Đánh giá thời gian thực trong Học Liên tục

Trực tuyến (OCL): Một hy vọng mới

# **Yasir Ghunaim Adel Bibi Kumail Alhamoud Motasem Alfarra**

# **Hasan Abed Al Kader Hammoud Ameya Prabhu Philip H.S. Torr Bernard Ghanem**

# **King Abdullah University of Science and Technology (KAUST) University of Oxford**

# 

# **Tóm tắt**

Các đánh giá hiện tại về phương pháp học liên tục (CL - Continual Learning) thường giả định rằng không có giới hạn về thời gian huấn luyện và tính toán. Đây là một giả định không thực tế ở thế giới thực, thúc đẩy chúng tôi đề xuất ra: một đánh giá thực tế thời gian thực về học liên tục, trong đó luồng sẽ không đợi mô hình hoàn thành việc huấn luyện trước khi đưa thêm những dữ liệu tiếp theo cho việc dự đoán. Để làm điều này, chúng tôi đánh giá các phương pháp CL hiện tại theo chi phí tính toán của chúng.

Chúng tôi tiến hành các thí nghiệm kỹ càng trên CLOC, một bộ dữ liệu quy mô lớn chứa 39 triệu hình ảnh được đánh dấu thời gian với nhãn địa lý. Chúng tôi đưa ra một mô hình cơ sở đơn giản vượt trội hơn các phương pháp CL tiên tiến trong việc đánh giá này, đặt dấu hỏi về tính ứng dụng của các phương pháp hiện có trong môi trường thực tế. Thêm vào đó, chúng tôi khám phá ra các thành phần CL thường được sử dụng trong tài liệu, bao gồm các chiến lược lấy mẫu bộ nhớ và các phương pháp chính quy hoá. Chúng tôi nhận thấy rằng tất cả các phương pháp đã xem xét đều không cạnh tranh được với mô hình cơ sở đơn giản của chúng tôi. Điều này đáng ngạc nhiên cho thấy rằng hầu hết các tài liệu CL hiện có được tùy chỉnh cho một loại dòng dữ liệu cụ thể không thực tế. Chúng tôi hy vọng rằng đánh giá mà chúng tôi cung cấp sẽ là bước đầu tiên trong việc thay đổi mô hình để xem xét chi phí tính toán trong việc phát triển các phương pháp học tiếp tục trực tuyến.

# **1. Giới thiệu**

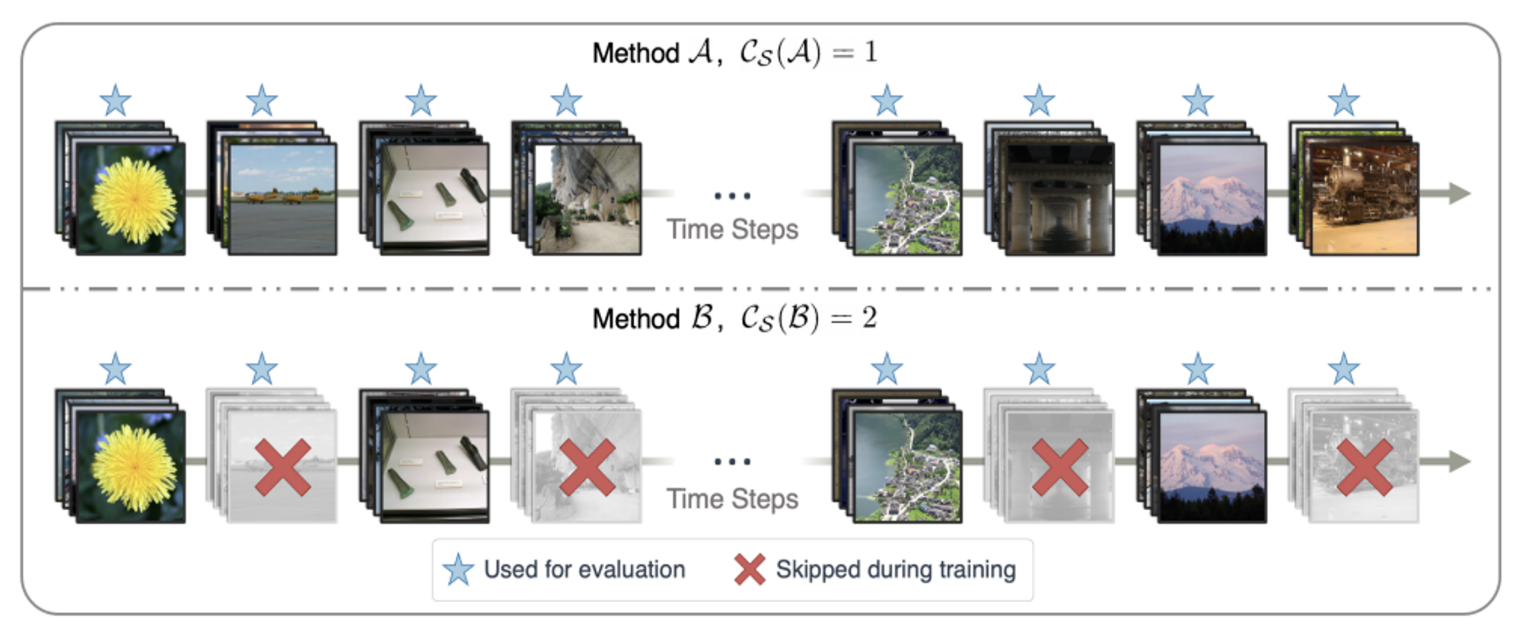
Mạng lưới Nơ-ron sâu (DNN - Deep Neural Network) đã thành công trong việc giải các bài toán phức tạp, khi huấn luyện offline, lặp lại nhiều lần và trên các bộ dữ liệu có nhãn lớn được chọn lọc kỹ lưỡng. Tuy nhiên, trong bối cảnh thực tế, dữ liệu chỉ có sẵn ở dạng luồng với phân bố thay đổi. Do đó, ngày càng có sự quan tâm đến vấn đề học từ một luồng dữ liệu thay đổi theo thời gian, được gọi là CL, là một thách thức lớn với DNN do một hiện tượng gọi là: quên thảm khốc (catastrophic forgetting). Cụ thể, khi một DNN được huấn luyện với dữ liệu từ một phân phối mới, hiệu suất của DNN giảm đáng kể trên dữ liệu đã học trước đó.

Mặc dù những nỗ lực giảm thiểu điều đó được đưa ra, như chính quy hoá việc training, lặp lại các ví dụ trước đó, và những tiếp cận khác, những đánh giá hiện tại vẫn còn xa so với thực tế. Ví dụ, phần lớn các tài liệu về Offline CL, các phương pháp được cho phép có nguồn lực không giới hạn, cả về thời gian và tính toán. Hơn nữa, phần lớn các đánh giá CL được tiến hành trên các bộ dữ liệu quy mô nhỏ với ranh giới phân phối thời gian được xác định rõ ràng dưới dạng học một chuỗi các nhiệm vụ.

Với mục tiêu đó, gần đây đã có sự quan tâm ngày càng tăng đối với cài đặt thực tế hơn - OCL (Online Continual Learning). Theo cách này, các phương pháp CL bị hạn chế chỉ được huấn luyện qua một lần duy nhất trên một phân chia được xáo trộn của các bộ kiểm tra Offline CL hiện có. Điều này chắc chắn là một bước tiến về phía việc giải quyết một số giả định không thực tế của CL ngoại tuyến. Tuy nhiên, các đánh giá hiện tại chưa đủ giải quyết các thách thức của việc học theo thời gian thực với cho các luồng dữ liệu có thông lượng cao và sự thay đổi phân phối nhanh chóng.

Để minh họa điều này, hãy xem xét vấn đề học liên tục từ một luồng Twitter, trong đó có 350.000 tweet được tải lên mỗi phút về các chủ đề đang hot. Mỗi tweet được tải lên cần được dự đoán bằng một mạng neural sâu (DNN) để phát hiện thông tin sai lệch, các từ ngữ hận thù, và các vấn đề khác, đồng thời học và thích nghi với chúng. Với quy mô dữ liệu được cập nhật liên tục, có một hạn chế quan trọng về thời gian và nguồn lực tính toán dành cho việc học các tweet mới, một khía cạnh thường được bỏ qua trong các tài liệu OCL trước đây. Hãy xem xét một phương pháp OCL mà tốc độ huấn luyện chậm hơn 10 lần so với luồng dữ liệu Twitter thông lượng cao, tức là mất 10 phút để huấn luyện trên một phút dữ liệu tweet (350.000 tweets). Sự không hiệu quả này dẫn đến việc tích tụ khoảng 3,1 triệu mẫu mới cần được dự đoán và huấn luyện. Vì không thể chấp nhận việc tạm dừng tất cả các tweet được đăng lên cho đến khi quá trình huấn luyện hoàn thành, các dự đoán cho tất cả các mẫu mới sẽ được thực hiện bằng phiên bản cũ hơn của mô hình. Điều này đặt ra một thách thức quan trọng, trong đó việc học hiệu quả từ luồng dữ liệu trở nên cần thiết. Lý do là vì các phương pháp OCL huấn luyện chậm có thể dẫn đến hiệu suất không tốt, như chúng sử dụng mô hình cũ để dự đoán dữ liệu mới trong luồng. Điều này sẽ trở nên tồi tệ hơn đối với các luồng dữ liệu có sự thay đổi phân phối nhanh hơn.

Trong bài báo này, chúng tôi đề xuất một giao thức đánh giá thời gian thực cho OCL, cân nhắc độ phức tạp tính toán trong quá trình huấn luyện. Cho một luồng dữ liệu, xem xét phương pháp OCL ***A*** có tốc độ bằng với luồng dữ liệu, tức là ***A*** có thể huấn luyện trên mỗi bước dữ liệu được đưa ra trước khi có các mẫu mới. Tiếp theo, nếu một phương pháp OCL *B* tốn gấp đôi so với ***A***, thì ***B*** sẽ cập nhật mô hình cho quá trình đánh giá sau mỗi bước luồng dữ liệu khác nhau, tức là mô hình sẽ được cập nhật một nửa số lần so với ***A***. Hình 1 minh họa giao thức đánh giá thời gian thực mà chúng tôi đề xuất. Điều này trái ngược với các giải pháp kỹ thuật trước đây, khi mà (1) cho phép một khả năng tính toán không giới hạn để huấn luyện trên bất kỳ dữ liệu luồng nhất định nào một cách bất hợp lý, và (2) so sánh không công bằng các phương pháp OCL mặc dù có độ phức tạp huấn luyện khác nhau. Sử dụng giao thức đánh giá thời gian thực của chúng tôi, chúng tôi đánh giá nhiều phương pháp OCL hiện có so với một tiêu chuẩn cơ bản đơn giản và giá rẻ, giảm thiểu hiện tượng quên bằng cách đơn giản là lưu trữ và phát lại các mẫu đã được quan sát gần đây.



***Hình 1.*** *OCL Real-Time Evaluation Example*

**Những đóng góp:**

(1) Chúng tôi chỉ ra rằng dưới giao thức đánh giá thời gian thực của chúng tôi, mô hình cơ sở đơn giản của chúng tôi vượt trội hơn tất cả các phương pháp được xem xét từ tài liệu OCL, bao gồm các phương pháp tiên tiến như ACE.

(2) Chúng tôi xem xét một cài đặt bổ sung với luồng dữ liệu chậm bằng với phương pháp OCL tốn nhiều chi phí huấn luyện nhất, và so sánh phương pháp đó với tiêu chuẩn tính toán tương đương. Dưới cài đặt tính toán được chuẩn hóa này, chúng tôi phát hiện rằng tiêu chuẩn tính toán tương đương vượt trội hơn tất cả các phương pháp hiện có.

(3) Các thí nghiệm của chúng tôi nhất quán, áp dụng cho tất cả các chiến lược học liên tục được xem xét, và rộng rãi, với hơn 2 tháng GPU tính toán. Kết quả của chúng tôi nhấn mạnh rằng quá trình hiện tại trong OCL cần được xem xét lại và cần một sự chuyển đổi mô hình. Chúng tôi hy vọng công việc của chúng tôi sẽ dẫn đến một hướng mới cho học liên tục, mà có liên quan đến chi phí tính toán của mỗi phương pháp.

# **2. Nghiên cứu liên quan**

**Học liên tục ngoại tuyến**

Các chiến lược học liên tục truyền thống, với mục đích giảm sự quên, có thể chia làm 3 nhóm phương pháp:

(i) Replay based method: Lưu trữ các mẫu đầu vào, hoặc học cách tự tạo các mẫu đại diện, trong khi mô hình học từ luồng dữ liệu. Sau đó, phương pháp sẽ huấn luyện lại mô hình từ mẫu phát lại trong khi mô hình học từ dữ liệu mới.

(ii) Regularization methods: tránh chi phí lưu trữ các mẫu và đơn giản chỉnh sửa mục tiêu mất mát của mô hình để chính quy hóa quá trình huấn luyện. Trong khi một số phương pháp thay đổi các tham số quan trọng của mô hình, các phương pháp khác chính quy hóa quá trình huấn luyện bằng cách rút trích kiến thức từ một mô hình đã được huấn luyện trên các mẫu luồng dữ liệu trước đó.

(iii) Parameter isolation methods: huấn luyện các tham số cụ thể với task trong khi đóng băng các tham số khác.

Mặc dù đã có tiến bộ bởi các phương pháp này, chúng giả định một luồng dữ liệu ngoại tuyến cho phép nhiều lần đi qua mỗi tác vụ học liên tục. Công trình đồng thời của Prabhu và đồng nghiệp, giảm thiểu vấn đề này bằng cách ràng buộc các nguồn lực tính toán, và phát hiện rằng các phương pháp đơn giản dựa trên phát lại vượt trội hơn hầu hết các công trình học liên tục trước đó. Ngược lại, chúng tôi nghiên cứu cài đặt thực tế hơn, trong đó luồng dữ liệu tiết lộ dữ liệu theo thời gian thực.

