh qua các thời kỳ. Từ đó giả thuyết rằng chúng tuân theo một phân phối để có thể xấp xỉ lỗi là không thể sử dụng được, dẫn đến việc các mô hình học máy khó có thể dự đoán chính xác giá trị cũng như xu hướng trong tương lai; (2) - Aperiodic data không tuân theo bất kỳ quy luật rõ ràng nào nên việc học các đặc trưng ẩn của dữ liệu để tiến hành dự đoán gặp rất nhiều khó khăn; (3) - Aperiodic time-series data (e.g. giá cổ phiếu của một công ty) không hoàn toàn phụ thuộc vào dữ liệu quá khứ mà còn phụ thuộc nhiều yếu tố ngoại lai (e.g. tin tức, tình hình kinh tế, chính trị).

Đối với thách thức đầu tiên, các mô hình ensemble [1, 27, 32] thường được sử dụng để hạn chế ảnh hưởng của sự biến đổi variance. Ensemble model giúp cung cấp một cái nhìn tổng quát, tổng hợp từ nhiều khía cạnh dựa trên các sub-models, từ đó giúp mô hình tổng quát thích ứng được với sự thay đổi mạnh của variance. Tuy nhiên, dưới sự tổng hợp cứng nhắc của các mô hình ensemble truyền thống, kết quả thu được trên các bộ dữ liệu phi chu kỳ là rất thấp.

Để khắc phục thách thức thứ hai, các nghiên cứu phần lớn sử dụng các đặc trưng rút trích được từ Long short-term memory neural network (`LSTM`) [14], Artificial neural network (`ANN`), và Convolution neural network (`CNN`) [21]. Cụ thể, trong năm 2022, 20% tổng số các bài báo liên quan đến dự đoán chỉ số tài chính sử dụng `LSTM`, 20% sử dụng `ANN` và 6% sử dụng `CNN` [3]. Điều này là hoàn toàn dễ hiểu vì `LSTM` và `CNN` đã chứng minh được khả năng của mình trong việc rút trích các đặc trưng không gian, thời gian một cách hiệu quả. Mặc dù vậy, khi đối mặt với tính phi chu kỳ trên 2600 mẫu dữ liệu FX, sau khoảng 100 training steps, `LSTM` vẫn bị underfitting, còn `CNN` bị overfitting chỉ sau 5 training steps

Đối với thách thức thứ ba, nghiên cứu [9] đưa ra giả thuyết rằng các time-series datasets khác nhau của cùng một lĩnh vực tại cùng thời điểm phản ánh sự tác động của các yếu tố ngoại lai. Các nghiên cứu [2, 24] cũng chỉ ra sự phụ thuộc giữa chỉ số tài chính của một công ty nhất định và các chỉ số của các công ty khác. Điều này càng làm tăng tính đúng đắn của giả thuyết trong nghiên cứu [9]. Do đó, bằng việc khai thác các nguồn dữ liệu có cùng domain, chúng ta có thể capture được các thông tin phản ánh tác động bên ngoài.

Chúng tôi tiếp cận các thách thức nêu trên thông qua việc kết hợp đặc trưng và tổng hợp mô hình. Cụ thể, bằng cách sử dụng Meta-learning (ML) [10], chúng tôi tổng hợp hiệu quả tham số của các mô hình cục bộ, giúp giảm thiểu đáng kể variance loss cũng như tăng khả năng thu thập các thông tin phản ánh tác động bên ngoài từ các tập dữ liệu có cùng domain. Để học được các laten patterns, chúng tôi đề xuất phương pháp kết hợp các đặc trưng từ `CNN` và `LSTM`. Ngoài ra, chúng tôi cho rằng, aperiodic time-series data còn có những phụ thuộc ngầm

* mwklng2309@icloud.com
† kenta_hongo@mac.com
‡ rmaezono@mac.com
§ ichibha@icloud.com

vào các thời điểm nhất định trong quá khứ (hidden long-term dependency). Đối với cách tiếp cận truyền thống, người ta sử dụng một lượng dữ liệu quá khứ cố định (lookback window) để huấn luyện mô hình. Điều này gây một trở ngại lớn cho quá trình học vì các đặc trưng dài hạn theo thời gian sẽ bị quên. Mặt khác, ML chia nhỏ tập dữ liệu thành nhiều phần để học và tổng hợp hiệu quả các tham số học được nên có thể xử lý tốt thách thức này.

Cuối cùng, chúng tôi chứng minh tính ưu việt của thuật toán đề xuất bằng cách giải quyết bài toán dự đoán xu hướng (tăng hoặc giảm) của dữ liệu chuỗi thời gian và so sánh kết quả với mô hình SOTA năm 2023 (`NHITS` [4]) trên hai loại dữ liệu: (1) - Dữ liệu phi chu kỳ, bao gồm dữ liệu tỷ giá hối đoái USD/JPY và dữ liệu tỷ giá hối đoái của 60 cặp tiền tệ, bao gồm 18 quốc gia từ năm 2000 đến năm 2024; (2) - Dữ liệu có chu kỳ bao gồm tập dữ liệu Electricity Transformer Temperature [33] và Weather [20]. Các tập dữ liệu này được cung cấp công khai trên Internet và có thể tải xuống dễ dàng. Bản cài đặt chính thức có thể xem tại `https://github.com/baolongnguyenmac/multi_fx`.

Tóm lại, đóng góp chính của chúng tôi như sau:

- Kết hợp đặc trưng: Rút trích đặc trưng bằng cách kết hợp các đặc trưng của `LSTM` và `CNN`.
- Hidden long-term dependency: Chứng minh thực nghiệm rằng dữ liệu time-series tồn tại các phụ thuộc ẩn với chính nó tại nhiều thời điểm khác nhau trong quá khứ.
- Tổng hợp hiệu quả tham số mô hình: Sử dụng ML thay cho các mô hình ensemble truyền thống trong việc tổng hợp kết quả từ các mô hình học máy.
- Experiment: Thực nghiệm trên dữ liệu phi chu kỳ (tỷ giá hối đoái của 60 cặp tiền tệ) và dữ liệu có chu kỳ (Electricity Transformer Temperature, Weather) rồi so sánh với mô hình SOTA 2023 `NHITS` để chứng minh tính hiệu quả của phương pháp đề xuất.

II. RELATED WORKS

A. LSTM & CNN model

Như đã đề cập, `LSTM` là mạng neural rất phổ biến trong việc handle các bài toán liên quan đến dự đoán trên dữ liệu time-series. `LSTM` được sử dụng phổ biến như vậy bởi vì nó xử lý tốt vấn đề vanishing gradient (dễ dàng bắt gặp khi sử dụng Recurrent neural network) và có thể khai thác hiệu quả các mối quan hệ phi tuyến trong dữ liệu. Thật vậy, bằng cách duy trì cell-state trong mỗi iteration, `LSTM` có thể khắc phục vấn đề vanishing gradient, từ đó bảo toàn khả năng capture các phụ thuộc dài hạn [6]. Ngoài ra, `LSTM` thực hiện rút trích đặc trưng với các hàm kích hoạt phi tuyến, giúp các tham số mô hình có thể capture tính phi tuyến của dữ liệu [12]. Hai yếu tố nêu trên khiến cho `LSTM` trở thành lựa chọn đầu tiên được nghĩ đến khi giải quyết các bài toán trên dữ liệu time-series.

