# Lý do chọn đề tài

Tin sinh học là một lĩnh vực đang phát triển mạnh mẽ và có tính ứng dụng cao trong cuộc sống, đặc biệt là trong lĩnh vực công nghệ sinh học, nông nghiệp và y-dược. Đây là một lĩnh vực khoa học liên ngành, trong đó sinh học và tin học đóng vai trò chủ đạo. Về cơ bản, tin sinh học tập trung vào nghiên cứu, phát triển và áp dụng các phương pháp và công cụ tin học để giải quyết các bài toán trong sinh học.

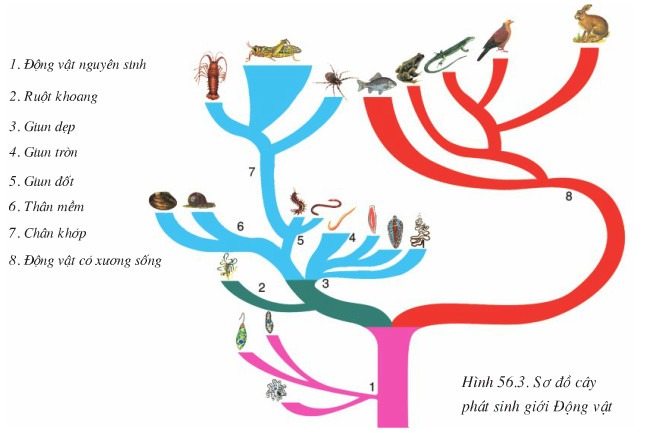
Cây tiến hóa hay cây phát sinh loài hay cây phân loài là chìa khóa giải quyết hầu hết các bài toán Tin sinh học, vì vậy việc giải quyết bài toán xây dựng cây phân loài là công việc vô cùng quan trọng, ảnh hưởng trực tiếp tới kết quả của nhiều nghiên cứu, ứng dụng lớn trên thế giới. Trong tin sinh học cổ điển, có bốn phương pháp sử dụng để xây dựng cây phân loài tương ứng với các tiêu chuẩn:

1. Phương pháp dựa vào khoảng cách
2. Phương pháp cực tiểu hóa biến đổi (Maximum Parsimony)
3. Phương pháp cực đại hóa khả năng (Maximum Likelihood)
4. Phương pháp Bayesian dựa trên MCMC

# Bài toán

Theo học thuyết tiến hóa của Darwin, mọi loài sinh vật đều tiến hóa từ một tổ tiên chung. Mối quan hệ giữa các loài được biểu diễn được biểu diễn bằng một cấu trúc cây được gọi là cây phân loài.

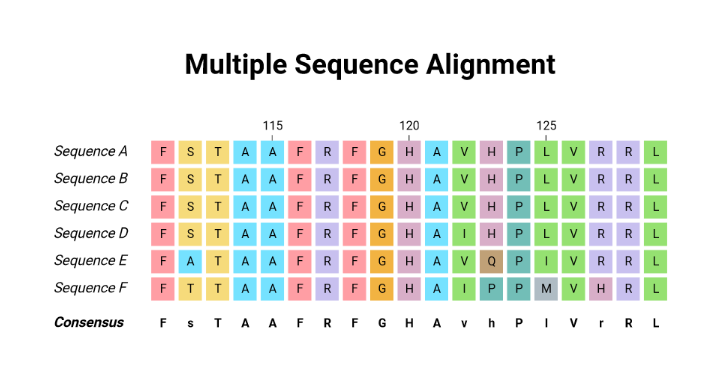
Trong cây phân loài, mỗi nút lá biểu diễn cho một loài sinh vật hiện tại, mỗi nút cha đại diện cho tổ tiên gần nhất của các sinh vật ở nút con. Độ dài cạnh có thể được hiểu như là ước lượng khoảng cách tiến hóa giữa các loài, có thể là thời gian, hay số lượng biến đổi nucleotide giữa hai trình tự DNA/amino acid giữa các loài.