**Học trực tuyến để giảm việc quên**

OCL (Online Continual Learning) được định nghĩa với một giao thức trong đó dữ liệu huấn luyện chỉ được nhìn thấy một lần trong một chuỗi các nhiệm vụ được gán nhãn. Để giảm thiểu việc quên thảm khốc, lĩnh vực ban đầu đã tiến bộ trong việc áp dụng các ràng buộc dựa trên gradient tốt hơn như GEM và AGEM. RWALK định lượng việc quên và cung cấp một cách tiếp cận hiệu quả hơn để giới hạn sự thay đổi các tham số quan trọng trong mô hình. TinyER [11] khám phá lại hiệu quả của việc phát lại kinh nghiệm và HAL [9] nâng cao nó bằng các anchor đã học. Tuy nhiên, cài đặt này giả định sẵn có một "oracle" trong thời gian kiểm tra để xác định đầu ra phân loại nào được sử dụng cho suy luận. Ngoài ra, các bộ kiểm tra trong cài đặt này thường có ràng buộc lớn về kích thước bộ đệm phát lại. Do những hạn chế này, sự tăng dần theo lớp (class-incremental) bắt đầu thu hút sự chú ý, là một cài đặt thực tế hơn cho OCL.

Trong việc học liên tục trực tuyến tăng dần theo lớp, các bộ kiểm tra đã giảm bớt yêu cầu về mô tả tác vụ (task descriptors) trong thời gian kiểm tra và giới thiệu kích thước bộ đệm lớn hơn đáng kể so với các phương pháp trước đó. Có nhiều hướng tiếp cận đã xuất hiện để giải quyết hiện tượng quên thảm khốc trong cài đặt này. Nó có thể được chia làm các nhóm:

* Regularization-based approaches: sửa đổi các mục tiêu phân loại để giữ lại các biểu diễn đã học trước đó hoặc khuyến khích các biểu diễn có ý nghĩa hơn. Ví dụ: DER, ACE, và CoPE.
* Sampling-based techniques: tập trung vào việc tối ưu lựa chọn và lưu trữ bộ nhớ phát lại đại diện nhất trong quá trình huấn luyện trực tuyến, ví dụ như GSS [4], OCS [52], CBRS [12], CLIB [26], và InfoRS [47]. Ngoài ta, có một số phương pháp lấy mẫu tập trung vào các chiến lược truy xuất bộ nhớ tốt hơn để giảm tình trạng quên, ví dụ: MIR [3], ASER [44] và GMED
* Trong một số phương pháp khác, GDumb [38] đề xuất một giải pháp đơn giản cho vấn đề học trực tuyến bằng cách bỏ qua dữ liệu luồng và chỉ tập trung vào việc học từ các mẫu trong bộ nhớ. Mặc dù những nỗ lực này đã đạt được tiến bộ đáng kể trong lĩnh vực OCL, chúng thường được đánh giá trên các bài kiểm tra không phản ánh điều kiện triển khai thực tế:
  + Đầu tiên, những bài kiểm tra này dựa nhiều vào các tập dữ liệu nhỏ được tạo một cách nhân tạo với sự thay đổi đột ngột trong các lớp.
  + Thứ hai, những bài kiểm tra này không thể đo lường được khả năng của mô hình thích nghi nhanh với dữ liệu mới dưới sự thay đổi nhanh chóng của phân phối, đây là một trong những vấn đề chính trong tài liệu học trực tuyến cổ điển.

Đã có những nỗ lực để giải quyết giới hạn thứ hai bằng cách sử dụng các độ đo mới để đo tần suất chính xác của bài kiểm tra trong quá trình huấn luyện. Tuy nhiên, những độ đo này chỉ đánh giá sự thích nghi với những dữ liệu đã được giữ lại mà không đánh giá sự thích nghi với dữ liệu tương lai. Để khắc phục những giới hạn này, đã có một sự gia tăng mới về các bài kiểm tra đề xuất tập dữ liệu và giao thức đánh giá, chúng ta sẽ thảo luận về điều này tiếp theo.

**Học trực tuyến để thích ứng nhanh chóng**

Các bài kiểm tra OCL gần đây, ví dụ như CLOC và CLEAR, giới thiệu dữ liệu được sắp xếp theo thời gian, tạo thành một luồng thời gian dài của các khái niệm hình ảnh tiến hóa . Họ chứng minh rằng dữ liệu của họ có sự thay đổi phân phối tự nhiên theo thời gian, đòi hỏi sự thích nghi nhanh chóng với dữ liệu mới hơn. Ngoài ra, họ mô phỏng cài đặt học trực tuyến truyền thống bằng cách đo khả năng thích nghi nhanh chóng với việc đánh giá được thực hiện trên dữ liệu tương lai từ dòng dữ liệu. Công việc của chúng tôi mở rộng những nỗ lực trong hướng này. Chúng tôi áp dụng bài kiểm tra CLOC và đề xuất một phương pháp đánh giá thời gian thực hiện thực tế hơn để khuyến khích việc học hiệu quả. Đáng lưu ý là Tyler L Hayes và Christopher Kanan (19) đã khám phá một số khía cạnh của việc đánh giá thời gian thực trong học liên tục. Tuy nhiên, họ tập trung vào hiệu quả cho việc triển khai trên thiết bị nhúng, chứ không phải sự thích nghi nhanh chóng. Hơn nữa, trong khi công việc của họ điều tra chi phí tính toán liên quan đến mỗi phương pháp huấn luyện, họ chỉ báo cáo chi phí là một chỉ số đánh giá. Ngược lại, chúng tôi giới hạn quy trình huấn luyện theo chi phí tính toán của mỗi phương pháp.

# **3. Phương pháp**

Chúng ta bắt đầu với bài toán cổ điển cho bài toán OCL. Sau đó, chúng tôi chính thức giới thiệu đánh giá theo thời gian thực được đề xuất của mình, các yếu tố gây ra độ phức tạp trong đào tạo thông qua các đánh giá trễ.

## **3.1 Học liên tục trực tuyến (OCL)**

OCL là bài toán học một hàm tham số θ: dự đoán nhãn y ∈ **Y** với input là hình ảnh x ∈ **X.**  Không giống với học giám sát cổ điển, việc học được thực hiện trên một luồng dữ liệu phân phối thay đổi S, đưa ra dữ liệu tuần tự qua các bước t ∈ {1, 2,…, ∞}. Cụ thể, ở mỗi bước t:

* S đưa ra tập các hình ảnh:
* đưa ra dự đoán: với các x
* S đưa ra nhãn thật sự:
* được đánh giá bằng việc so sánh nhãn dự đoán và nhãn thật sự
* Một phương pháp học huấn luyện , tính toán hàm mất mát, và cập nhật tham số thành

Lưu ý rằng là một phân phối biến đổi có thể không cần thiết phải thay đổi tại mỗi bước luồng t. Ví dụ, nếu ở bước t = 5 chúng ta có j = 1, điều này có nghĩa là dữ liệu được tiết lộ qua năm bước trước đó được lấy mẫu từ cùng một phân phối . Hơn nữa, chúng ta cũng nhận thấy rằng, khác với việc học liên tục ngoại tuyến, việc học liên tục trực tuyến bao gồm khả năng thích nghi với dữ liệu mới vì dự đoán được thực hiện trước quá trình huấn luyện.

### **Các vấn đề chính.**

Như đã giới thiệu trước, đánh giá OCL trong các tài liệu thường bỏ qua độ phức tạp huấn luyện ở bước (5). Trong các cài đặt như trên, các phương pháp OCL cập nhật tham số thành trước khi S đưa ra hình ảnh tiếp theo. Do đó, các phương pháp OCL khác nhau được đánh giá mà không quan tâm đến thời gian huấn luyện và độ phức tạp tính toán ở bước (5). Điều này tương đương với việc đánh giá các phương pháp OCL khác nhau trên các luồng dữ liệu mà đưa ra dữ liệu ở tốc độ khác nhau. Trong cài đặt thời gian thực, luồng dữ liệu đưa ra dữ liệu với một tốc độ độc lập với độ phức tạp huấn luyện của phương pháp OCL. Vì vậy, chúng tôi đề xuất một mô hình đánh giá thời gian thực cho OCL, tính đến độ phức tạp huấn luyện thông qua việc đánh giá trễ.

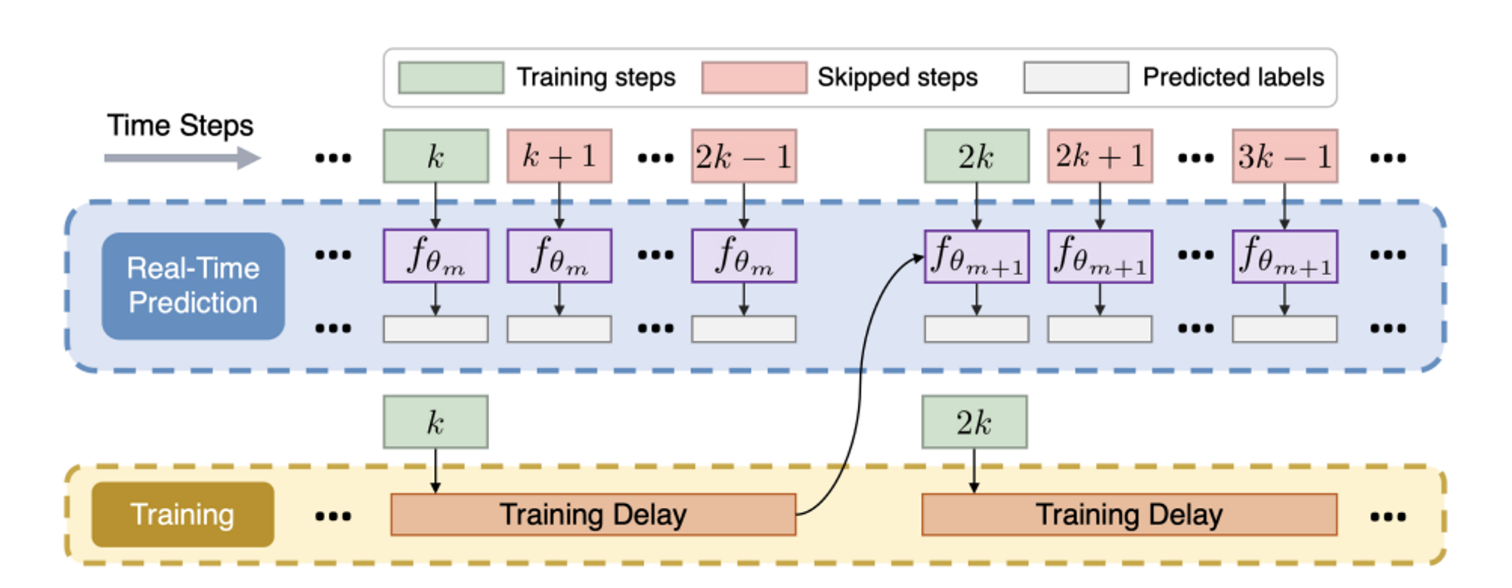
## **3.2. Đánh giá thời gian thực cho OCL**

Như đã đề cập trước đó, chúng ta cần định nghĩa khái niệm về tốc độ luồng cố định. Điều này có nghĩa là luồng dữ liệu tiết lộ dữ liệu với một tốc độ cố định, bất kể OCL mất bao lâu để cập nhật mô hình ở bước (5). Đối với các luồng dữ liệu nhanh gấp đôi so với thời gian huấn luyện , các phương pháp OCL sẽ chỉ cập nhật mô hình trên một nửa số lượng tập hình ảnh được đưa ra từ luồng. Lưu ý rằng phiên bản được cập nhật mới nhất sẽ tiếp tục cung cấp dự đoán cho các hình ảnh mới được đưa ra trong quá trình huấn luyện, ngay cả đối với những hình ảnh mà mô hình không được huấn luyện. Cách thiết lập này phản ánh thực tế khi máy chủ nhận nhiều truy vấn từ người dùng, trong đó mỗi truy vấn cần một dự đoán bất kể việc có đang huấn luyện một mô hình nội bộ hay không.

Cho một luồng S và phương pháp OCL A, ban đầu định nghĩa độ phức tạp tương đối của mô hình luồng: . Với = 1, A có thể cập nhật mô hình θ trong bước 5 trước khi luồng S cung cấp thêm dữ liệu. Với = k > 1, luồng sẽ nhanh hơn k lần so với A. Với định nghĩa này, chúng tôi cung cấp một phiên bản đã chỉnh sửa của OCL, cùng với việc đánh giá thời gian thực tương ứng.

Cho mô hình , luồng , . Với mỗi bước t:

* S đưa ra tập các hình ảnh: .
* đưa ra dự đoán: với các x
* S đưa ra nhãn thật sự:
* được đánh giá bằng việc so sánh nhãn dự đoán và nhãn thật sự
* Nếu mod(t - 1, k) = 0, thì A sẽ cập nhật và một phiên huấn luyện trên bắt đầu (Xem hình 2).

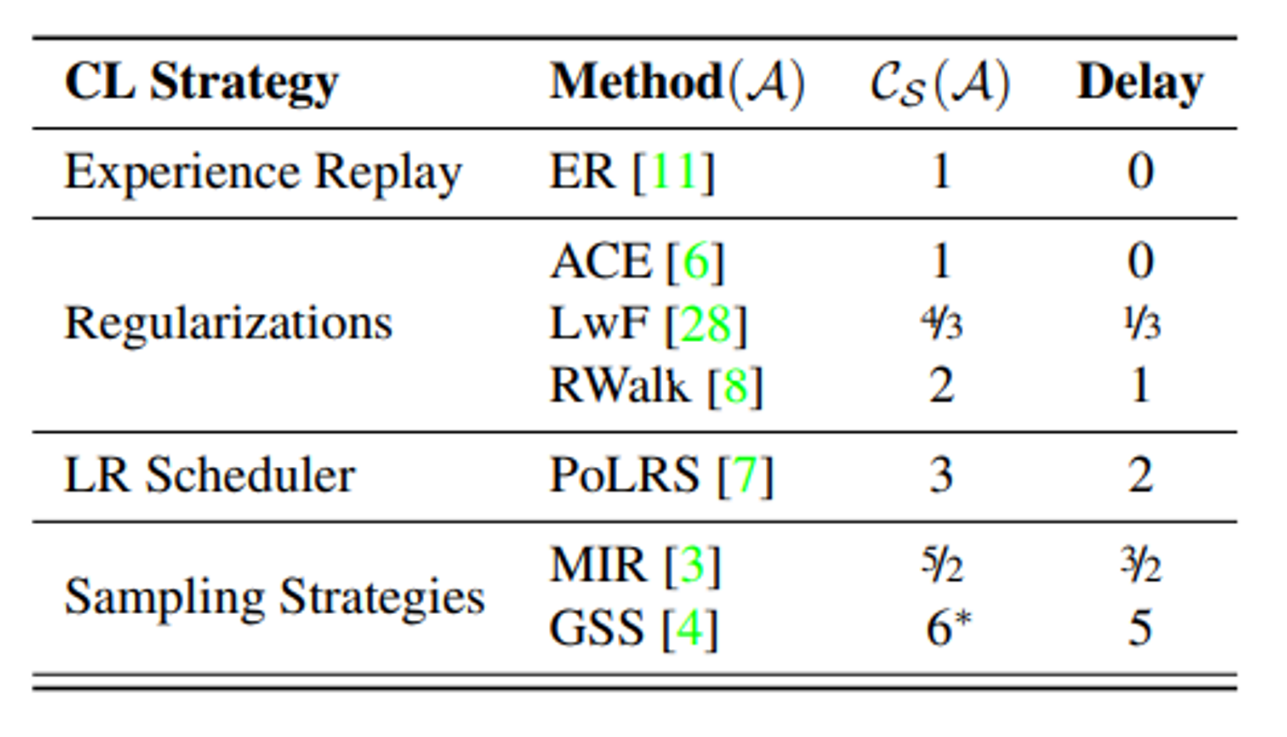


***Hình 2.*** *Do không khớp giữa tốc độ luồng và chi phí tính toán của mô hình, K - 1 lô huấn luyện tiềm năng được bỏ qua cho mỗi bước huấn luyện.*

Trong cài đặt này, các phương pháp OCL mà có chi phí tính toán để huấn luyện cao hơn sẽ được cập nhật ít lần hơn. Vì thế, với mỗi luồng S có phân phối Dj thay đổi nhanh, phương pháp OCL có độ phức tạp tương đối lớn giữa luồng và mô hình có thể tạo ra dự đoán sai lầm, vì các mô hình không được cập nhật đủ và không thể thích nghi với sự thay đổi phân phối.