Mạng `CNN` được sử dụng rất nhiều trong các tác vụ xử lý hình ảnh [25, 28] bởi khả năng tổng hợp các quan hệ cục bộ. Không chỉ vậy, `CNN` còn được dùng rất nhiều trong các tác vụ xử lý dữ liệu time-series như speech recognition [7], natural language processing [29]. Điều đó chứng tỏ được khả năng của `CNN` trong việc khám phá mối quan hệ thời gian cục bộ giữa các mẫu dữ liệu. Mặc dù vậy, `CNN` lại rất ít được dùng trong các tác vụ dự đoán stock price hay foreign exchange. Trong nghiên cứu này, chúng tôi tận dụng khả năng rút trích đặc trưng cục bộ tuyệt vời của `CNN` để tích hợp thêm thông tin ẩn vào quá trình huấn luyện của mô hình.

B. Optimization-based Meta-learning

Các thuật toán Meta-learning (ML) dựa trên tối ưu, điển hình là MAML [10] được biết đến với khả năng huấn luyện một mô hình có tính tổng quát cao, thích ứng nhanh trên tập dữ liệu mới thông qua một lượng nhỏ dữ liệu và số bước huấn luyện [15, 30]. Với khả năng này, ML được sử dụng rất nhiều trong các tác vụ đòi hỏi khả năng đáp ứng của mô hình trên dữ liệu (e.g. cá nhân hóa mô hình học [5, 8, 26], domain adaptation trong online learning [16, 19]).

Một thuật toán ML cơ bản sẽ được học trên nhiều tác vụ t rút ra từ cùng một phân phối tác vụ $\mathcal{T}$ [15]. Dữ liệu của mỗi tác vụ được chia thành tập support $\mathcal{D}_t^{support}$ (thường có kích thước nhỏ, khoảng 20%) và tập query $\mathcal{D}_t^{query}$. Trong qua trình học, hai bước tối ưu inner và outer optimization được perform đan xen. Inner optimization cố gắng tìm ra một bộ tham số tối ưu θ_t^* cho từng mô hình học máy trên tập support của mỗi tác vụ bằng phương trình 1.

$$\theta_t^* = \theta_t(\phi) = \arg\min_{\theta} \mathcal{L}_t^{task}\left(\phi, \mathcal{D}_t^{support}\right) \tag{1}$$

Trong đó, ϕ là kết quả của quá trình outer optimization, được sử dụng làm tham số khởi tạo cho θ_t. $\mathcal{L}_t^{task}$ là hàm lỗi của mô hình trên tập support của task t.

Sau đó, thuật toán sử dụng các bộ tham số tối ưu θ_t^* để perform trên tập query tương ứng. Lỗi của toàn bộ mô hình sau đó được tổng hợp để thực hiện quá trình outer optimization như phương trình 2.

$$\begin{aligned}\phi^* &= \arg\min_{\phi} \sum_t \mathcal{L}_t^{meta}\left(\theta_t^*, \mathcal{D}_t^{query}\right) \\ &= \arg\min_{\phi} \sum_t \mathcal{L}_t^{meta}\left(\theta_t(\phi), \mathcal{D}_t^{query}\right)\end{aligned} \tag{2}$$

Bằng hình thức huấn luyện trên, mô hình ϕ^* sẽ có mức tổng quát hóa cao trên các tác vụ khác nhau, có thể nhanh chóng đáp ứng một tác vụ mới chỉ sau một vài bước huấn luyện.

Trong inference phase, giá trị khởi tạo cho tham số của mô hình được gán bằng ϕ^*. Mô hình sau đó được huấn luyện nhanh trên tập support sau đó perform trên tập query. Kết quả trên tập query chính là kết quả của mô hình.

Các mô hình hybrid ensemble vốn được sử dụng rất nhiều trong các bài toán xử lý time-series và được chứng minh thực nghiệm là có độ chính xác cao hơn so với các mô hình handle time-series data tiêu chuẩn vì có thể tổng hợp được sức mạnh của nhiều mô hình [3]. Tuy vậy, các hình thức tổng hợp của ensemble model hiện nay vẫn còn rất cứng nhắc vì chỉ có thể tổng hợp dựa trên kết quả cuối (cơ chế voting của bagging models) và kết quả gần cuối (đối với stacking models). Dưới góc nhìn của ensemble model, có thể coi phương trình 2 là một phương pháp tổng hợp hiệu quả các sub-model, giúp tận dụng khả năng rút trích đặc trưng của từng mô hình. Nói cách khác, mô hình sau khi tổng hợp có thể rút trích đặc trưng ở mức sâu hơn, cải thiện đáng kể khả năng dự đoán so với các mô hình ensemble truyền thống.

C. Neural Hierarchical Interpolation for Time Series (NHITS)

`NHITS` được thiết kế để hướng đến việc dự đoán một khoảng dữ liệu time-series tương lai. Cấu trúc của `NHITS` bao gồm nhiều stack liên tiếp nhau. Mỗi stack bao gồm nhiều block nối tiếp nhau. Tại mỗi block, dữ liệu lịch sử được sử dụng để dự đoán dữ liệu tương lai và dữ liệu quá khứ. Cụ thể, tại block l, với L mẫu dữ liệu quá khứ $(\mathbf{y}_{t-L:t,l-1})$, các đặc trưng sẽ được rút trích như sau [4]:

$$\mathbf{y}^{(p)}_{t-L:t,l} = \textbf{Pooling}\left(\mathbf{y}_{t-L:t,l-1}\right) \tag{3}$$

$$\theta^b_l = \textbf{FullyConnected}^b\left(\mathbf{y}^{(p)}_{t-L:t,l}\right) \tag{4}$$

$$\theta^f_l = \textbf{FullyConnected}^f\left(\mathbf{y}^{(p)}_{t-L:t,l}\right) \tag{5}$$

$$\hat{\mathbf{y}}_{t-L:t,l} = g\left(\theta^b_l\right) \tag{6}$$

$$\hat{\mathbf{y}}_{t+1:t+H,l} = g\left(\theta^f_l\right) \tag{7}$$

Trong đó, `Pooling` nén dữ liệu ban đầy vào $\mathbf{y}^{(p)}_{t-L:t,l}$. `FullyConnected` gồm lớp multi-layer perception xếp chồng với hàm kích hoạt phi tuyến. θ^f_l, θ^b_l là các hệ số nội suy forecast và backcast, được dùng để tổng hợp các giá trị đầu ra của block l bằng hàm nội suy $g(\cdot)$. Số chiều của các hệ số nội suy trong mỗi stack được quy định bởi expressiveness ratio r_l: $|\theta_l| = \lceil r_l H \rceil$. Thông thường, expressiveness ratio sẽ rất nhỏ ở các stack đầu tiên và tăng dần về cuối. Theo đó, các stack có thể mô phỏng lại các tần số từ thấp đến cao. Ngoài ra, bằng việc sử dụng hàm nội suy, `NHITS` không cần quá nhiều phần cứng để huấn luyện mạng neural trong trường hợp H lớn.