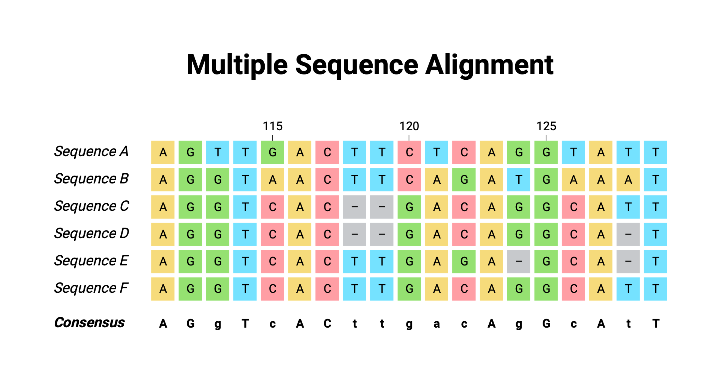
*Cây phân loài (Nguồn: SGK Sinh học lớp 7)*

Để giải quyết bài toán trên, trước hết ta cần đi đến khái niệm sắp hàng đa trình tự. Sắp hàng đa trình tự là quá trình chèn thêm các kí tự '-' (cho biết một nucleotide đã bị xóa khỏi trình tự) vào các trình tự DNA để sau khi sắp hàng chúng ta thu được các trình tự DNA thỏa mãn các điều kiện ràng buộc sau:

1. Các trình tự sau khi sắp hàng có độ dài bằng nhau và độ dài đó được gọi là độ dài của đa sắp hàng.
2. Các nucleotide ở cùng một vị trí trên các trình tự được cho tương đồng, tức là cùng tiến hóa từ một nucleotide tổ tiên chung.
3. Không tồn tại bất cứ một vị trí i nào mà tất cả các trình tự cùng chứa kí tự '-'. Nói một cách khác, việc chèn kí tự '-' vào cùng một vị trí trên tất cả các trình tự là không có ý nghĩa.

**

*Sắp hàng đa trình tự với mỗi hàng là một chuỗi Protein*

**

*Sắp hàng đa trình tự với mỗi hàng là một chuỗi DNA*

Khi đó, bài toán xây dựng cây phân loài có thể phát biểu ngắn gọn như sau:

|  |
| --- |
| **Dữ liệu vào:** Một đa sắp hàng A = (X1,..., Xn) gồm n trình tự DNA/protein của n sinh vật. Sinh vật thứ i được biểu diễn bởi trình tự Xi.  **Dữ liệu ra:** Một cây nhị phân không gốc biểu diễn mỗi quan hệ của n sinh vật. Mỗi sinh vật được biểu diễn ở một nút lá của cây. Độ dài các cạnh của cây biểu diễn số lượng biến đổi nucleotide/amino acid giữa các nút của cây. |

TODO: Viết thêm về Newick format:

|  |  |  |
| --- | --- | --- |
| Thành phần | Ký hiệu | Ví dụ |
| Gộp nhánh | ( ) | (A,B) |
| Phân cách | , | A,B |
| Độ dài nhánh | : | A:2 |
| Kết thúc | ; | (...); |