**Về mặt tính toán của** CS. Vì CS chỉ đo độ phức tạp giữa luồng và phương pháp OCL, chúng tôi trước tiên giới thiệu một phương pháp OCL tối thiểu với chi phí thấp như một baseline (được ký hiệu là ***A***). Để dễ dàng so sánh các phương pháp được xem xét, chúng tôi giả định rằng, do ***A*** có chi phí thấp, việc học liên tục trực tuyến sẽ nhanh như luồng, trong đó độ phức tạp tương đối giữa luồng và mô hình luồng CS = 1.

Cụ thể, chúng tôi xem xét một baseline giảm thiểu việc quên thông qua việc lưu trữ và phát lại các mẫu đã được học gần đây. Sau đó, với bất kỳ phương pháp OCL ***B*** khác, chúng tôi sử dụng số lượng phép tính (FLOPs: floating-point operations) huấn luyện tương đối giữa ***B*** và ***A*** để xác định . Ví dụ, ACE [6] chỉ thay đổi mục tiêu mất mát của baseline, do đó tương đương với baseline về phức tạp tính toán. Trong khi đó, PoLRS [7] duy trì và thực hiện các hoạt động trên ba bản sao của mô hình. Các bản sao này được huấn luyện trên mỗi lô dữ liệu đến, khiến cho PoLRS yêu cầu 3 lần số FLOPs cần thiết so với baseline, do đó gấp 3 lần baseline về tính toán. Điều này tương ứng với việc trì hoãn cập nhật mô hình hai bước. Trong Bảng 1, chúng tôi tóm tắt sự trì hoãn tương ứng của một số phương pháp OCL phổ biến trong cài đặt đánh giá thời gian thực của chúng tôi.



*Bảng 1: Độ phức tạp và độ trễ huấn luyện của các phương pháp OCL*

**So sánh công bằng các phương pháp OCL với khác nhau**

Chúng tôi đề xuất một cài đặt thực tế để đánh giá các hệ thống học liên tục dựa trên độ phức tạp huấn luyện của chúng đối với tốc độ luồng dữ liệu. Tuy nhiên, có thể có người cho rằng các quy trình huấn luyện với chi phí cao hơn có thể được tăng tốc bằng cách triển khai nhiều phần cứng tính toán hơn để huấn luyện trên mỗi mẫu được đưa ra từ luồng dữ liệu. Ví dụ, trong khi PoLRS [7] yêu cầu 3 x FLOPs so với baseline, ta có thể triển khai thêm 3 lần phần cứng để làm cho C(A) = 1 đối với PoLRS.

Trong khi thiết lập nói trên bình thường hóa yêu cầu tính toán của một phương pháp học ***A*** để phù hợp với tốc độ luồng, ta nên cho phép các phương pháp huấn luyện đơn giản có cùng độ phức tạp tính toán. Với mục đích đó, chúng tôi đề xuất tăng cường các phương pháp huấn luyện đơn giản và nhanh hơn để phù hợp với yêu cầu tính toán của các phương pháp phức tạp hơn. Ví dụ, chúng tôi tăng cường phương pháp experience replay đơn giản bằng cách lấy mẫu một số lượng lớn hơn từ bộ đệm tại mỗi bước t để phù hợp với FLOPs của các kế hoạch huấn luyện khác. Chúng tôi lưu ý rằng mặc dù sửa đổi này đơn giản và ngây thơ, nhưng chúng tôi đã thấy từ thực nghiệm rằng nó đủ để vượt qua tất cả các phương pháp OCL được xem xét. Chúng tôi để lại các chi tiết thực hiện còn lại cho Phần 4.2.

# **4. Thực nghiệm**

Chúng tôi so sánh các phương pháp OCL trong cài đặt đánh giá thời gian thực mà chúng tôi đề xuất, với hai tốc độ luồng khác nhau: nhanh và chậm, mô phỏng các kịch bản ứng dụng khác nhau. Chúng tôi đánh giá hiệu suất của hầu hết các phương pháp OCL thông qua việc phân loại chúng vào ba nhóm khác nhau.

Regularization-based OCL methods: chính quy hoá huấn luyện bằng việc điều chỉnh hàm mất mát phân loại. Ví dụ, phương pháp RWalk [8] sử dụng một chính quy hoá để phạt các thay đổi đột ngột trong các tham số của mô hình. Phương pháp ACE [6] giới thiệu một hàm mất mát không đối xứng để xử lý các mẫu trong bộ nhớ đệm khác biệt với các mẫu trong luồng dữ liệu đến, trong khi phương pháp LwF [28] trích xuất kiến thức từ các bước trước đó.

Learning rate scheduler methods: trong trường hợp này là PoLRS(7), linh hoạt điều chỉnh tốc độ học dựa theo thay đổi trong phân bổ luồng

Sampling-based method: sửa đổi chiến lược được sử dụng để cập nhật bộ nhớ đệm, ví dụ như phương pháp GSS [4], hoặc thay đổi chiến lược truy xuất bộ nhớ, ví dụ như phương pháp MIR [3].

**Các bộ dữ liệu.**

Chúng tôi sử dụng bộ dữ liệu OCL quy mô lớn CLOC [7], bao gồm 39 triệu hình ảnh được đánh dấu thời gian, thể hiện các sự thay đổi phân phối tự nhiên. Nhiệm vụ là xác định vị trí địa lý của một hình ảnh cụ thể, trong đó tổng số nhãn vị trí địa lý là 712. Để đảm bảo tính nhất quán với CLOC [7], chúng tôi áp dụng cùng phương pháp chia tập dữ liệu. Cụ thể, chúng tôi sử dụng 5% đầu tiên của luồng dữ liệu để lựa chọn siêu tham số, lấy mẫu đều 1% từ phần còn lại của luồng để xây dựng một tập dữ liệu giữ ra để đo đạc chuyển đổi ngược/ xuôi (backward/forward transfer, được báo cáo trong phụ lục), và sử dụng phần còn lại của luồng dữ liệu để huấn luyện và đánh giá trực tuyến. Tương tự như CLOC [7], chúng tôi so sánh các phương pháp OCL bằng cách sử dụng chỉ số Độ chính xác Trung bình Trực tuyến (**Average Online Accuracy**), đo lường khả năng của mô hình thích nghi với các mẫu luồng dữ liệu đến.

**Chi tiết cài đặt.**

Tại mỗi bước t, S hiển thị một tập hình ảnh gồm 128 hình ảnh, và các phương pháp OCL mở rộng bộ huấn luyện bằng 128 hình ảnh khác được lấy mẫu từ bộ nhớ đệm. Quá trình này được thực hiện cho đến khi hoàn thành một lượt đi qua toàn bộ luồng dữ liệu. Chúng tôi sử dụng cơ sở là ResNet50 [7, 20] đã được huấn luyện trước trên ImageNet [15]. Chúng tôi sử dụng SGD với tỷ lệ học (learning rate) là cho tất cả các phương pháp OCL trừ PoLRS [7], phương pháp này hoạt động tốt nhất với tỷ lệ học là . Trừ khi có ghi chú khác, chúng tôi đặt kích thước của bộ đệm replay là cho tất cả các phương pháp được xem xét.

Chúng tôi sử dụng các siêu tham số được báo cáo trong các bài báo tương ứng, trừ thông số chính quy cho phương pháp RWalk [8]. Chúng tôi nhận thấy rằng thông số chính quy hoá λ = 0.1 được báo cáo quá nhỏ cho bộ dữ liệu CLOC; chúng tôi đã thực hiện xác thực chéo (cross validate) và tìm ra rằng λ = 2 là tốt nhất. Phần thực hiện của chúng tôi mở rộng mã nguồn từ Avalanche [31], một framework học liên tục phổ biến. Chúng tôi để lại các chi tiết cài đặt còn lại và thí nghiệm khác cho phụ lục.

## **4.1 Luồng nhanh: Đánh giá thời gian thực**

Đầu tiên, chúng tôi xem xét kịch bản luồng nhanh, trong đó luồng dữ liệu hiển thị dữ liệu với tốc độ cao. Chúng tôi xem xét một mô hình cơ sở đơn giản là ER [11], mô hình này thực hiện một bước gradient trên một lô gồm 256 hình ảnh (128 từ luồng dữ liệu và 128 từ bộ nhớ) tại mỗi bước luồng. 128 hình ảnh từ bộ nhớ được lấy mẫu đồng đều. Để đơn giản và dễ so sánh, chúng tôi giả định rằng ER có độ phức tạp tương đối của mô hình luồng là = 1. Do đó, bất kỳ phương pháp OCL nào tốn nhiều tài nguyên tính toán hơn ER sẽ không theo kịp tốc độ luồng, và do đó sẽ buộc phải bỏ qua việc huấn luyện trên một phần của các mẫu luồng dữ liệu đến.

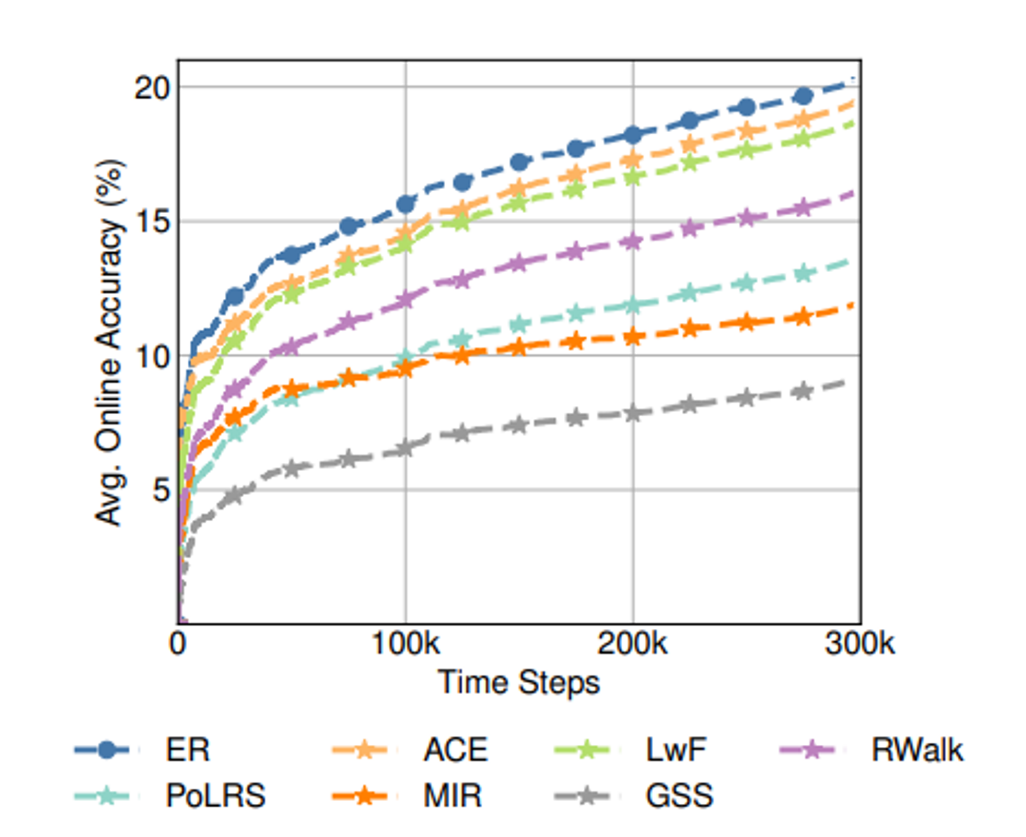
**Đo lường độ phức tạp tương đối của các phương pháp OCL.**

Như đã thảo luận trước đó, chúng tôi sử dụng tổng số FLOP (Floating Point Operations) cần thiết cho mỗi phương pháp huấn luyện trên một luồng dữ liệu như một chỉ số đại diện cho độ phức tạp tương đối của mô hình so với luồng dữ liệu, gọi là (stream-model relative complexity). Đầu tiên, chúng tôi tính toán tổng số FLOP cần thiết để huấn luyện ER trên một tập con nhỏ của CLOC gồm 103 hình ảnh với kích thước lô là 10. Sau đó, chúng tôi tính toán các FLOP tương ứng cho các phương pháp OCL khác nhau và chuẩn hóa chúng theo FLOP của ER. Vì cùng sử dụng cấu trúc chính (backbone), chúng tôi tiếp tục xác minh tính phức tạp đã chuẩn hóa này bằng cách đếm số lần truyền tiến và lùi hiệu quả của mỗi phương pháp. Vì chúng tôi giả định rằng ER nhanh chóng như luồng dữ liệu, các phương pháp tốn gấp đôi FLOP sẽ có độ phức tạp bằng 2.