Đầu ra của block l là giá trị forecast $\hat{\mathbf{y}}_{t+1:t+H,l}$ và giá trị backcast $\hat{\mathbf{y}}_{t-L:t,l}$. Input của block $l+1$ được tính theo phương trình 8.

$$\mathbf{y}_{t-L:t,l+1} = \mathbf{y}_{t-L:t,l-1} - \hat{\mathbf{y}}_{t-L:t,l} \tag{8}$$

Giả sử mô hình gồm có S stacks, mỗi stack có B blocks. Tổng hợp các giá trị forecast của các block như phương trình 9, ta được giá trị forecast của một stack. Backcast của block cuối cùng của một stack chính là đầu vào cho stack tiếp theo. Cuối cùng, tổng hợp giá trị forecast của các stack như phương trình 10, ta được giá trị forecast dự đoán của toàn mạng.

$$\hat{\mathbf{y}}^s_{t+1:t+H} = \sum_{l=1}^{B} \hat{\mathbf{y}}_{t+1:t+H,l} \tag{9}$$

$$\hat{\mathbf{y}}_{t+1:t+H} = \sum_{s=1}^{S} \hat{\mathbf{y}}^s_{t+1:t+H} \tag{10}$$

Bằng cách xếp chồng các stack, stack sau nhận vào phần dư của stack trước, kiến trúc trên được kỳ vọng là sẽ phân rã dữ liệu thành các frequency bands khác nhau (weekly, daily, even hourly). Trên thực tế, `NHITS` perform rất tốt đối với các bộ dữ liệu có tính chu kỳ cao như mức tiêu thụ điện, thời tiết, giao thông. Tuy nhiên, chúng tôi đang hướng đến aperiodic time-series dataset, vốn có tính chu kỳ rất thấp, thậm chí không có (see figure 1). Điều này gây ra khó khăn rất lớn cho `NHITS`.

FIG. 1: Exchange rate (close price) between US dollar and Japanese yen by day (2014-2024).

III. METHODOLOGY

Phương pháp của chúng tôi bao gồm hai phần chính hoạt động song song nhau: (1) - Feature extraction; (2)

- Parameter synthesis. Tổng quan phương pháp được minh họa trong hình 2. Trong phần feature extraction, chúng tôi kết hợp hai loại đặc trưng từ mạng `CNN` và `LSTM`. Trong phần parameter synthesis, chúng tôi sử dụng `MAML` để tổng hợp tham số của các mô hình. Với sự góp mặt của các đặc trưng `LSTM` và `CNN`, chúng tôi kỳ vọng sẽ rút trích được các đặc trưng ẩn trong dữ liệu aperiodic. Bằng việc sử dụng `MAML` trong quá trình tổng hợp trọng số, phương pháp đề xuất được kỳ vọng là một giải pháp thay thế hợp lý và hiệu quả cho các mô hình ensemble truyền thống trong việc giảm thiểu tác động của sự biến thiên variance, tổng hợp hiệu quả các yếu tố ngoại lai, cũng như giữ được các hidden long-term dependency ẩn trong quá khứ.

A. Data preparation

Phương pháp đề xuất sử dụng các thuật toán ML để huấn luyện mô hình. Do đó, dữ liệu cần được tổ chức lại để các thuật toán ML có thể hoạt động được. Trong trường hợp dữ liệu bao gồm nhiều datasets khác nhau thuộc cùng một lĩnh vực, mỗi dataset sẽ được coi là một task của `MAML`. Trong trường hợp dữ liệu bao gồm một dataset duy nhất, cần chia nhỏ dataset này thành các tập con ứng với các task riêng biệt. Tóm lại, tập dữ liệu sau khi chuẩn bị bao gồm n tasks: $\mathcal{D} = \{\mathcal{D}_t\}_{t=1}^{n}$. Dữ liệu tại mỗi task được chia thành tập support và query: $\mathcal{D}_t = \left\{\mathcal{D}_t^{support}, \mathcal{D}_t^{query}\right\}$.

Một sample dữ liệu bao gồm các cặp giá trị $(\mathbf{x}_{t-L:t}, y)$. Trong đó, $\mathbf{x}_{t-L:t}$ bao gồm L giá trị lịch sử tính từ thời điểm t trở về trước; $y \in \{0, 1\}$ là nhãn dữ liệu, thể hiện xu hướng giảm, hoặc tăng của mẫu dữ liệu x_{t+1} so với x_t. Tùy vào từng bài toán và cách cài đặt mà các phần tử trong $\mathbf{x}_{t-L:t}$ có thể là các vector hoặc các scalar number. Ví dụ, đối với dữ liệu chứng khoán, $\mathbf{x}_{t-L:t}$ có thể chứa các vector dữ liệu $\vec{x}_i$ = (open, low, high, close) hoặc chỉ một giá trị close price duy nhất.

B. Feature extraction

Lấy cảm hứng từ nghiên cứu [31], chúng tôi đề xuất kết hợp các đặc trưng rút trích được từ mạng `LSTM` và `CNN`. Cụ thể, chúng tôi đưa từng phần tử trong vector $\mathbf{x}_{t-L:t}$ qua một lớp `FullyConnected` có đầu ra lớn hơn số chiều của $\vec{x}_i, i \in [t-L, t]$ để phân giải thành các đặc trưng nhỏ $\vec{x}'_i$. Các đặc trưng này sau đó được truyền qua mạng `LSTM` và `CNN` để lần lượt rút trích các phụ thuộc thời gian dài hạn ($\mathbf{h}_{LSTM}$) và các đặc trưng thời gian cục bộ ($\mathbf{h}_{CNN}$). Để có thể khai thác tối đa các ràng buộc thời gian dài hạn, chúng tôi sử dụng `BidirectionalLSTM` để rút trích từ hai phía của của $\mathbf{x}_{t-L:t}$. Toàn bộ quy trình rút trích đặc trưng được tóm tắt như sau:

$$\mathbf{x'}_{t-L:t} = \textbf{FullyConnected}\left(\mathbf{x}_{t-L:t}\right) \tag{11}$$
$$\mathbf{h}_{LSTM} = \textbf{BidirectionalLSTM}\left(\mathbf{x'}_{t-L:t}\right) \tag{12}$$
$$\mathbf{h}_{CNN} = \textbf{Convolution1D}\left(\mathbf{x'}_{t-L:t}\right) \tag{13}$$

Mạng `LSTM` duy trì giá trị cell-state nhằm lưu trữ có chọn lọc các phụ thuộc dài hạn. Điều này rất thích hợp trong việc giải quyết các bài toán time-series data. Mặt khác, giá trị tương lai thường phụ thuộc rất lớn vào các giá trị lịch sử gần nhất. Chúng tôi đề xuất sử dụng mạng `CNN` để nhấn mạnh các đặc trưng cục bộ, từ đó hướng một phần sự chú ý của mô hình vào các thời điểm nhất định. Do đó, phương pháp đề xuất không chỉ nhớ được các đặc trưng long-term mà còn highlight được các đặc trưng short-term.

Tiếp đến, $\mathbf{h}_{LSTM}$ và $\mathbf{h}_{CNN}$ được nối với nhau (phương trình 14) sau đó chuyển đến phần phân lớp của mạng NN (phương trình 15).

$$\mathbf{h}_{t-L:t} = \textbf{Concatenate}\left(\mathbf{h}_{LSTM}, \mathbf{h}_{CNN}\right) \tag{14}$$
$$\hat{y} = \textbf{FullyConnected}\left(\mathbf{h}_{t-L:t}\right) \tag{15}$$

C. Effective synthesis of models' parameters

Chúng tôi sử dụng `MAML` để huấn luyện và tổng hợp trọng số của các mô hình tại các task. Như đã đề cập trong phần II, tối ưu tham số theo cách tiếp cận của ML chính là đi giải hai phương trình 1 và 2 bằng các phương pháp tối ưu trên dữ liệu support và query. Cụ thể, quá trình tối ưu bao gồm nhiều bước toàn cục (outer optimization), thực hiện trên tất cả các tasks tham gia huấn luyện. Mỗi bước toàn cục bao gồm nhiều bước cục bộ (inner optimization) thực hiện trên từng task riêng lẻ. Tại bước toàn cục r, quá trình tối ưu cục bộ lần thứ e tại tập support của task t diễn ra như sau:

$$\begin{cases} \theta_t^{(0)} & = \phi_{r-1} \\ \theta_t^{(e)} & = \theta_t^{(e-1)} - \alpha \nabla_\theta \mathcal{L}_t^{task}\left(\theta_t^{(e-1)}, \mathcal{D}_t^{support}\right) \end{cases} \tag{16}$$

In which, ϕ_{r-1} is the result of the $r-1$ outer optimization process, α is the inner learning rate.