# Thuật toán cổ điển Quartet Puzzling

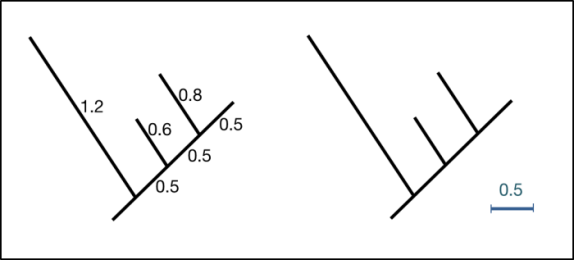
* Định nghĩa Quartet phylogenetic tree (gọi tắt là quartet): là cây phân loài không gốc gồm 4 trình tự. Như vậy, chỉ có 3 cách xây dựng cây từ 4 trình tự này: ((A, B), C, D), ((A, C), B, D), ((A, D), B, C)
* Giải thuật quartet puzzling (QP) của Strimmer & von Haeseler đề xuất năm 1996 (<https://doi.org/10.1093/oxfordjournals.molbev.a025664>):
* **Bước 1:** Gọi ML(A, B, C, D) là cây có giá trị maximum likelihoodcủa bộ bốn trình tự (A, B, C, D).
* **Bước 2: Puzzling (ghép cây):**
  1. Xáo trộn ngẫu nhiên thứ tự n trình tự, WLOG: A,B,C,D,E,…
  2. Khởi tạo cây con bằng cây bốn trình tự (A,B,C,D) từ bước 1.
  3. Lần lượt thêm các trình tự tiếp theo E bằng cơ chế “bỏ phiếu”: với mỗi bộ 4 trình tự (i,j,k,E), giả sử ML(i,j,k,E) = ((i,j),k,E) thì E sẽ bị “cấm” gắn lên các cạnh nằm trên đường đi giữa i và j. Mỗi cạnh được đánh dấu tương ứng, rồi E được chèn vào cạnh có số lần bị cấm thấp nhất (cạnh cùng giá trị sẽ được chọn ngẫu nhiên) .
  4. Tiếp tục cho đến khi cây đầy đủ n trình tự.
* **Bước 3: Xây dựng cây đồng thuận (consensus tree)**
  1. Thực hiện Puzzlingvới nhiều lần xáo (≥ 1000) để thu về tập hợp các cây trung gian.
  2. Tính cây đồng thuận majority-rule từ tập hợp này: chỉ giữ các nhánh xuất hiện ở > 50% số cây trung gian, đồng thời ghi lại độ tin cậy (tần suất xuất hiện) của mỗi nhánh. Ví dụ:{A,B} | {C,D,E} xuất hiện trong 620/1000 cây thì giữ lại nhánh này.

# Dữ liệu

* Sử dụng dữ liệu nucleotide.

## Dữ liệu mô phỏng

* Mô phỏng dựa trên độ dài nhánh (branch length) của một số dữ liệu đã công bố (chi tiết dữ liệu công bố ở Figure S2). Độ dài nhánh là số đột biến trung bình / vị trí (site) so với nút gốc.



* So sánh giữa các phân bố độ dài nhánh sử dụng Jensen–Shannon divergence (JSD). JSD càng cao thì hai phân bố càng khác biệt. 🡪 Giúp ta so sánh dữ liệu mô phỏng so với thực tế

Quy trình chi tiết để sinh các cặp dữ liệu mô phỏng “(MSA, Phylogeny)” bắt đầu từ chuỗi nucleotide thô, theo đúng các tham số và kịch bản chúng ta đã thảo luận:

---

### Bước 1. Chọn kịch bản mô phỏng và tham số mô phỏng

1. Từ dữ liệu thực, để mô phỏng, ta cần chia các dữ liệu thành các nhóm dựa trên đặc tính của chúng:

* 3 loại chuỗi: coding (C), noncoding (N), và hỗn hợp coding–noncoding (standard – S)
* 2 loại chiều dài chuỗi gốc: 200 (1) và 1000 (2). Sau đó cắt hoặc chèn gap sao cho độ dài đầu vào cố định, tương ứng là thành 240 hoặc 1200.
* 2 cách lấy phân bố độ dài nhánh mô phỏng: uniform (U) và gamma (G).
* Đặt tên các kịch bản này theo biểu thức chính quy “[SCN][1–2][UG]” như đã mô tả trong phần Kết quả.