Chúng tôi báo cáo trong Bảng 1 cho các phương pháp OCL so với ER. Lưu ý rằng PoLRS tốn 3 lần tài nguyên so với ER vì nó yêu cầu huấn luyện 3 mô hình khác nhau để cập nhật tỷ lệ học. Ngoài 128 hình ảnh được hiển thị bởi luồng dữ liệu và 128 hình ảnh được lấy mẫu từ bộ nhớ tại bước t, GSS thực hiện các truyền tiến và lùi thêm trên 10 × 128 mẫu từ bộ nhớ để đảm bảo đa dạng khi cập nhật bộ đệm tái trải nghiệm [4]. Do đó, GSS gần 6 lần tốn kém hơn so với ER. Đối với mỗi phương pháp trong Bảng 1, chúng tôi báo cáo độ trễ tương ứng trong các bước luồng, đây là hậu quả của một số phương pháp chậm hơn so với luồng. Một độ trễ là 2 có nghĩa là luồng sẽ hiển thị hình ảnh tương đương với 2 bước trước khi mô hình được cập nhật.

**Hiệu ứng của Hiệu Quả Đào Tạo**

Chúng tôi bắt đầu phân tích bằng việc nghiên cứu tác động của huấn luyện hiệu quả đến hiệu suất của các phương pháp OCL khác nhau trong thiết lập đánh giá thời gian thực. Chúng tôi vẽ đồ thị đường cong Độ chính xác Trực tuyến Trung bình (**Average Online Accuracy)** theo từng bước trong Hình 3, trong đó ER là mô hình cơ sở màu xanh lam. Mỗi phương pháp được đánh giá dựa trên độ trễ huấn luyện tương ứng được báo cáo trong Bảng 1.



***Hình 3.*** *Đánh Giá Dòng Nhanh*

Đáng ngạc nhiên, ER vượt trội hơn tất cả các phương pháp được xem xét, và trong một số trường hợp, chênh lệch rất lớn. Thú vị là những phương pháp tốn nhiều tài nguyên tính toán nhất, MIR, PoLRS và GSS, cho thấy hiệu suất thấp nhất khi đánh giá thời gian thực. Cụ thể, khoảng cách hiệu suất ở cuối luồng đạt tới 11%. Thực tế, chúng tôi nhận thấy rằng giá trị Độ chính xác Trực tuyến Trung bình của tất cả các phương pháp được xem xét đều được xếp gần như theo thứ tự dựa trên CS; CS càng lớn, hiệu suất càng kém trong đánh giá thời gian thực. Chúng tôi cho rằng xu hướng này có nguyên nhân từ việc các phương pháp không hiệu quả này có thể không đủ khả năng theo kịp sự thay đổi nhanh chóng của phân phối Dj trong luồng dữ liệu khi chúng phải chịu độ trễ cao hơn.

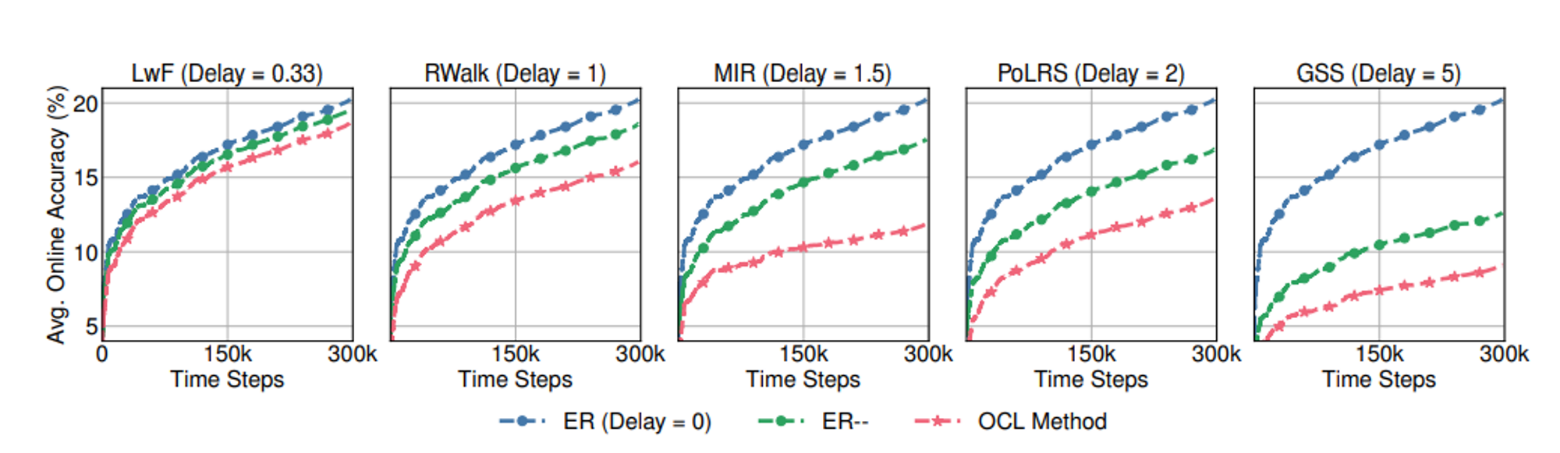
Ví dụ, mặc dù PoLRS được đề xuất và tinh chỉnh cho bộ kiểm tra CLOC gần đây, nó vẫn bị LwF, một phương pháp cũ hơn, vượt trội đáng kể khi được đánh giá trong môi trường thời gian thực. Điều này gợi ý rằng độ phức tạp tính toán có thể là một điểm yếu cho các phương pháp OCL học trên các luồng dữ liệu nhanh. Kết quả của chúng tôi cho thấy trong khi các tài liệu hiện tại nhằm cải thiện quá trình học bằng cách tận dụng các quy trình học chi phí cao và tiên tiến hơn, hình phạt do độ trễ làm lu mờ bất kỳ cải tiến tiềm năng nào đạt được từ các đóng góp thuật toán. Đáng chú ý, phương pháp ACE tiên tiến nhất, có hiệu suất tương đương với ER và do đó được đánh giá mà không có độ trễ huấn luyện, đạt hiệu suất gần như tương đương với mô hình cơ sở. Hơn nữa, ACE vượt trội hơn các phương pháp đắt đỏ hơn khi được đánh giá trên các tập dữ liệu quy mô nhỏ (xem Small-Scale Experiments trong phụ lục). Tổng thể, những kết quả của chúng tôi cho thấy rằng các phương pháp OCL thực tế, triển khai trong các ứng dụng thời gian thực, nên ưu tiên học hiệu quả hơn các quá trình tối ưu hóa đắt đỏ.

**Chuẩn hóa Dữ liệu Huấn luyện**

Trước đây, chúng tôi đã chỉ ra rằng việc đánh giá trễ dẫn đến sự suy giảm hiệu suất lớn hơn khi độ trễ càng lớn. Tuy nhiên, điều này có thể được cho là do các phương pháp OCL với độ trễ lớn cuối cùng sẽ được huấn luyện hiệu quả trên ít mẫu hơn. Điều này đặt ra câu hỏi: *Có phải các phương pháp đắt đỏ hơn về tính toán với độ trễ có hiệu suất kém hơn so với ER vì chúng không thể đối phó với sự thay đổi của phân phối trong luồng dữ liệu?*, hay *Liệu sự suy giảm hiệu suất là do các phương pháp OCL đắt đỏ bị huấn luyện trên ít tập hơn?*

Để giải quyết những câu hỏi này, chúng tôi tiến hành so sánh hai phương pháp ER và mỗi phương pháp OCL đã được báo cáo trước đó. Giả sử tốc độ luồng được cố định, chúng tôi sửa đổi ER để phù hợp với chi phí tính toán của phương pháp tương ứng bằng cách huấn luyện trên cùng một lượng dữ liệu trong một khoảng thời gian lâu hơn. Để đạt được điều này, chúng tôi đặt ER chịu một độ trễ bằng cách thực hiện các bước gradient descent bổ sung tại mỗi bước trong luồng. Chúng tôi gọi phiên bản cơ sở được sửa đổi này là ER--. Số lượng cập nhật bổ sung trong ER-- được xác định sao cho độ trễ của nó phù hợp với độ trễ của phương pháp OCL tương ứng. Điều này đảm bảo rằng ER-- được huấn luyện trên cùng số lượng ví dụ huấn luyện so với các phương pháp OCL khác.

Hình 4 chỉ ra sự so sánh của ER và ER-- với mỗi phương pháp OCL.



***Hình 4.*** *Fast Stream - Training Data Normalization*

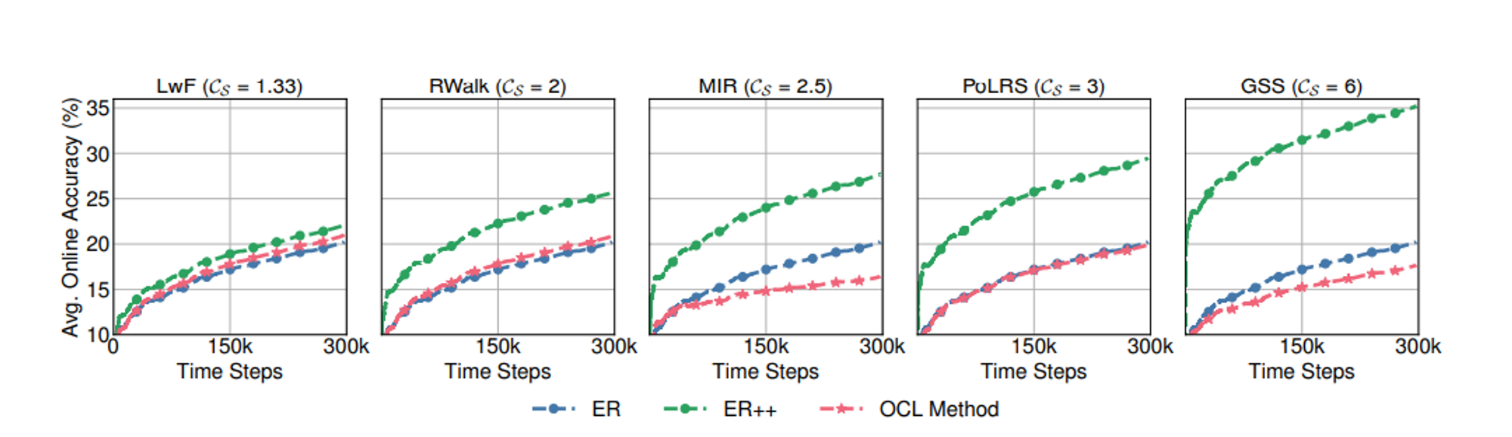
Vì ER hiệu quả hơn ER--, ER liên tục vượt trội hơn ER--, xác nhận rằng sự hiệu quả là yếu tố quan trọng cho việc đánh giá thời gian thực của OCL. Đáng chú ý hơn, mặc dù ER-- phù hợp với độ phức tạp của mỗi phương pháp được so sánh và chịu cùng một độ trễ, nó vẫn vượt trội hơn tất cả các phương pháp OCL được xem xét. Hơn nữa, trong khi khoảng cách giữa ER-- và các phương pháp đắt đỏ, ví dụ như GSS, nhỏ hơn khoảng cách đến ER, các phương pháp đắt đỏ vẫn kém hơn ER-- lên đến 3,5%. Điều này cho thấy hiệu suất suy giảm của các phương pháp OCL được xem xét không phải do số lượng ví dụ huấn luyện quan sát được ít hơn. Ngược lại, các phương pháp OCL đắt đỏ dường như không thể đối phó được với các luồng dữ liệu thay đổi phân phối. Trong quá trình đánh giá thời gian thực của các luồng nhanh, các phương pháp đơn giản như ER và ER-- có thể phù hợp hơn cho triển khai thực tế.

## **4.2. Luồng chậm: Chuẩn hóa độ phức tạp**

Trong cài đặt luồng nhanh, chúng tôi xem xét kịch bản trong đó luồng chạy nhanh như cơ sở ER, tức = 1. Trong phần này, chúng tôi xem xét các luồng chạy chậm giống với các phương pháp OCL đắt đỏ hơn. Thiết lập này đặt ra câu hỏi sau: *Các phương pháp OCL hiện tại hoạt động như thế nào nếu chúng được triển khai trên các luồng phù hợp với độ phức tạp huấn luyện của chúng?* Luồng chạy chậm có thể cho phép các phương pháp đắt đỏ huấn luyện trên toàn bộ luồng mà không cần bỏ qua bất kỳ bước nào. *Liệu các phương pháp đã được xem xét trước đây gặp khó khăn trong cài đặt luồng nhanh, có phù hợp cho triển khai OCL trong tình huống luồng chạy chậm này không?*

Chúng tôi so sánh các phương pháp OCL khác nhau, không có độ trễ với cơ sở ER++ tương ứng, tính toán thêm số lượng bước gradient cho mỗi lô để phù hợp với độ phức tạp của phương pháp được so sánh, như được xác định từ Bảng 1. Điều này mô phỏng việc so sánh trên tốc độ luồng chạy chậm, trong đó không có bước luồng nào được bỏ qua và so sánh được thực hiện dựa trên độ phức tạp chuẩn hóa (**normalized complexity**). Lưu ý rằng ACE chỉ sửa đổi mục tiêu mất mát của ER, vì vậy nó phù hợp với độ phức tạp cơ sở . Do đó, việc đánh giá ACE trong cài đặt luồng chạy chậm hoàn toàn giống việc đánh giá nó trong cài đặt luồng chạy nhanh, đã được thực hiện trước đó trong Hình 3. Vì vậy, chúng tôi không so sánh với ACE lại trong các thí nghiệm tiếp theo.

Chúng tôi hiển thị các so sánh trong Hình 5, trong đó ER++ được hiển thị bằng màu xanh lá cây và các phương pháp OCL được hiển thị bằng màu đỏ. Hơn nữa, chúng tôi hiển thị ER bằng màu xanh.