Tiếp đó, quá trình outer optimization tại bước toàn cục được thực hiện bằng cách tổng hợp độ lỗi trên tập query của các task và tối ưu trên đó (phương trình 17).

$$\begin{cases} \phi_0 = \text{Random Initialization} \\ \phi_r = \phi_{r-1} - \beta \nabla_\phi \sum_{t=1}^{n} \mathcal{L}_t^{meta}\left(\theta_t^*(\phi), \mathcal{D}_t^{query}\right) \end{cases} \tag{17}$$

Where, β is the outer learning rate.

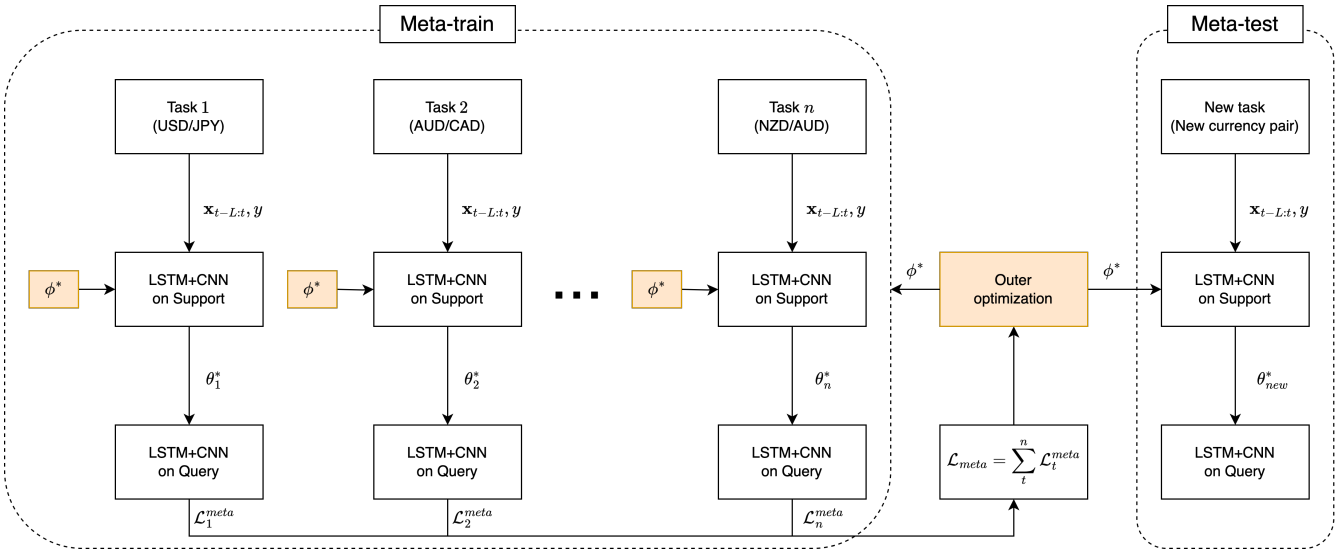

FIG. 2: The full-flow of meta-training and meta-testing on multi-fx data. Each currency pair is regarded as a task.

Giả sử thuật toán chạy E steps trong inner optimization, lượng đạo hàm tại phương trình 17 được viết lại như sau (the notations of dataset are removed):

$$\begin{aligned}
&\beta \nabla_\phi \sum_{t=1}^{n} \mathcal{L}_t^{meta} \left(\theta_t - \alpha \nabla_\theta \mathcal{L}_t^{task}\left(\theta_t\right)\right) \\
=&\beta \sum_{t=1}^{n} \frac{\partial \mathcal{L}_t^{meta}\left(\theta_t^{(E)}\right)}{\partial \theta_t^{(E)}} \frac{\partial \theta_t^{(E)}}{\partial \phi} \\
=&\beta \sum_{t=1}^{n} \nabla_\theta \mathcal{L}_t^{meta}\left(\theta_t^{(E)}\right) \prod_{j=0}^{E-1} \left[\mathbb{I} - \alpha \nabla_\theta^2 \mathcal{L}_t^{task}\left(\theta_t^{(j)}\right)\right]
\end{aligned} \tag{18}$$

Sự xuất hiện của tích các đạo hàm bậc hai trong phương trình 18 khiến quá trình đạo hàm trở nên phức tạp vì phải tốn rất nhiều chi phí để duy trì các ma trận Hessian. Do đó, số bước tính toán để tìm ra θ^* cần phải hạn chế. Trên thực tế, các phương pháp sử dụng ML [5, 8, 10, 23, 26] thường sẽ chọn $E \in [1, 5]$.

IV. NUMERICAL EXPERIMENT

A. Dataset & Metric

FX nói riêng và các chỉ số tài chính nói chung là dạng dữ liệu điển hình cho aperiodic time-series data. Do đó, chúng tôi chọn loại dữ liệu này để kiểm thử mô hình. Cụ thể, chúng tôi cấu hình hai bộ dữ liệu sử dụng dữ liệu FX. Bộ dữ liệu `USD/JPY` chỉ gồm dữ liệu của cặp tiền tệ USD/JPY, được chia thành 60 tập dữ liệu con theo trình tự thời gian với kích thước bằng nhau. Dữ liệu được sample theo giờ từ năm 2000 đến năm 2024, bao gồm các thuộc tính open, low, high, và close price. Bộ dữ liệu `multi-fx` bao gồm 60 cặp tiền tệ giữa 18 quốc gia Australia, Canada, Switzerland, Denmark, EU, United Kingdom, Hong Kong, Iceland, Japan, Norway, New Zealand, Singapore, Sweden, Turkey, United States, Mexico, China, South Africa. Dữ liệu có thuộc tính tương tự như `USD/JPY` và được sample theo ngày từ năm 2014 đến 2024.

Bộ dữ liệu `multi-fx` được sử dụng để rút trích và tổng hợp thông tin về các yếu tố ngoại lai (i.e. thông tin thị trường, kinh tế, chính trị,...), vốn được cho là có ảnh hưởng đến kết quả của một chỉ số tài chính nhất định [2, 9, 24]. Bộ dữ liệu `USD/JPY` được sử dụng để kiểm chứng giả thuyết của chúng tôi về việc dữ liệu tương lai ngầm phụ thuộc vào các thời điểm nhất định trong quá khứ và cần phải tổng hợp đặc trưng quá khứ một cách hiệu quả để làm rõ được các phụ thuộc này.