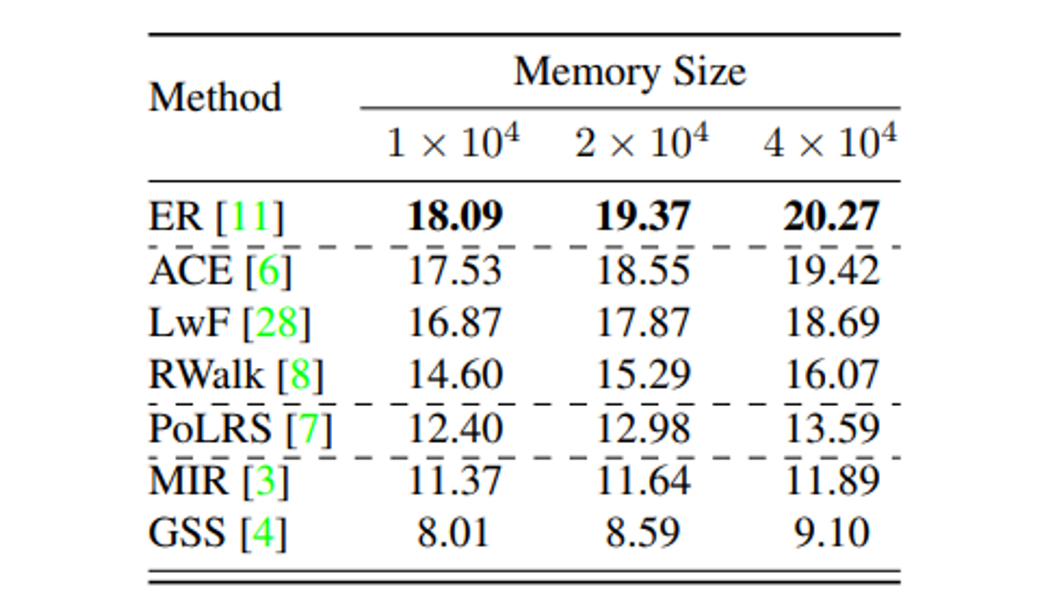
***Hình 5.*** *Slow Stream*

Trong khi RWalk vượt trội hơn ER trong cài đặt luồng chạy chậm, điều này là một so sánh không công bằng, vì ER không tận dụng được việc thực tế rằng luồng chạy chậm và nó có thể được hưởng lợi từ việc huấn luyện thêm. Khi chúng tôi chỉ đơn giản thêm một vài vòng lặp cho ER (để có thể huấn luyện thêm), tức ER++, để phù hợp với độ phức tạp huấn luyện của RWalk, ER++ thực hiện tốt hơn rất nhiều so với RWalk. Chúng tôi thấy rằng điều này nhất quán trên tất cả các phương pháp.

Thú vị là các phương pháp tính toán phức tạp, ví dụ như GSS và MIR, thậm chí thực hiện kém hơn so với ER, mà không tận dụng thực tế rằng luồng chạy chậm. ER++ tốt hơn đáng kể so với GSS và MIR, lần lượt là 17% và 11%, mở rộng khoảng cách hiệu suất so với baseline khoảng 15% và 7%. Kết quả này đặt câu hỏi về tính thích hợp của các phương pháp đã được đề xuất trước đó đối với các thiết lập OCL thực tế với tập dữ liệu quy mô lớn. Hơn nữa, kết quả của chúng tôi cho thấy một thiết lập đánh giá không công bằng về mặt tính toán (tức không chuẩn hóa độ phức tạp huấn luyện) trong OCL luồng chạy chậm có thể dẫn đến nhận định hiệu suất sai lệch, như trường hợp của RWalk. Chúng tôi nhấn mạnh rằng PoLRS ban đầu được kiểm tra trên một thiết lập hơi khác so với luồng chạy chậm, bao gồm thông tin bổ sung được đại diện bởi một album người dùng [7]. Chúng tôi nhận thấy rằng khi thông tin album được loại bỏ và cơ sở báo cáo của họ, tức ER, được điều chỉnh một cách phù hợp, PoLRS có hiệu suất tương tự với ER như được thể hiện trong Hình 5. Điều này vẫn là một điều đáng ngạc nhiên vì PoLRS tốn ba lần so với ER. Đáng ngạc nhiên hơn, khi nối độ phức tạp huấn luyện của ER với PoLRS trên luồng chạy chậm, ER++ vượt trội hơn PoLRS 9,5%. Kết quả này trong cài đặt luồng chạy chậm hỗ trợ quan sát trước đó của chúng tôi rằng các phương pháp OCL hiện tại vẫn chưa phù hợp cho việc triển khai thực tế.

## **4.3. Ảnh hưởng của kích thước bộ nhớ**

Chúng tôi tiến hành các thí nghiệm để kiểm tra xem các phương pháp hiện có có thể vượt trội hơn ER với các kích thước bộ nhớ đệm khác nhau. Chúng tôi lặp lại thí nghiệm luồng chạy nhanh từ phần 4.1 với giới hạn bộ nhớ lần lượt là , và mẫu. Kết quả được tóm tắt trong Bảng 2.



***Bảng 2.*** *Phân Tích Ngân Sách Bộ Nhớ*

Kích thước bộ nhớ lớn hơn dẫn đến độ chính xác trực tuyến trung bình cao hơn trên tất cả các phương pháp. Tuy nhiên, chúng tôi nhấn mạnh rằng xu hướng qua các phương pháp tương ứng với kết quả tìm thấy trước đó là các phương pháp OCL đắt đỏ hơn trong huấn luyện thực hiện kém hơn so với các phương pháp khác ít tốn kém hơn. Điều này rõ ràng từ thứ tự các phương pháp trong Hình 3, điều đó vẫn đúng bất kể giới hạn bộ nhớ. Đáng chú ý, ER vượt trội hơn tất cả các phương pháp được xem xét.

# **5. Kết luận**

Chúng tôi đề xuất một tiêu chuẩn đánh giá thời gian thực cho OCL, bao gồm một luồng chạy nhanh tiếp tục đưa dữ liệu mới ngay cả khi mô hình chưa hoàn thành quá trình huấn luyện. Thiết lập thực tế này buộc các phương pháp tính toán phức tạp bỏ qua nhiều mẫu huấn luyện hơn, làm nổi bật sự khác biệt về độ phức tạp huấn luyện giữa các phương pháp được xem xét. Kết quả của chúng tôi cho thấy, trong cài đặt này, tất cả các phương pháp được xem xét đều kém hơn so với cơ sở ER đơn giản. Chúng tôi cũng khám phá các tình huống trong đó chúng tôi chuẩn hóa số lượng dữ liệu huấn luyện đã nhìn thấy hoặc độ phức tạp tính toán cho mỗi bước thời gian, dẫn đến cùng kết luận rằng các phương pháp hiện tại vẫn chưa được tối ưu hóa cho triển khai thực tế.

# **Lời cảm ơn**

Công việc này được hỗ trợ bởi Đại học Khoa học và Công nghệ King Abdullah (KAUST), Văn phòng Nghiên cứu được tài trợ (OSR) dưới số giải thưởng OSR-CRG2021-4648, Trung tâm SDAIA-KAUST về Khoa học Dữ liệu và Trí tuệ Nhân tạo (SDAIA-KAUST AI), Saudi Aramco và dự án Turing AI Fellowship EP/W002981/1 do UKRI tài trợ. Chúng tôi gửi lời cảm ơn tới Học viện Hoàng gia Kỹ thuật (Royal Academy of Engineering) và FiveAI vì sự hỗ trợ của họ. Ameya Prabhu được tài trợ bởi Meta AI dưới số Grant DFR05540.

# **Tham khảo**

[1] Motasem Alfarra, Zhipeng Cai, Adel Bibi, Bernard Ghanem, and Matthias M¨uller. Simcs: Simulation for online domain-incremental continual segmentation. arXiv preprint arXiv:2211.16234, 2022. 1

[2] Rahaf Aljundi, Francesca Babiloni, Mohamed Elhoseiny, Marcus Rohrbach, and Tinne Tuytelaars. Memory aware synapses: Learning what (not) to forget. In European Conference on Computer Vision (ECCV), 2018. 1, 2

[3] Rahaf Aljundi, Lucas Caccia, Eugene Belilovsky, Mas- simo Caccia, Laurent Charlin, and Tinne Tuytelaars. Online continual learning with maximally interfered retrieval. In Conference on Neural Information Processing Systems (NeurIPS), 2019. 2, 3, 5, 8

[4] Rahaf Aljundi, Min Lin, Baptiste Goujaud, and Yoshua Bengio. Gradient based sample selection for online continual learning. In Conference on Neural Information Processing Systems (NeurIPS), 2019. 1, 2, 3, 5, 6, 8

[5] Pietro Buzzega, Matteo Boschini, Angelo Porrello, Davide Abati, and Simone Calderara. Dark experience for general continual learning: a strong, simple baseline. In Conference on Neural Information Processing Systems (NeurIPS), 2020. 3

[6] Lucas Caccia, Rahaf Aljundi, Nader Asadi, Tinne Tuytelaars, Joelle Pineau, and Eugene Belilovsky. New insights on reducing abrupt representation change in online continual learning. In International Conference on Learning Representations (ICLR), 2022. 2, 3, 5, 8, 11

[7] Zhipeng Cai, Ozan Sener, and Vladlen Koltun. Online con- tinual learning with natural distribution shifts: An empirical study with visual data. In International Conference on Computer Vision (ICCV), 2021. 2, 3, 4, 5, 8

[8] Arslan Chaudhry, Puneet K Dokania, Thalaiyasingam Ajanthan, and Philip HS Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In the European Conference on Computer Vision (ECCV), 2018. 2, 3, 5, 6, 8

[9] Arslan Chaudhry, Albert Gordo, David Lopez-Paz, Puneet K. Dokania, and Philip Torr. Using hindsight to anchor past knowledge in continual learning. In Association for the Advancement of Artificial Intelligence (AAAI), 2021. 3

[10] Arslan Chaudhry, Marc’Aurelio Ranzato, Marcus Rohrbach, and Mohamed Elhoseiny. Efficient lifelong learning with a- gem. In International Conference on Learning Representations (ICLR), 2019. 3

[11] Arslan Chaudhry, Marcus Rohrbach, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, Puneet K Dokania, Philip HS Torr, and Marc’Aurelio Ranzato. Continual learning with tiny episodic memories. In International Conference on Machine Learning (ICML), 2019. 1, 3, 5, 6, 8

[12] Aristotelis Chrysakis and Marie-Francine Moens. Online continual learning from imbalanced data. In International Conference on Machine Learning (ICML), 2020. 3

[13] Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Aleˇs Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. A continual learning survey: Defying forgetting in classification tasks. IEEE Transactions on Pattern Analysis and Machine Intelligence (IEEE TPAMI), 2021. 2

[14] Matthias De Lange and Tinne Tuytelaars. Continual prototype evolution: Learning online from non-stationary data streams. In International Conference on Computer Vision

(ICCV), 2021. 3

[15] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In Conference on Computer Vision and Pattern Recognition (CVPR), 2010. 5

[16] Chrisantha Fernando, Dylan Banarse, Charles Blundell, Yori Zwols, David Ha, Andrei A Rusu, Alexander Pritzel, and Daan Wierstra. Pathnet: Evolution channels gradient descent in super neural networks. arXiv preprint arXiv:1701.08734, 2017. 1

[17] Robert M. French. Catastrophic forgetting in connectionist networks, 1999. 1

[18] Qiankun Gao, Chen Zhao, Bernard Ghanem, and Jian Zhang. R-dfcil: Relation-guided representation learning for data-free class incremental learning. In the European Conference on Computer Vision (ECCV), 2022. 2

[19] Tyler L Hayes and Christopher Kanan. Online continual learning for embedded devices. In Conference on Lifelong Learning Agents, 2022. 3

[20] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In Conference on Computer Vision and Pattern Recognition (CVPR), 2016. 1, 5

[21] Huiyi Hu, Ang Li, Daniele Calandriello, and Dilan Gorur. One pass imagenet. In Conference on Neural Information Processing Systems Workshops (NeurIPSW), 2021. 1

[22] Wenpeng Hu, Zhou Lin, Bing Liu, Chongyang Tao, Zhengwei Tao Tao, Dongyan Zhao, Jinwen Ma, and Rui Yan. Overcoming catastrophic forgetting for continual learning via model adaptation. In International Conference on Learning Representations (ICLR), 2019. 3

[23] David Isele and Akansel Cosgun. Selective experience replay for lifelong learning. In Association for the Advancement of Artificial Intelligence (AAAI), 2018. 1

[24] Xisen Jin, Junyi Du, and Xiang Ren. Gradient based memory editing for task-free continual learning. 2021. 3

[25] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka GrabskaBarwinska, et al. Overcoming catastrophic forgetting in neural networks. In Proceedings of the National Academy of Sciences (PNAS), 2017. 1, 2

[26] Hyunseo Koh, Dahyun Kim, Jung-Woo Ha, and Jonghyun Choi. Online continual learning on class incremental blurry task configuration with anytime inference. In International Conference on Learning Representations (ICLR), 2022. 3

[27] Soochan Lee, Junsoo Ha, Dongsu Zhang, and Gunhee Kim. A neural dirichlet process mixture model for task-free continual learning. In International Conference on Learning Representations (ICLR), 2020. 3

[28] Zhizhong Li and Derek Hoiem. Learning without forgetting. IEEE Transactions on Pattern Analysis and Machine Intelligence (IEEE TPAMI), 2017. 2, 5, 8

[29] Ming Liang and Xiaolin Hu. Recurrent convolutional neural network for object recognition. In Conference on Computer Vision and Pattern Recognition (CVPR), 2015. 1

[30] Zhiqiu Lin, Jia Shi, Deepak Pathak, and Deva Ramanan. The clear benchmark: Continual learning on real-world imagery. In Conference on Neural Information Processing Systems (NeurIPS), 2021. 3

[31] Vincenzo Lomonaco, Lorenzo Pellegrini, Andrea Cossu, Antonio Carta, Gabriele Graffieti, Tyler L. Hayes, Matthias De Lange, Marc Masana, Jary Pomponi, Gido M. Van De Ven, Martin Mundt, Qi She, Keiland Cooper, Jeremy Forest, Eden Belouadah, Simone Calderara, German I. Parisi, Fabio Cuzzolin, Andreas S. Tolias, Simone Scardapane, Luca Antiga, Subutai Ahmad, Adrian Popescu, Christopher Kanan, Joost Van De Weijer, Tinne Tuytelaars, Davide Bacciu, and Davide Maltoni. Avalanche: An end-to-end library for continual learning. In Conference on Computer Vision and Pattern Recognition (CVPR), 2021. 6

[32] David Lopez-Paz and Marc’Aurelio Ranzato. Gradient episodic memory for continual learning. In Conference on Neural Information Processing Systems (NeurIPS), 2017. 1, 2, 3

[33] Zheda Mai, Ruiwen Li, Jihwan Jeong, David Quispe, Hyunwoo Kim, and Scott Sanner. Online continual learning in image classification: An empirical survey. Neurocomputing, 2022. 2, 3

[34] Arun Mallya and Svetlana Lazebnik. Packnet: Adding multiple tasks to a single network by iterative pruning. In Conference on Computer Vision and Pattern Recognition (CVPR), 2018. 3

[35] Andrea Maracani, Umberto Michieli, Marco Toldo, and Pietro Zanuttigh. Recall: Replay-based continual learning in semantic segmentation. In International Conference on Computer Vision (ICCV), 2021. 2

[36] Michael McCloskey and Neal J Cohen. Catastrophic inter- ference in connectionist networks: The sequential learning problem. In Psychology of learning and motivation. 1989. 1

[37] Ameya Prabhu, Hasan Abed Al Kader Hammoud, Puneet Dokania, Philip HS Torr, Ser-Nam Lim, Bernard Ghanem, and Adel Bibi. Computationally budgeted continual learning: What does matter? In IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2023. 3

[38] Ameya Prabhu, Philip HS Torr, and Puneet K Dokania. Gdumb: A simple approach that questions our progress in continual learning. In European Conference on Computer Vision (ECCV), 2020. 3

[39] Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H Lampert. icarl: Incremental classifier and representation learning. In Conference on Computer Vision and Pattern Recognition (CVPR), 2017. 1

[40] Andrei A Rusu, Neil C Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. arXiv preprint arXiv:1606.04671, 2016. 1, 3

[41] David Sayce. The number of tweets per day in 2022, Aug 2022. 1

[42] J¨urgen Schmidhuber. Deep learning in neural networks: An overview. Neural Networks, 2015. 1

[43] Shai Shalev-Shwartz. Online learning and online convex optimization. Foundations and trends in Machine Learning, 2011. 3, 4

[44] Dongsub Shim, Zheda Mai, Jihwan Jeong, Scott Sanner, Hyunwoo Kim, and Jongseong Jang. Online class incremental continual learning with adversarial shapley value. In Association for the Advancement of Artificial Intelligence (AAAI), 2021. 3

[45] Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. In Conference on Neural Information Processing Systems (NeurIPS), 2017. 2

[46] James Smith, Yen-Chang Hsu, Jonathan Balloch, Yilin Shen, Hongxia Jin, and Zsolt Kira. Always be dreaming: A new approach for data-free class-incremental learning. In International Conference on Computer Vision (ICCV), 2021. 2

[47] Shengyang Sun, Daniele Calandriello, Huiyi Hu, Ang Li, and Michalis Titsias. Information-theoretic online memory selection for continual learning. In International Conference on Learning Representations (ICLR), 2022. 3

[48] Gido M. van de Ven and Andreas S. Tolias. Three scenarios for continual learning. arXiv preprint arXiv:1904.07734, 2019. 3

[49] Andr´es Villa, Juan Le´on Alc´azar, Motasem Alfarra, Kumail Alhamoud, Julio Hurtado, Fabian Caba Heilbron, Alvaro Soto, and Bernard Ghanem. Pivot: Prompting for video continual learning. arXiv preprint arXiv:2212.04842, 2022. 2

[50] Andr´es Villa, Kumail Alhamoud, Victor Escorcia, Fabian Caba, Juan Le´on Alc´azar, and Bernard Ghanem. vclimb: A novel video class incremental learning benchmark. In Conference on Computer Vision and Pattern Recognition (CVPR), 2022. 2

[51] Zifeng Wang, Zizhao Zhang, Chen-Yu Lee, Han Zhang, Ruoxi Sun, Xiaoqi Ren, Guolong Su, Vincent Perot, Jennifer Dy, and Tomas Pfister. Learning to prompt for continual learning. Conference on Computer Vision and Pattern Recognition (CVPR), 2021. 1

[52] Jaehong Yoon, Divyam Madaan, Eunho Yang, and Sung Ju Hwang. Online coreset selection for rehearsal-based continual learning. In International Conference on Learning Representations (ICLR), 2022. 3

[53] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In the International Conference on Machine Learning (ICML), 2017.

# 

# 

# **Phụ lục**

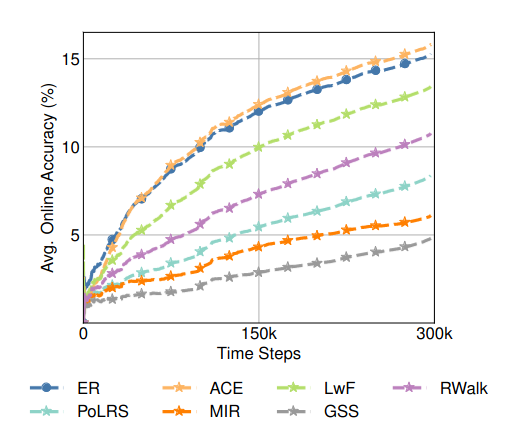
## **A. Giới hạn và công việc tương lai**

Trong bài báo chính, chúng tôi đề xuất một mô hình đánh giá mới mẻ cho việc học liên tục trực tuyến thời gian thực, nhằm mô phỏng càng gần càng tốt các tình huống thực tế. Tuy nhiên, chúng tôi nhận thấy có một số hạn chế mà có thể được giải quyết trong tương lai. Mặc dù ER và ER++ đã đạt được tỷ lệ Chính xác Trực tuyến Trung bình (Average Online Accuracy) cao nhất so với tất cả các phương pháp khác được xem xét, nhưng hiệu suất của họ chỉ đạt được tối đa là 20% và 35% tương ứng. Hiệu suất thấp này cho thấy còn tiềm năng để cải thiện với các phương pháp Học liên tục trực tuyến (OCL) tốt hơn. Hơn nữa, ER có thể có lợi thế trong thiết lập này do sự tương quan thời gian tiềm năng trong tập dữ liệu CLOC. Một giải pháp có thể là thiết kế các tập dữ liệu Học liên tục trực tuyến (OCL) mới mà có ít phụ thuộc thời gian hơn. Hơn nữa, trong khi ER đã thể hiện sự thích nghi trực tuyến tốt nhất so với các phương pháp khác, nó lại thể hiện sự kém hơn trong việc duy trì thông tin (xem sự truyền dữ liệu ngược trong Appendix D). Công việc trong tương lai có thể khám phá các phương pháp Học liên tục trực tuyến (OCL) có thể tối ưu hóa cả việc thích nghi trực tuyến và việc duy trì thông tin.

## **B. Học liên tục trực tuyến không qua tiền đào tạo**

Trong các cài đặt thế giới thực, việc học liên tục diễn ra sau khi một mô hình đã được chuẩn bị trước trên dữ liệu cục bộ đã được ghi chú; như đã báo cáo trong tất cả các thí nghiệm trước đây trong bài báo chính. Tuy nhiên, chúng tôi xem xét trong phần này một thiết lập trong đó các mô hình được đào tạo liên tục mà không cần chuẩn bị trước. Chúng tôi lặp lại các thí nghiệm trên tập dữ liệu CLOC trong khi đào tạo nguồn cơ sở ResNet50 từ đầu. Chúng tôi thấy rằng trong thiết lập này, tốc độ học tập 5 × 10^(-2) hoạt động tốt nhất cho tất cả các phương pháp ngoại trừ PoLRS, mà hoạt động tốt hơn với tốc độ học tập 10^(-2).

Theo nhóm đánh giá mà chúng tôi sử dụng trong bài chính, chúng tôi trình bày kết quả trong ba hình ảnh: Fast Stream (Hình 1), Fast Stream với Chuẩn hóa Dữ liệu (Hình 2), và Slow Stream (Hình 3). Chúng tôi nhận thấy rằng kết quả mà không có chuẩn bị trước cho thấy một kết luận tương tự với bài báo chính, trong đó giới hạn cơ bản (ER) hoạt động hiệu quả hơn so với các phương pháp cụ thể của Học liên tục trực tuyến (OCL) đắt tiền hơn. Đáng chú ý, ACE và ER hiệu quả như nhau, và cả hai luôn hoạt động hiệu quả hơn các phương pháp đắt tiền hơn. Thú vị, ACE chỉ vượt qua ER một cách rất nhỏ mặc dù ER không sử dụng bất kỳ kỹ thuật Học liên tục trực tuyến cụ thể nào.



***Hình 1.*** *Đánh giá Dòng Nhanh (Không cần tiền đào tạo). Tương tự với các thí nghiệm trong bản báo chính, các phương pháp hiệu quả nhất, ER và ACE, vượt trội đáng kể so với các phương pháp khác được xem xét, ngay cả khi backbone được đào tạo từ đầu.*

## **C. Thí nghiệm quy mô nhỏ**

Trong phần này, chúng tôi kiểm tra xem các phương pháp trước đó hoạt động tốt trong đánh giá liên tục thời gian thực nếu tập dữ liệu được sử dụng quá nhỏ và không thực tế. Để làm điều này, chúng tôi lặp lại các thí nghiệm Fast Stream và Slow Stream trên tập dữ liệu CIFAR10 và CIFAR100, mà nhiều phương pháp được xem xét ban đầu đã được thử nghiệm. Chúng tôi xáo trộn ngẫu nhiên các tập dữ liệu này mà không có ranh giới nhiệm vụ, tạo thành một dòng trực tuyến và đào tạo tuần tự các mô hình trên dòng trực tuyến trong một lần chạy.

Chi tiết Thực hiện. Đối với tất cả các thí nghiệm trên CIFAR10 và CIFAR100, dòng trực tuyến hiển thị 10 hình ảnh cho mỗi bước thời gian, mà được bổ sung thêm 10 hình ảnh được lấy mẫu từ bộ nhớ. Quy trình này tạo ra một nhóm con đào tạo có 20 mẫu cho mỗi bước thời gian. Hơn nữa, chúng tôi đặt kích thước bộ nhớ là 100. Tất cả các mô hình sử dụng một ResNet18 thu gọn làm cơ sở [6], trong đó chúng tôi sử dụng SGD để tối ưu hóa mô hình. Đối với việc lựa chọn tốc độ học, chúng tôi thực hiện kiểm định chéo trên 5% dữ liệu được giữ lại của mỗi tập dữ liệu. Trong CIFAR10, chúng tôi tìm ra các giá trị tốc độ học sau đây hoạt động tốt nhất:

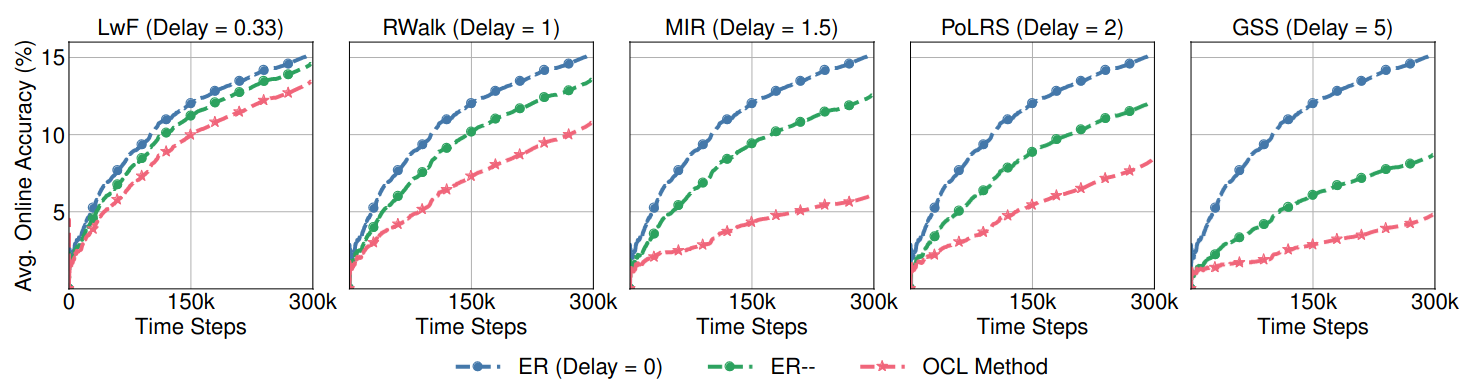
* cho ER, ACE, LwF và RWalk
* 5 x cho PoLRS, MIR và GSS

Còn đối với CIFAR100, chúng tôi sử dụng các giá trị tốc độ học sau đây:

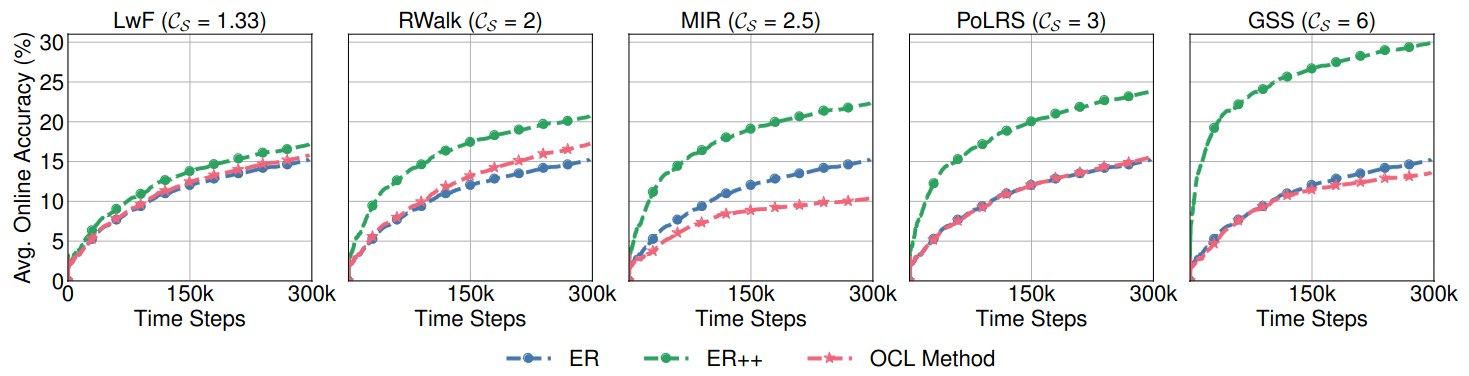
* cho ER, ACE, PoLRS
* 5 x cho LwF, RWalk, GSS
* cho MIR

### **C.1. Luồng nhanh**

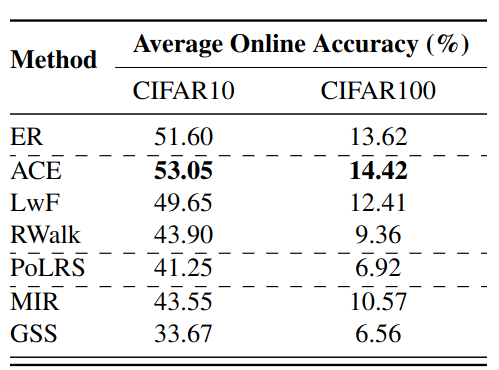
Ở phần trước, để dễ so sánh, chúng ta giả định rằng ER có một độ phức tạp tính toán phù hợp với tốc độ dòng trực tuyến và do đó có khả năng đào tạo trên tất cả các mẫu đến trong dòng nhanh. Kết quả là, các phương pháp đắt tiền hơn ER bị bị tụt lại so với dòng. Sự trễ này buộc các phương pháp này phải bỏ qua một số mẫu đến, nơi họ thực hiện việc đào tạo một phần của các mẫu trong dòng. Trong Bảng 1, chúng tôi báo cáo hiệu suất của các phương pháp Học liên tục trực tuyến được xem xét khi được đánh giá trên các tập dữ liệu CIFAR10 và CIFAR100 trong bối cảnh dòng nhanh. Chúng tôi nhận thấy rằng ngay cả khi sử dụng các tập dữ liệu quy mô nhỏ, ACE và ER vượt trội hơn tất cả các phương pháp khác được xem xét trong thiết lập dòng nhanh thực tế. Đáng chú ý, hiệu suất của ACE chỉ cao hơn ER một cách rất nhỏ, ít hơn 1.5% trên cả hai tập dữ liệu, mặc dù ER không thực hiện bất kỳ điều gì cụ thể cho việc học liên tục.