Ngoài ra, chúng tôi sử dụng thêm 2 bộ dữ liệu: Electricity Transformer Temperature (`ETT-m2`) và Weather (`WTH`). Tập dữ liệu `ETT-m2` bao gồm 7 trường dữ liệu, đo đạc các thông số của máy biến áp tại một tỉnh của Trung Quốc sau mỗi 15 phút trong khoảng thời gian từ July 2016 to July 2018. Tập dữ liệu `WTH` bao gồm 12 trường dữ liệu, ghi nhận các thông số của thời tiết tại Weather Station of the Max Planck Biogeochemistry Institute in Jena, Germany sau mỗi 10 phút trong năm 2020. Đây là hai tập dữ liệu thể hiện tính chu kỳ rất mạnh mà `NHITS` đã rất thành công trong việc dự đoán. Thực nghiệm trên chúng giúp so sánh toàn diện hơn về khả năng của phương pháp đề xuất đối với `NHITS`.

Nghiên cứu sử dụng macro accuracy, macro precision, macro recall, và macro F1 để đánh giá các mô hình. Theo đó, trong quá trình inference, mô hình sẽ chạy trên từng task để tính metrics của mỗi task. Sau đó tính trung bình cộng các metrics của các task để thu được kết quả cuối.

TABLE I: Statistics on datasets.

Dataset	Attribute	Task	Sample	Sample/Task
USD/JPY	4	60	150,175	2,503
multi-fx	4	60	154,440	2,575
ETT-m2	7	48	69,680	1,452
WTH	12	40	35,064	877

B. Experiment

Chúng tôi so sánh phương pháp đề xuất với NHITS bằng cách sử dụng các số liệu và tập dữ liệu trên. Đối với mỗi tập dữ liệu, mỗi trường dữ liệu sẽ được dự đoán xu hướng tương lai dựa trên dữ liệu trong quá khứ. Kết quả cuối cùng được tính bằng cách lấy trung bình kết quả trên các trường dữ liệu. Chúng tôi cũng thử nghiệm các bộ trích xuất tính năng khác nhau (LSTM, CNN) cho phương pháp đề xuất để chứng minh khả năng nắm bắt dữ liệu của LSTM+CNN.

Đối với mô hình NHITS, dữ liệu được cấu trúc thành các tập huấn luyện, tập xác thực và tập kiểm tra với tỷ lệ 6:2:2. Đối với thuật toán đề xuất, vì quy trình meta-training và meta-testing yêu cầu chia dữ liệu thành các tác vụ nhỏ và cho phép mô hình thích ứng trên tập hỗ trợ của từng tác vụ, chúng tôi chia dữ liệu thành 40-60 tác vụ (xem bảng I để biết chi tiết thống kê). Trong mỗi tác vụ, tập support chiếm 20% với mục đích cho phép mô hình thích ứng với dữ liệu, tập query chiếm 80% để kiểm tra tính tương thích của mô hình. Chúng tôi sử dụng 50% tác vụ cho meta-training, 25% cho meta-validating và 25% cho meta-testing. Với cách chia này, chúng tôi đảm bảo tính công bằng rằng mô hình ML được huấn luyện với cùng lượng dữ liệu như NHITS (xem phần phụ lục A). Chi tiết về các thí nghiệm có thể được tìm thấy ở phần phụ lục B và C.

V. RESULT & DISCUSSION

Bảng II so sánh kết quả giữa mô hình đề xuất và mô hình NHITS trên các tập dữ liệu kể trên. Trên dữ liệu phi chu kỳ (multi-fx and USD/JPY), phương pháp của chúng tôi đạt mức hội tụ cao hơn hẳn so với mô hình NHITS thể hiện trên tất cả các metrics. Cụ thể, độ chính xác cải thiện từ 5% đến 11% so với NHITS trên dữ liệu USD/JPY và multi-fx. Các metrics còn lại cũng đều tăng từ 5% đến 20%. Trên dữ liệu có chu kỳ, phương pháp của chúng tôi ghi nhận các kết quả cao hơn hoặc xấp xỉ kết quả của NHITS. Cụ thể, trên tập WTH, phương pháp của chúng tôi cho giá trị cao hơn khoảng 1% trên tất cả các độ đo, trên tập ETT-m2, chúng tôi thu được độ chính xác cao hơn khoảng 1% trong khi các metrics khác thấp hơn khoảng 3%. Sau đây, chúng tôi phân tích kết quả này dựa trên hai yếu tố: (1) - Khả năng rút trích đặc trưng; (2) - Khả năng tổng hợp mô hình.

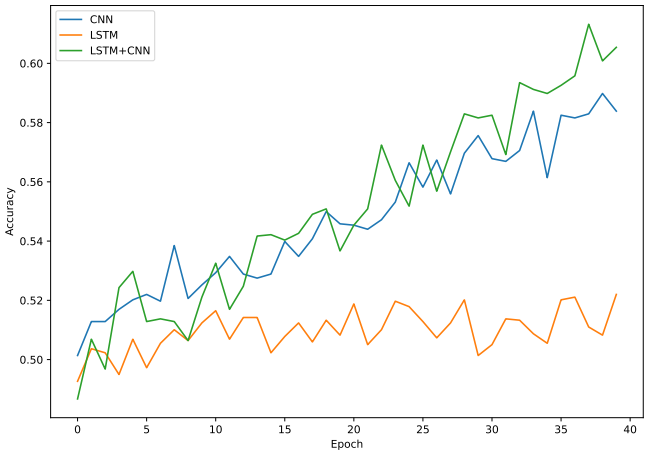

FIG. 3: Training process of LSTM, CNN, and LSTM+CNN using 2600 samples from USD/JPY dataset.

A. Feature extraction ability

Chúng tôi huấn luyện ba mô hình CNN, LSTM, LSTM+CNN trên 2600 mẫu dữ liệu CHF/NZD theo cách thông thường. Sau khi quan sát quá trình huấn luyện của chúng (hình 3), chúng tôi nhận thấy khả năng rút đặc trưng cục bộ tuyệt vời trên dữ liệu ngắn hạn của CNN. Mặt khác, LSTM không cải thiện sau 40 epochs, chứng tỏ rằng nếu chỉ tập trung vào khai thác các phụ thuộc ngắn/dài hạn mà không rút trích đặc trưng đủ sâu thì sẽ không mang lại kết quả cao. Khi kết hợp cả hai đặc trưng, mô hình cho thấy sự cải thiện trong khả năng học trên tập train. Mức hội tụ nhìn chung đều tăng so với việc sử dụng từng mô hình riêng lẻ. Điều này có thể được lý giải thông qua việc LSTM đã tích hợp thành công các long-term feature vào mô hình, cũng như việc tận dụng các đặc trưng cục bộ của CNN.

Tuy nhiên, khi quan sát kết quả trên tập validation, cả ba mô hình trên không đều cho kết quả rất tệ. Lý giải cho điều này, chúng tôi nhận định rằng dù có thể học tốt trên tập train nhưng các mô hình đều mất đi tính tổng quát hóa vì đây là loại dữ liệu phức tạp, khác với các dữ liệu như hình ảnh, video, hoặc dữ liệu có chu kỳ. Nếu khả năng thích ứng của mô hình không cao, kết quả tệ trên tập validation là điều có thể lường trước được. Mặc dù vậy, **đồ thị trong hình 3 cho thấy chúng ta không thể phủ nhận khả năng rút trích đặc trưng của** LSTM+CNN.