***Hình 2.*** *Dòng Nhanh - Chuẩn hóa Dữ liệu Đào tạo (Không cần tiền đào tạo). Chúng tôi so sánh mỗi phương pháp với ER và phiên bản có trễ ER--. ER-- thực hiện các bước gradient bổ sung cho mỗi bước thời gian để phù hợp với sự trễ của phương pháp được so sánh, vì vậy nó đào tạo trên ít mẫu hơn ER. Chúng tôi muốn nhấn mạnh rằng ngay cả khi không sử dụng tiền đào tạo, ER-- vẫn vượt trội hơn tất cả các phương pháp được xem xét.*



***Hình 3.*** *Đánh giá Dòng Chậm (Không cần tiền đào tạo). Chúng tôi so sánh mỗi phương pháp với ER và ER++, mà thực hiện thêm các bước gradient cho mỗi bước thời gian để phù hợp với độ phức tạp CS của phương pháp được so sánh. ER++ vượt trội hơn tất cả các phương pháp được xem xét, ngay cả khi backbone không được tiền đào tạo.*

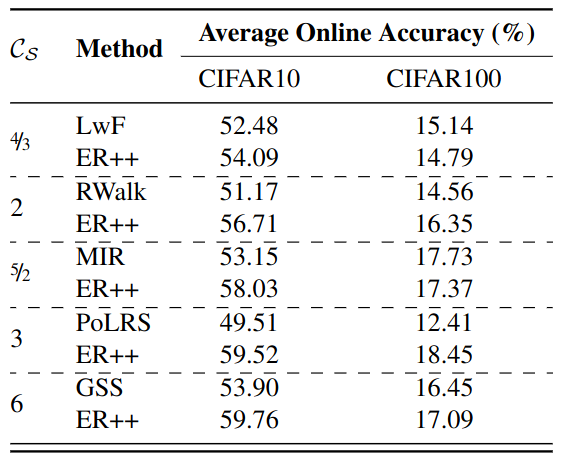


***Bảng 1.*** *Thí nghiệm Quy Mô Nhỏ - Dòng Nhanh. Chúng tôi lặp lại các thí nghiệm dòng nhanh trên các bộ dữ liệu quy mô nhỏ CIFAR10 và CIFAR100. Chúng tôi nhận thấy rằng bất kể kích thước của bộ dữ liệu, ACE và ER đều vượt trội hơn tất cả các phương pháp được xem xét.*

### **C.2. Luồng chậm**

Trong thiết lập này, tương tự như trong bài chính, dòng chậm chậm bằng với các phương pháp Học liên tục trực tuyến đắt tiền hơn. Nghĩa là, các phương pháp trong dòng này có khả năng đào tạo trên tất cả các mẫu trong dòng. Hãy nhớ rằng vì các phương pháp có độ phức tạp đào tạo khác nhau trong dòng này, chúng tôi sử dụng mô hình cơ sở đã điều chỉnh ER++ để có một so sánh công bằng. ER++ phù hợp với độ phức tạp của mỗi phương pháp được so sánh bằng cách thực hiện các cập nhật gradient bổ sung.

Chúng tôi trình bày kết quả của đánh giá dòng chậm trong Bảng 2 khi xem xét các tập dữ liệu quy mô nhỏ như CIFAR10 và CIFAR100. Tương tự như kết quả của dòng chậm trên tập dữ liệu CLOC, ER++ vượt trội hơn tất cả các phương pháp Học liên tục trực tuyến được xem xét khi được đánh giá trên CIFAR10. Đối với CIFAR100, ER++ cũng vượt trội hơn tất cả các phương pháp được xem xét ngoại trừ LwF và MIR, mà chỉ cao hơn ER++ một cách nhẹ, không nhiều hơn 0.36%.



***Bảng 2.*** *Thí nghiệm Quy Mô Nhỏ - Dòng Chậm. Chúng tôi lặp lại các thí nghiệm dòng chậm trên các bộ dữ liệu quy mô nhỏ CIFAR10 và CIFAR100. Chúng tôi lưu ý rằng trong hầu hết các trường hợp, ER++ vượt trội hơn phương pháp được so sánh. Các trường hợp ngoại lệ duy nhất là LwF và MIR, và chỉ khi được đánh giá trên CIFAR100. Ngay cả khi đó, chúng có sự tăng trưởng không đáng kể so với phiên bản cơ sở ER++ tương ứng của chúng, điều này cho thấy cần phải phát triển các phương pháp tốt hơn và hiệu quả hơn cho việc học liên tục trực tuyến.*

## **D. Hiệu suất trên mẫu giữ lại**

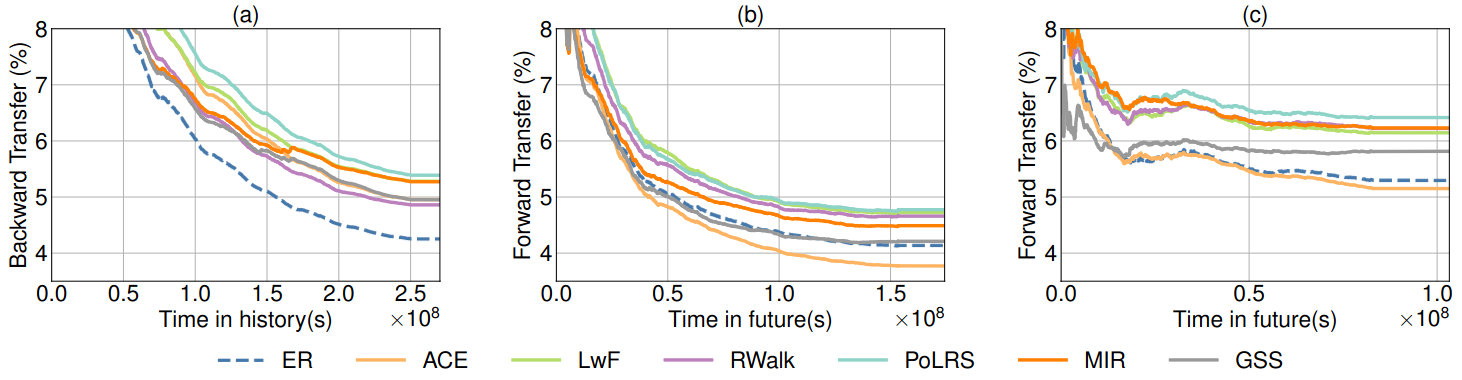
Trong bài báo chính, chúng tôi tập trung vào Độ chính xác Trực tuyến Trung bình như độ đo chính yếu vì nó đo lường sự thích nghi với lô huấn luyện tiếp theo (tức là phân phối gần tương lai), đây là một thuộc tính rất mong muốn trong các dòng thời gian thời gian thực với sự thay đổi phân phối nhanh chóng. Trong phần này, chúng tôi xem xét hai độ đo đánh giá khác cho việc học liên tục, đó là độ chuyển tiến ngược và chuyển tiến xuôi. Độ đo chuyển tiến ngược đánh giá việc quên mất về các phân phối trong quá khứ, trong khi chuyển tiến xuôi đánh giá sự thích nghi của các mô hình với các phân phối tương lai trên các mẫu được giữ lại. Quan trọng là lưu ý rằng chuyển tiến xuôi khác biệt với Độ chính xác Trực tuyến Trung bình ở chỗ nó đánh giá hiệu suất trên các mẫu được giữ lại thay vì các mẫu huấn luyện chưa từng được thấy trong tương lai. Hơn nữa, nó theo dõi hiệu suất trên các phân phối trong tương lai, thay vì chỉ tập trung vào phân phối của lô tiếp theo.

Để đo độ chuyển tiến ngược và chuyển tiến xuôi, chúng tôi lấy mẫu đồng nhất và giữ lại 1% tập dữ liệu CLOC từ các bước thời gian khác nhau. Độ chuyển tiến ngược của mô hình cuối cùng ở cuối các bước trong dòng thời gian được đo bằng cách tính trung bình độ chính xác trên các mẫu chưa từng thấy từ các thời điểm trong quá khứ ở từng bước thời gian. Tương tự, chuyển tiến xuôi được đánh giá bằng cách tính trung bình độ chính xác trên các mẫu chưa từng thấy bắt đầu từ thời điểm mà mô hình được thu thập đến cuối dòng thời gian. Theo sau CLOC, chúng tôi đo độ chuyển tiến xuôi cho mô hình được lưu lại ở 1/3 và 2/3 của dòng thời gian.

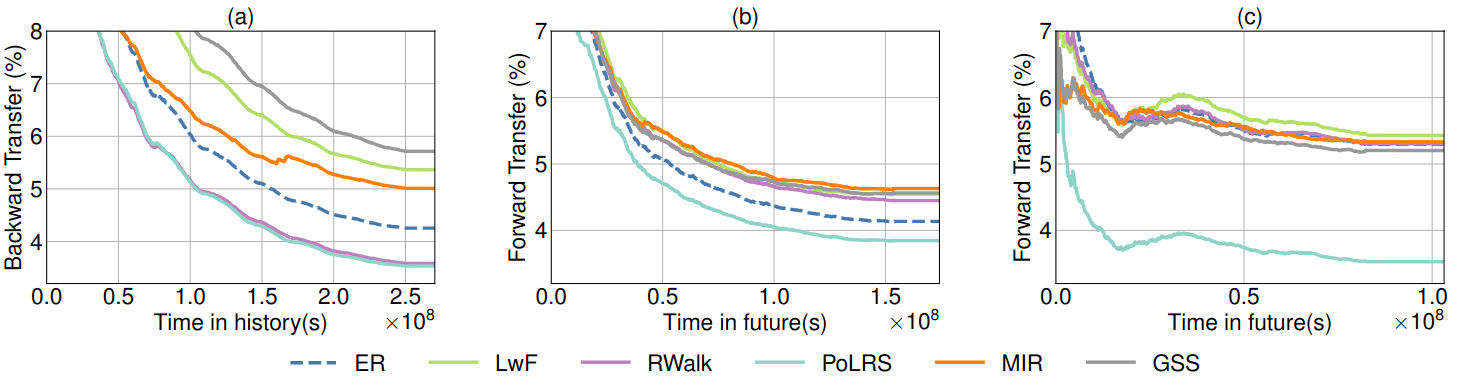
Hình 4 và 5 cho thấy kết quả về độ chuyển tiến ngược và chuyển tiến xuôi trong các dòng thời gian chậm và nhanh, tương ứng. Chúng tôi quan sát rằng tất cả các phương pháp OCL thực hiện tương tự như phương pháp ER, chỉ có một ưu điểm nhỏ đối với một số phương pháp. Ví dụ, trong dòng thời gian chậm, mặc dù GSS và MIR đòi hỏi 6 và 2,5 lần chi phí tính toán so với ER, tương ứng, họ chỉ vượt trội ER ở độ chuyển tiến ngược 1,5% và 0,7%. Tương tự, trong dòng thời gian nhanh, họ vượt trội ER ở độ chuyển tiến xuôi (ở 1/3 của dòng thời gian) chỉ 0,4% và 0,5%. Trong kết quả dòng thời gian nhanh được thể hiện trong Hình 4, MIR và GSS cũng đặc biệt vượt trội ER, mặc dù họ gặp phải một số sự trễ trong quá trình đào tạo, cho thấy họ có một ưu điểm nhỏ trong các độ đo chuyển tiến xuôi và ngược mà họ đã được tối ưu hóa. Quan trọng là sự tăng nhẹ về hiệu suất trong GSS và MIR đi kèm với chi phí thấp về Độ chính xác Trực tuyến Trung bình, điều này quan trọng trong việc học liên tục thời gian thực. Khi xem xét Độ chính xác Trực tuyến Trung bình cả trong dòng thời gian chậm và nhanh, vai trò của hiệu suất đào tạo trở nên rõ ràng.

Có điều đáng lưu ý là độ đo đánh giá độ chuyển tiến ngược khác biệt một chút so với mục tiêu của việc học liên tục trực tuyến thời gian thực. Trong việc học liên tục ngoại tuyến, thường thì các phương pháp được đánh giá vào cuối dòng trên một tập kiểm tra được giữ lại. Tuy nhiên, cài đặt đánh giá này không liên quan đến việc học liên tục trực tuyến thời gian thực, nơi mà các dòng thường xuyên trải qua sự biến đổi phân phối nhanh chóng. Do động của các dòng này, hiệu suất trên các mẫu từ các phân phối trong quá khứ không phải là chỉ báo tốt về việc các mô hình sẽ đưa ra dự đoán đúng cho các mẫu mới đang đến. Điều này bởi vì trong việc học liên tục trực tuyến với một dòng tự nhiên, chỉ các mẫu hiện tại trong dòng mới quan trọng cho quá trình đánh giá. Ví dụ, hãy xem xét một dòng tự nhiên phát hiện hình ảnh của thiết bị điện tử. Dòng này ít có khả năng hiển thị hình ảnh của các thiết bị năm 1995 so với hình ảnh của thiết bị từ năm 2022. Đây chính là động lực cổ điển đằng sau việc học liên tục trực tuyến từ một dòng tự nhiên so với việc học liên tục ngoại tuyến.