Quá trình rút trích và tổng hợp đặc trưng của NHITS thể hiện rằng phương pháp này đang cố gắng mô phỏng lại quá trình phân giải tần số của Fourier transform. Đặc trưng về các giải tần là thông tin tối quan trọng đối với dữ liệu có chu kỳ. Do đó, các dự đoán của NHITS có thể dễ dàng đạt được độ chính xác cao trên loại dữ liệu này. Tuy nhiên, rất khó để có thể phân giải tần số trên dữ liệu phi chu kỳ. Hơn nữa, không khó để nhận ra các kỹ

TABLE II: Classification results (%) of NHITS and our method. Best results per metrics are boldfaced.

		accuracy	*precision*	*recall*	$F1-score$
USD/JPY	NHITS	58.09	58.32	57.53	56.53
	Ours(LSTM)	69.45 ± 1.2	69.87 ± 1.04	69.49 ± 0.59	68.89 ± 1.43
	Ours(LSTM+CNN)	$\mathbf{69.67 \pm 1.07}$	$\mathbf{70.12 \pm 1.02}$	$\mathbf{69.74 \pm 0.33}$	$\mathbf{69.08 \pm 0.34}$
multi-fx	NHITS	52.63	52.62	52.59	52.49
	Ours(LSTM)	63.63 ± 2.79	64.58 ± 2.65	63.87 ± 1.93	62.33 ± 5.07
	Ours(LSTM+CNN)	$\mathbf{63.78 \pm 2.4}$	$\mathbf{64.84 \pm 0.76}$	$\mathbf{64.06 \pm 1.67}$	$\mathbf{62.37 \pm 3.89}$
ETT-m2	NHITS	71.72	**67.22**	**63.69**	**63.28**
	Ours(LSTM)	71.41 ± 6.36	63.48 ± 2.41	60.57 ± 3.16	59.77 ± 3.42
	Ours(LSTM+CNN)	$\mathbf{72.09 \pm 5.37}$	64.12 ± 2.66	61.13 ± 2.78	60.45 ± 3.19
WTH	NHITS	74.17	68.18	66.77	67.08
	Ours(LSTM)	74.65 ± 2.31	68.76 ± 2.38	65.68 ± 1.8	65.84 ± 2.03
	Ours(LSTM+CNN)	$\mathbf{75.45 \pm 2.07}$	$\mathbf{69.64 \pm 1.96}$	$\mathbf{67.18 \pm 1.49}$	$\mathbf{67.09 \pm 1.82}$

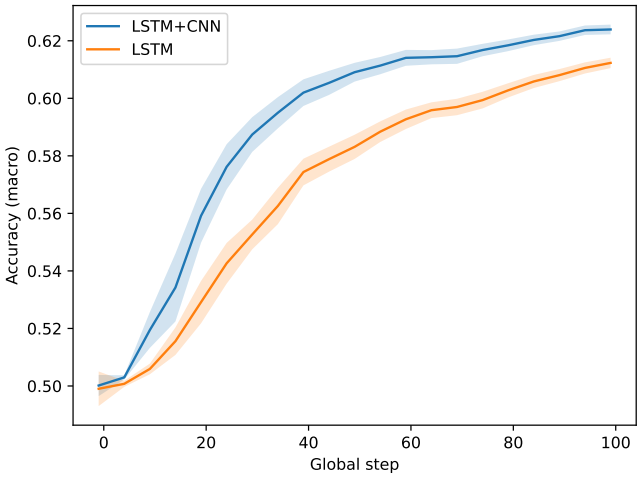

FIG. 4: Convergence process on validation set of ML model using LSTM+CNN feature and LSTM feature only (predict close price of multi-fx dataset). The blur domains cover 99.73% ($\pm 3\sigma$) values of accuracies.

thuật rút trích đặc trưng của NHITS là vô cùng đơn giản (chỉ gồm các lớp FullyConnected và Pooling). Chính những điều này đã khiến cho NHITS gặp trở ngại lớn trên multi-fx, USD/JPY datasets và các dữ liệu phi chu kỳ nói chung. Mặt khác, phương pháp đề xuất của chúng tôi sử dụng các mạng deep learning để rút trích đặc trưng nên có thể dễ dàng thu được các deep hidden feature. Do đó, trên cả periodic data lẫn aperiodic data, phương pháp của chúng tôi đều đạt được kết quả xấp xỉ hoặc cao hơn NHITS.

B. Sub-model synthesis ability

Lý do phương pháp huấn luyện thông thường không đạt hiệu quả cao mặc cho khả năng rút trích đặc trưng đã được cải thiện là vì thiếu khả năng sử dụng đặc trưng một cách hiệu quả. Khả năng này trong các nghiên cứu trước đây thường được cải thiện bằng việc sử dụng các mô hình ensemble. Chúng tôi đã thử sử dụng các mô hình truyền thống (boosting, stacking) nhằm tăng độ chính xác của mô hình nhưng sớm nhận ra rằng phương pháp này không khả thi. Ngay cả khi chúng tôi vét cạn cạn trên không gian hyper-parameter để tìm ra kiến trúc mô hình con tốt nhất bằng AutoKeras [18], độ chính xác thu được trên tập USD/JPY chỉ ở mức 52.53% và 53.71%, respectively.

Khi sử dụng MAML để tổng hợp đặc trưng của mô hình LSTM và LSTM+CNN, khả năng tương thích (thể hiện qua việc hội tụ sớm) cũng như kết quả (thể hiện qua độ chính xác) của mô hình sử dụng LSTM+CNN cao hơn hoàn toàn so với mô hình chỉ sử dụng LSTM (xem hình 4). Thật vậy, chỉ sau 40 epochs, các đặc trưng LSTM+CNN cho mức hội tụ trên 60% và sau 100 epochs, mô hình hội tụ ở mức 62%, vượt trên những gì các đặc trưng LSTM có thể làm được. Độ biến thiên của các giá trị accuracy là rất nhỏ, cho thấy sự ổn định của MAML. Đối với mô hình ML chỉ sử dụng CNN, độ chính xác cũng như giá trị lỗi thậm chí còn không cải thiện trong toàn bộ quá trình huấn luyện. Giải thích cho việc này, chúng tôi cho rằng các đặc trưng của CNN đã quá tập trung vào tính cục bộ của dữ liệu mà bỏ qua các phụ thuộc dài hạn, vốn là một đặc trưng quan trọng của dữ liệu time-series. Trái lại, đặc trưng của LSTM+CNN không chỉ nắm bắt được các phụ thuộc dài hạn mà còn có thể highlight các tính chất cục bộ trong dữ liệu, khiến cho nó hoàn toàn outperform CNN cũng như NHITS. Do đó, **chúng tôi kết luận rằng ML tỏ ra linh hoạt và đạt hiệu quả cao trong việc tổng hợp các sub-model so với các mô hình ensemble cứng nhắc truyền thống**.

Khi tiếp cận dữ liệu USD/JPY bằng ML, mô hình cũng cho độ chính xác cao. Bằng việc nhìn vào các khoảng thời gian trong quá khứ, **phương pháp đề xuất đã tỏ ra hiệu quả hơn trong việc nắm bắt các hidden long-**

term dependency thông qua việc tổng hợp hiệu quả các đặc trưng mà `LSTM` **dễ dàng quên đi trong quá trình huấn luyện**. Điều này đã chứng minh giả thuyết về sự tồn tại của các hidden long-term dependency mà chúng tôi nêu ra ở phần I.

Trong quá trình tổng hợp các đặc trưng để đưa ra dự đoán cuối, `NHITS` chỉ sử dụng các phép cộng trừ đơn giản. Điều này đặt trong ngữ cảnh của việc tổng hợp thông tin dựa trên các hàm nội suy và phần dư của input là hợp lý. Tuy nhiên, đối diện với sự phức tạp trong dữ liệu đòi hỏi một cấu trúc phức tạp hơn để có thể tổng hợp các đặc trưng một cách hợp lý hơn. Việc sử dụng nội suy trên dữ liệu có thể coi là rất đơn giản so với quá trình meta-optimization của các thuật toán ML. Hơn nữa, việc phân tách và kết hợp các dải tần số dữ liệu của `NHITS` chỉ có thể hoạt động tốt nếu dữ liệu thực sự tồn tại chu kỳ. Dữ liệu phi chi kỳ như FX hoặc stock price có thể là điểm yếu chí mạng cho hướng tiếp cận này.

VI. CONCLUSION & FUTURE DIRECTION

Các bài toán trên aperiodic time-series data đặt ra một thử thách lớn cho các mô hình học máy hiện nay bởi tính bất định trong variance cũng như tính phi chu kỳ loại dữ liệu này. Bằng việc kết hợp các đặc trưng cục bộ và các phụ thuộc dài hạn cũng như khai thác các phụ thuộc ẩn trong từng khoảng thời gian quá khứ, chúng tôi đã cải khả năng của mô hình học máy trên các classification metrics trong bài toán dự đoán xu hướng giá của ngày tiếp theo trên dữ liệu FX. Thuật toán đề xuất đã chứng minh được khả năng vượt trội của mình khi so sánh với mô hình `NHITS`.

In the future, we propose two main directions of development related to the architecture and personalization of the learning model.

Model architecture. Phương pháp của chúng tôi được phát triển dưới dạng module với hai module chính hoạt động song song: Module rút trích đặc trưng và Module tổng hợp mô hình. Điều này cung chấp cho phương pháp của chúng tôi một khả năng nâng cấp linh hoạt. Thí nghiệm của chúng tôi chỉ minh họa một trường hợp điển hình trong việc rút trích và tổng hợp hiệu quả các đặc trưng. Bằng việc thay thế các mô hình rút trích đặc trưng khác nhau và sử dụng các thuật toán ML khác nhau, hoàn toàn có thể tạo ra mô hình mới với độ chính xác cao hơn.

Long-horizon problem. Hoàn toàn có thể mở rộng phương pháp này để giải các bài toán về long-horizon prediction. Thật vậy, bằng việc thay đổi đầu ra và độ lỗi của mô hình, có thể tiến hành giải các bài toán này. Tuy nhiên, kiến trúc của các sub-model cần được nghiên cứu lại để phù hợp hơn với bài toán mới.

VII. ACKNOWLEDGEMENTS

VIII. AUTHOR CONTRIBUTIONS

All authors contributed to conceiving the idea. Bao-Long Nguyen, Tom Ichibha performed calculations. All authors contributed to the discussion and writing of the paper.

IX. DATA AVAILABILITY STATEMENT

The datasets used and/or analyzed during the current study available from the corresponding authors on reasonable request.

[1] Ali, M., Prasad, R., Xiang, Y., Yaseen, Z.M.: Complete ensemble empirical mode decomposition hybridized with random forest and kernel ridge regression model for monthly rainfall forecasts. Journal of Hydrology **584**, 124647 (2020)

[2] Andraw W. Lo, A.C.M.: When are contrarian profits due to stock market overreaction? The review of financial studies **3**(2), 175–205 (1990)

[3] Ayitey Junior, M., Appiahene, P., Appiah, O., Bombie, C.N.: Forex market forecasting using machine learning: Systematic literature review and meta-analysis. Journal of Big Data **10**(1), 9 (2023)

[4] Challu, C., Olivares, K.G., Oreshkin, B.N., Ramirez, F.G., Canseco, M.M., Dubrawski, A.: Nhits: Neural hierarchical interpolation for time series forecasting. In: Proceedings of the AAAI conference on artificial intelligence. vol. 37, pp. 6989–6997 (2023)

[5] Chen, F., Luo, M., Dong, Z., Li, Z., He, X.: Federated meta-learning with fast convergence and efficient communication. arXiv preprint arXiv:1802.07876 (2018)

[6] Cheng, L.C., Huang, Y.H., Wu, M.E.: Applied attention-based lstm neural networks in stock prediction. In: 2018 IEEE International Conference on Big Data (Big Data). pp. 4716–4718. IEEE (2018)

[7] Dua, S., Kumar, S.S., Albagory, Y., Ramalingam, R., Dumka, A., Singh, R., Rashid, M., Gehlot, A., Alshamrani, S.S., AlGhamdi, A.S.: Developing a speech recognition system for recognizing tonal speech signals using a convolutional neural network. Applied Sciences **12**(12), 6223 (2022)

[8] Fallah, A., Mokhtari, A., Ozdaglar, A.: Personalized federated learning: A meta-learning approach. arXiv preprint arXiv:2002.07948 (2020)

[9] Fama, E.F.: Efficient capital markets: A review of theory and empirical work. Journal of finance **25**(2), 383–417 (1970)

[10] Finn, C., Abbeel, P., Levine, S.: Model-agnostic meta-learning for fast adaptation of deep networks. In: International conference on machine learning. pp. 1126–1135. PMLR (2017)

[11] Garza, A.: Neural forecast - user friendly state-of-the-art neural forecasting models, https://github.com/Nixtla/neuralforecast

[12] He, T., Droppo, J.: Exploiting lstm structure in deep neural networks for speech recognition. In: 2016 IEEE international conference on acoustics, speech and signal processing (ICASSP). pp. 5445–5449. IEEE (2016)

[13] Heryadi, Y., Wibowo, A., et al.: Foreign exchange prediction using machine learning approach: A pilot study. In: 2021 4th International Conference on Information and Communications Technology (ICOIACT). pp. 239–242. IEEE (2021)

[14] Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation **9**(8), 1735–1780 (1997)

[15] Hospedales, T., Antoniou, A., Micaelli, P., Storkey, A.: Meta-learning in neural networks: A survey. IEEE transactions on pattern analysis and machine intelligence **44**(9), 5149–5169 (2021)

[16] Hu, N., Mitchell, E., Manning, C.D., Finn, C.: Meta-learning online adaptation of language models. arXiv preprint arXiv:2305.15076 (2023)

[17] Islam, M.S., Hossain, E.: Foreign exchange currency rate prediction using a gru-lstm hybrid network. Soft Computing Letters **3**, 100009 (2021)

[18] Jin, H., Chollet, F., Song, Q., Hu, X.: Autokeras: An automl library for deep learning. Journal of machine Learning research **24**(6), 1–6 (2023)

[19] Khoee, A.G., Yu, Y., Feldt, R.: Domain generalization through meta-learning: A survey. arXiv preprint arXiv:2404.02785 (2024)

[20] Kolle, O.: Weather dataset (2008), https://www.bgc-jena.mpg.de/wetter/

[21] LeCun, Y., Boser, B., Denker, J., Henderson, D., Howard, R., Hubbard, W., Jackel, L.: Handwritten digit recognition with a back-propagation network. Advances in neural information processing systems **2** (1989)

[22] Li, C., Song, D., Tao, D.: Multi-task recurrent neural networks and higher-order markov random fields for stock price movement prediction: Multi-task rnn and higer-order mrfs for stock price classification. In: Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining. pp. 1141–1151 (2019)