Còn về độ chuyển tiến xuôi, chúng tôi lưu ý rằng nó có thể không phải là chỉ báo chính xác về khả năng dự đoán của mô hình đối với các mẫu tương lai, đặc biệt trong các dòng có sự thay đổi nhanh chóng trong phân phối. Vì chuyển tiến xuôi được tính toán dưới dạng trung bình chạy của độ chính xác trên các phân phối tương lai, nó gán trọng số bằng nhau cho tất cả các mẫu tương lai. Tuy nhiên, điều này có thể gây hiểu nhầm trong các dòng động, thời gian thực nơi mà phân phối gần tương lai có khả năng được quan sát hơn so với phân phối ở tương lai xa. Nói cách khác, một mô hình hoạt động tốt trên chuyển tiến xuôi có thể vẫn gặp khó khăn trong việc thích nghi với phân phối gần tương lai. Do đó, trong khi chuyển tiến xuôi có thể là một chỉ báo hữu ích trong việc đánh giá khả năng thích nghi của mô hình với các phân phối tương lai, nó nên được hiểu cẩn thận trong các dòng thời gian thời gian thực và phải được bổ sung bằng các độ đo đánh giá khác tập trung vào phân phối gần tương lai, chẳng hạn như Độ chính xác Trực tuyến Trung bình (Average Online Accuracy).



***Hình 4.*** *Hiệu suất trên Mẫu giữ lại - Dòng Nhanh. (a) Chuyển tiếp ngược cho một mô hình được lấy vào cuối dòng. (b, c) Chuyển tiếp xuôi cho mô hình được lấy vào 1/3 và 2/3 của dòng tương ứng. Chúng tôi lưu ý rằng tất cả các phương pháp có hiệu suất tương tự với ER trong cả các chỉ số chuyển tiếp ngược và chuyển tiếp xuôi với sự khác biệt tối đa là 1.1% trên tất cả các biểu đồ.*



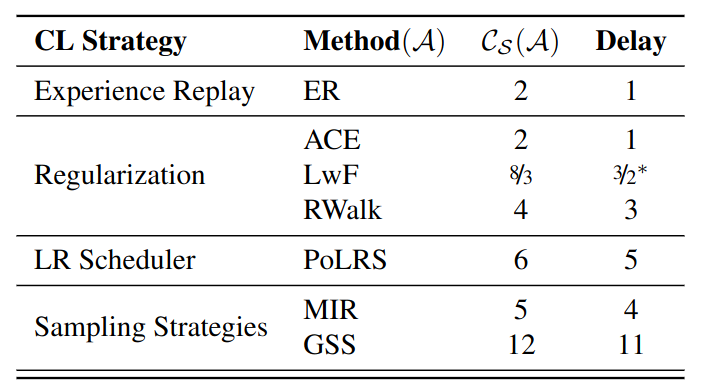
***Hình 5.*** *Hiệu suất trên Mẫu giữ lại - Dòng Chậm. (a) Chuyển tiếp ngược cho mô hình được lấy vào cuối dòng. (b, c) Chuyển tiếp xuôi cho mô hình được lấy vào 1/3 và 2/3 của dòng tương ứng. Chúng tôi quan sát rằng tất cả các phương pháp có hiệu suất tương tự với ER cả trong chỉ số chuyển tiếp ngược và chuyển tiếp xuôi. Ngay cả các phương pháp đòi hỏi nhiều tính toán nhất, cụ thể là GSS và MIR, chỉ vượt trội hơn ER trong chuyển tiếp ngược/chuyển tiếp xuôi tối đa 1.5%, đó là các chỉ số mà chúng đã được tối ưu hóa. Tuy nhiên, lợi thế nhỏ này đến với sự giảm điểm Average Online Accuracy nhiều. Lưu ý rằng ACE không được bao gồm trong các biểu đồ vì nó có độ phức tạp đào tạo là 1 và do đó không được đánh giá trong dòng chậm.*

## **E. Một luồng nhanh hơn**

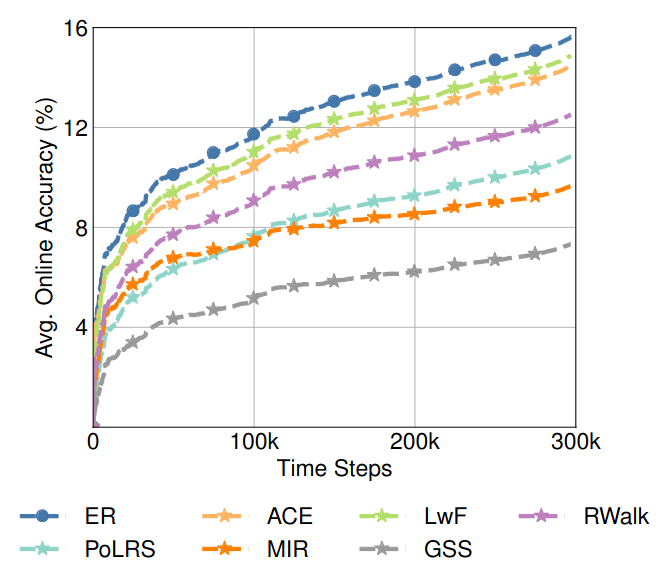
Trong các thí nghiệm dòng nhanh được báo cáo trong bản báo chí chính, chúng tôi đã giả định rằng ER chạy với tốc độ bằng với dòng để dễ dàng so sánh. Kết quả là, ER có độ phức tạp tương đối của mô hình dòng là , cho phép nó huấn luyện trên toàn bộ dòng dữ liệu. Ngược lại, các phương pháp có độ phức tạp tính toán cao hơn so với ER có độ phức tạp tương đối của mô hình dòng lớn hơn 1, dẫn đến một số sự trễ trong quá trình đào tạo.

Có thể có lý do rằng cài đặt của chúng tôi đã mang lại lợi thế cho ER bằng cách chuẩn hóa tốc độ dòng cho nó. Trong phần này, chúng tôi khám phá một cài đặt trong đó dòng chạy nhanh hơn so với tất cả các phương pháp được đánh giá.

Chúng tôi lặp lại các thí nghiệm với dòng nhanh nhưng với tốc độ gấp đôi so với ER. Dưới tốc độ dòng mới này, chúng tôi trình bày trong Bảng 3 độ phức tạp đào tạo và sự trễ tương ứng cho mỗi phương pháp được xem xét. Trong Hình 6, chúng tôi báo cáo hiệu suất của các phương pháp OCL được xem xét trong dòng nhanh. Ngay cả khi dòng nhanh hơn ER, chúng tôi quan sát các kết quả tương tự như trong bản báo chính. Đặc biệt, ER vẫn vượt trội so với tất cả các phương pháp được xem xét.



***Bảng 3.*** *Đánh giá Dòng Nhanh Hơn - Độ phức tạp đào tạo và sự trễ của các phương pháp OCL được xem xét khi tốc độ của dòng nhanh gấp đôi so với ER. ∗Lưu ý rằng chúng tôi đã làm tròn độ phức tạp của LwF từ 1.67 xuống 1.5 để dễ dàng thực hiện.*

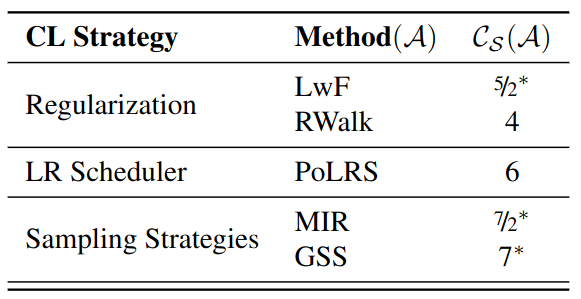


***Hình 6.*** *Đánh giá Dòng Nhanh Hơn. Chúng tôi đặt tốc độ của dòng gấp đôi so với ER. Ngay cả khi tốc độ dòng nhanh hơn ER, ER vẫn vượt trội hơn tất cả các phương pháp được xem xét, tương tự như các thí nghiệm trong bản báo chính.*

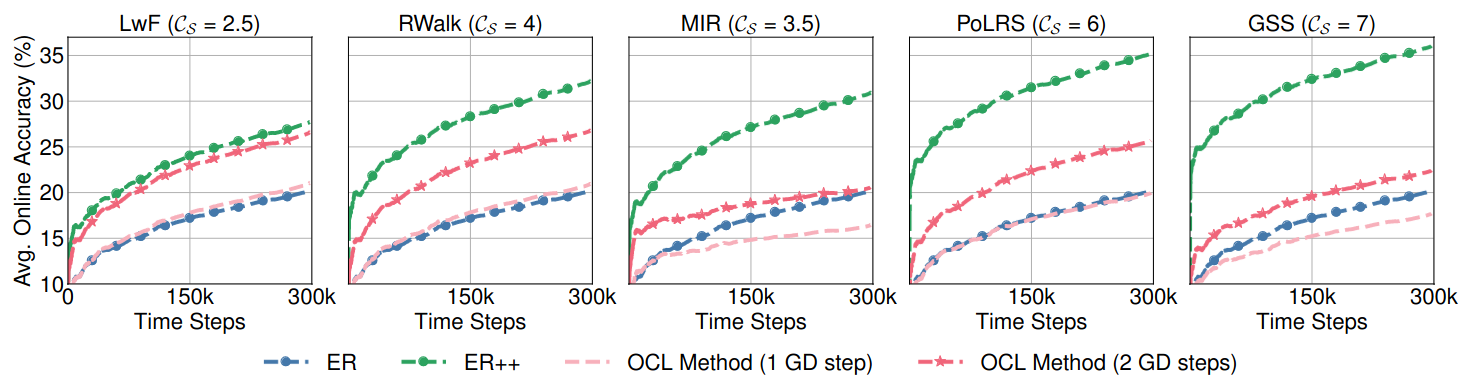
## **F. Một luồng chậm hơn**

Trong các thí nghiệm dòng chậm được trình bày trong bản báo chính, chúng tôi đã chứng minh rằng ER++ vượt trội hơn tất cả các phương pháp OCL so sánh. Những kết quả này đặt ra câu hỏi sau đây: "Vậy nếu dòng chậm lại đến mức mà các phương pháp đắt tiền tính toán hơn có thể thực hiện thêm các bước gradient?" Để khám phá câu hỏi này, chúng tôi cho phép mỗi phương pháp thực hiện hai bước Gradient Descent (GD) cho mỗi bước thời gian. Tương ứng, chúng tôi điều chỉnh ER++ để phù hợp với độ phức tạp của phiên bản tăng cường của mỗi phương pháp được hiển thị trong Bảng 4. Lưu ý rằng việc thực hiện hai bước GD trên LwF, RWalk và PoLRS dẫn đến việc gia đình độ phức tạp ban đầu của chúng. Ngược lại, việc thêm một bước GD vào MIR và GSS chỉ tăng độ phức tạp của chúng lên một. Điều này là do LwF, RWalk và PoLRS yêu cầu thêm các bước lan truyền ngược và/hoặc lan truyền xuôi với mỗi bước GD, trong khi các phương pháp dựa trên mẫu lấy, MIR và GSS, chỉ cần chúng với dữ liệu mới đến.

Trong Hình 7, chúng tôi báo cáo hiệu suất của mỗi phương pháp OCL với một và hai bước GD cho mỗi bước thời gian, trong đó kết quả cho một bước GD được hiển thị chỉ để tham khảo. Chúng tôi so sánh phiên bản tăng cường của mỗi phương pháp với ER và phiên bản tương ứng của ER++. Tương tự với kết quả dòng chậm được trình bày trong bản báo chính, ER++ vượt trội hơn tất cả các phương pháp OCL được xem xét. So với các phương pháp khác, LwF có khoảng cách độ chính xác nhỏ nhất so với ER++, gợi ý rằng đây là một phương pháp rất hiệu quả. Mặc dù việc thêm một bước GD vào GSS và MIR đã làm giảm khoảng cách so với ER++ từ 17% xuống còn 14% và từ 11% xuống còn 10%, tương ứng, nhưng cả hai phương pháp vẫn thua kém LwF, một phương pháp ít tốn kém tính toán hơn. Tóm lại, kết quả dòng chậm cho thấy rằng ER++ vẫn là sự lựa chọn tối ưu để đạt được Độ chính xác Trực tuyến Trung bình cao nhất trong khi sử dụng tài nguyên tính toán hiệu quả.



**Bảng 4.** Đánh giá Dòng Chậm Hơn - Độ phức tạp đào tạo của các phương pháp OCL được xem xét khi mỗi phương pháp thực hiện hai bước Gradient Descent cho mỗi bước thời gian. ∗Lưu ý rằng độ phức tạp của các phương pháp này đã được làm tròn xuống để dễ thực hiện.



***Hình 7.*** *Đánh giá Dòng Chậm Hơn. Chúng tôi cho phép mỗi phương pháp OCL thực hiện hai bước Gradient Descent cho mỗi bước thời gian. Chúng tôi so sánh mỗi phương pháp với ER và ER++, mà thực hiện thêm các bước GD bổ sung cho mỗi bước thời gian để phù hợp với độ phức tạp CS của phương pháp được so sánh. Và để tham khảo, chúng tôi cũng hiển thị hiệu suất của mỗi phương pháp OCL với chỉ một bước GD. Ngay cả khi các phương pháp OCL thực hiện thêm các bước GD, ER++ vẫn vượt trội hơn hết.*

## **G. Tìm kiếm trên lưới tốc độ học**

Để xác định tốc độ học tốt nhất cho các phương pháp trong các dòng chậm và nhanh được thảo luận trong bản báo chính, chúng tôi tiến hành tìm kiếm trên một lưới với 4 giá trị: {0.001, 0.005, 0.01, 0.05}, như chúng tôi báo cáo trong Bảng 5. Chúng tôi lưu ý rằng trong thực tế, tốc độ của dòng thường không được biết trước trong quá trình đào tạo. Vì sự trễ trong việc đào tạo được xác định bởi tốc độ của dòng không biết trước, chúng tôi tiến hành tìm kiếm trên lưới với giả định rằng không có sự trễ đào tạo cho tất cả các phương pháp.



***Bảng 5.*** *Tìm kiếm trên Lưới Tốc Độ Học. Chúng tôi thực hiện kiểm tra chéo trên mỗi phương pháp OCL được xem xét, kiểm tra chúng trên bốn giá trị tốc độ học khác nhau. Độ chính xác Trực tuyến Trung bình của giá trị tốc độ học đã được chọn được đánh dấu bằng chữ đậm cho mỗi phương pháp.*