[23] Li, Z., Zhou, F., Chen, F., Li, H.: Meta-sgd: Learning to learn quickly for few-shot learning. arXiv preprint arXiv:1707.09835 (2017)

[24] Mech, T.S.: Portfolio return autocorrelation. Journal of Financial Economics **34**(3), 307–344 (1993)

[25] Naranjo-Torres, J., Mora, M., Hernández-García, R., Barrientos, R.J., Fredes, C., Valenzuela, A.: A review of convolutional neural network applied to fruit image processing. Applied Sciences **10**(10), 3443 (2020)

[26] Nguyen, B.L., Cao, T.C., Le, B.: Meta-learning and personalization layer in federated learning. In: Asian Conference on Intelligent Information and Database Systems. pp. 209–221. Springer (2022)

[27] Sadeghi, A., Daneshvar, A., Zaj, M.M.: Combined ensemble multi-class svm and fuzzy nsga-ii for trend forecasting and trading in forex markets. Expert Systems with Applications **185**, 115566 (2021)

[28] Sharma, N., Jain, V., Mishra, A.: An analysis of convolutional neural networks for image classification. Procedia computer science **132**, 377–384 (2018)

[29] Varshitha, K.S., Kumari, C.G., Hasvitha, M., Fiza, S., Amarendra, K., Rachapudi, V.: Natural language processing using convolutional neural network. In: 2023 7th International Conference on Computing Methodologies and Communication (ICCMC). pp. 362–367. IEEE (2023)

[30] Vettoruzzo, A., Bouguelia, M.R., Vanschoren, J., Rognvaldsson, T., Santosh, K.: Advances and challenges in meta-learning: A technical review. IEEE Transactions on Pattern Analysis and Machine Intelligence (2024)

[31] Vo, Q.H., Nguyen, H.T., Le, B., Nguyen, M.L.: Multi-channel lstm-cnn model for vietnamese sentiment analysis. In: 2017 9th international conference on knowledge and systems engineering (KSE). pp. 24–29. IEEE (2017)

[32] Zafeiriou, T., Kalles, D.: Intraday ultra-short-term forecasting of foreign exchange rates using an ensemble of neural networks based on conventional technical indicators. In: 11th Hellenic Conference on Artificial Intelligence. pp. 224–231 (2020)

[33] Zhou, H., Zhang, S., Peng, J., Zhang, S., Li, J., Xiong, H., Zhang, W.: Informer: Beyond efficient transformer for long sequence time-series forecasting. In: Proceedings of the AAAI conference on artificial intelligence. vol. 35, pp. 11106–11115 (2021)

Appendix A: Fairness in data

Với NHITS, chúng tôi sử dụng các chia dữ liệu truyền thống. Tập dữ liệu ban đầu được chia thành training set, validation set, và testing set theo tỷ lệ 6:2:2.

Đối với phương pháp đề xuất, chúng tôi chia dữ liệu thành các tập dữ liệu nhỏ, mỗi tập dữ liệu lại được chia thành tập support và tập query. Gọi số lượng task sau khi chia là n, chúng tôi sử dụng $0.5n$ tasks cho meta-training, $0.25n$ tasks cho meta-validating, và $0.25n$ tasks cho meta-testing. Trong mỗi task, tập support chiếm 20%, tập query chiếm 80%.

Gọi M là số mẫu dữ liệu trong tập dữ liệu ban đầu, số mẫu dữ liệu sử dụng để train, validate, và test cho NHITS lần lượt là $0.6M, 0.2M, 0.2M$. Đối chiếu với lượng dữ liệu sử dụng cho phương pháp đề xuất, mỗi task sẽ bao gồm $\frac{M}{n}$ mẫu dữ liệu. Chúng tôi sử dụng $0.5M$ cho meta-training (vì số task cho meta-training là $0.5n$). Tiếp đến, trong quá trình validate hoặc test, chúng tôi sử dụng 20% dữ liệu mỗi task cho quá trình model adaptation và 80% dữ liệu còn lại cho validate hoặc test. Theo đó, tổng số mẫu dùng cho quá trình huấn luyện và adaptation là $0.5M + 0.25n \times 0.2\frac{M}{n} + 0.25n \times 0.2\frac{M}{n} = 0.6M$. Số mẫu dùng để đánh giá mô hình trong quá trình validation và testing đều là $0.25n \times 0.8\frac{M}{n} = 0.2M$. Như vậy, chúng tôi đã bảo đảm sự công bằng trong huấn luyện và kiểm thử cho các thí nghiệm.

Appendix B: Experimental detail for proposed method

Trong cài đặt của chúng tôi, chúng tôi sử dụng một lớp FullyConnected gồm 16 units với hàm kích hoạt ReLU để phân giải đặc trưng ban đầu. Sau đó, đặc trưng này được truyền song song đến các hai khối BidirectionalLSTM và CNN. Khối BidirectionalLSTM bao gồm 32 hidden units, các đầu ra được nối dài để tạo thành một vector cuối. Khối CNN bao gồm hai layer CNN có số filter lần lượt là 32 và 64. Kernel được sử dụng trong các layer có kích thước 3×3. Theo sau mỗi layer CNN là một layer MaxPooling sử dụng kernel kích thước 2×2. Kết thúc khối CNN là một layer Flatten. Đặc trưng của khối BidirectionalLSTM và CNN sau đó được nối lại rồi truyền qua một layer phân lớp nhị phân với hàm kích hoạt Sigmoid.

Quá trình fine-tune của thuật toán ML liên quan đến nhiều siêu tham số như inner batch size, outer batch size, inner training step, outer training step,... Để dễ dàng fine-tune, chúng tôi cố định hầu hết các tham số và chỉ fine-tune kích thước của lookback window, inner và outer learning rate. Chi tiết được trình bày trong bảng III.

TABLE III: Search space for fine-tuning our method.

Hyper-parameter	Seach space
Inner batch size (samples/batch)	{32}
Inner training step	{3}
Outer batch size (tasks/batch)	{5}
Outer training step	{100}
Lookback window	{10, 20, 30}
Inner learning rate	{0.001, 0.005, 0.01, 0.05}
Outer learning rate	{0.001, 0.005, 0.0015, 0.0055}

Appendix C: Experimental detail for NHITS

Đối với mô hình NHITS, chúng tôi dựa trên [4] để định nghĩa không gian tìm kiếm cho việc fine-tune tham số, cũng như kiến trúc mô hình. Đối với các tham số không được đề cập trong bảng, chúng tôi sử dụng giá trị mặc định của cài đặt NHITS trong thư viện NeuralForecast [11]. Chi tiết quá trình fine-tune được trình bày trong phần phụ lục **??**. Kết quả tốt nhất của các lần fine-tune được chọn ra và báo cáo trong nghiên cứu này.

TABLE IV: Search space for fine-tuning NHITS.

Hyper-parameter	Seach space
Random seed	{1}
Number of stacks	{3}
Number of blocks in each stack	{1}
Activation function	{ReLU}
Batch size	{256}
Epoch	{500}
Lookback window	{5, 20, 30}
Pooling kernel	{[2,2,2], [4,4,4], [8,8,8], [8,4,1], [16,8,1]}
Stacks' coefficients	{[168,24,1], [24,12,1], [180,60,1],[40,20,1], [64,8,1]}
Number of MLF layers	{1,2}
Learning rate	{0.001, 0.002, 0.005, 0.01, 0.02}