2. Chọn ra 11 tham số chính mô phỏng được lấy từ các nghiên cứu trước đó (14, 15, 11, 12)

### Bước 2. Sinh cây phát sinh (Topology + Branch lengths)

* Sinh topology ngẫu nhiên với ETE3 (python) với số trình tự là N (từ 5-40)
* Lấy JSD mục tiêu (d1) từ tham số mô phỏng “Mean pairwise divergence”
* Gán cho mỗi nhánh độ dài được lấy mẫu từ phân phối uniform hoặc gamma.
* Tính JSD (d2) rồi scale các cạnh trong cây với tỷ số d1/d2 để đạt được JSD mô phỏng

### Bước 3. Mô phỏng MSA sử dụng bài báo Indelible

### Bước 4. Xây cặp “(MSA, Phylogeny)”

* Truncate độ dài MSA còn L.
* Mã hóa MSA thành ma trận kích thước : A→0, T→1, C→2, G→3, -→4, N→4
* Mã hóa topology quartet: Cây ((1,2),3,4)→0, ((1,3),2,4)→1, ((1,4),2,3)→2

### Bước 5. Chia dataset

* Lặp lại bước 2–4 cho tất cả kịch bản và tất cả giá trị N, để sinh ra M mẫu.
* Phân chia dataset thành train – val - test theo tỷ lệ 8:1:1

### Kết quả

* tensor dữ liệu kích thước $(M,4,L)$ và vector nhãn độ dài $M$, sử dụng cho training DL của Fusang.

## Dữ liệu thực tế

### 2.1. **Các trường hợp cực đoan** từ Pfam

* Từ Pfam, lấy được 9.444 MSA có ít hơn 40 trình tự và mỗi trình tự dài dưới 10.000.
* Sau đó, lấy hai bộ dữ liệu cực đoan:
  + 100 cây có độ lệch chuẩn divergence chuỗi cao nhất
  + 100 cây có độ lệch chuẩn divergence loài cao nhất (dữ liệu divergence loài lấy từ TimeTree)

### 2.2. Động vật có xương sống 100 trình tự (Chr1 UCSC)

* Dữ liệu động vật có xương sống từ UCSC với MSA 100 trình tự thuộc nhiễm sắc thể thứ nhất, dài 352.245.148 bp (base pair – chiều dài chuỗi nucleotide).

### 2.3. Dữ liệu whole-gene và transcriptome từ UCSC

* **Oryza** (11 chuỗi, whole-genome alignment, 359.187.370 bp)
* **Bướm** (16 chuỗi, MSA cho nhiễm sắc thể 21, 13.580.644 bp)
* **Côn trùng** (4 chuỗi, whole-genome alignment, 133.202.631 bp)
* **Động vật có xương sống** (8 chuỗi, whole-genome alignment, 463.540.596 bp)

### 2.4. Cây tiến hóa thực nghiệm

* Sinh ra một dữ liệu đảm bảo hoàn toàn ground-truth trong phòng thí nghiệm bằng việc chọn một trình tự gốc rồi chủ động tạo biến dị và hình thành các nhánh.

# Phương pháp

## Bài toán

* Cho một sắp hàng đa trình tự (MSA) với n trình tự (lá) và độ dài tùy ý L.
* Xây dựng cây không có gốc (unrooted) trên n trình tự đó sao cho cực đại hóa khả năng (Maximum Likelihood)

## Pha 1. Tiền xử lý dữ liệu

### 1. Sinh ma trận MSA

* Từ MSA n trình tự, sinh tất cả  MSA con, mỗi con có đúng 4 trình tự.
* Mỗi MSA con  được mã hóa thành ma trận số theo quy tắc:

|  |  |
| --- | --- |
| A | 0 |
| T | 1 |
| C | 2 |
| G | 3 |
| - | 4 |
| N | 4 |

* Kết quả: Ta được một ma trận .

### 2. Chia nhỏ với sliding window

* Chia nhỏ sử dụng sliding window độ dài để chia MSA độ dài L thành các MSA con độ dài w:
  + Nếu  thì chọn .
  + Nếu  thì chọn .
* Như vậy, số cửa sổ (số MSA con) và độ dài bước nhảy là:

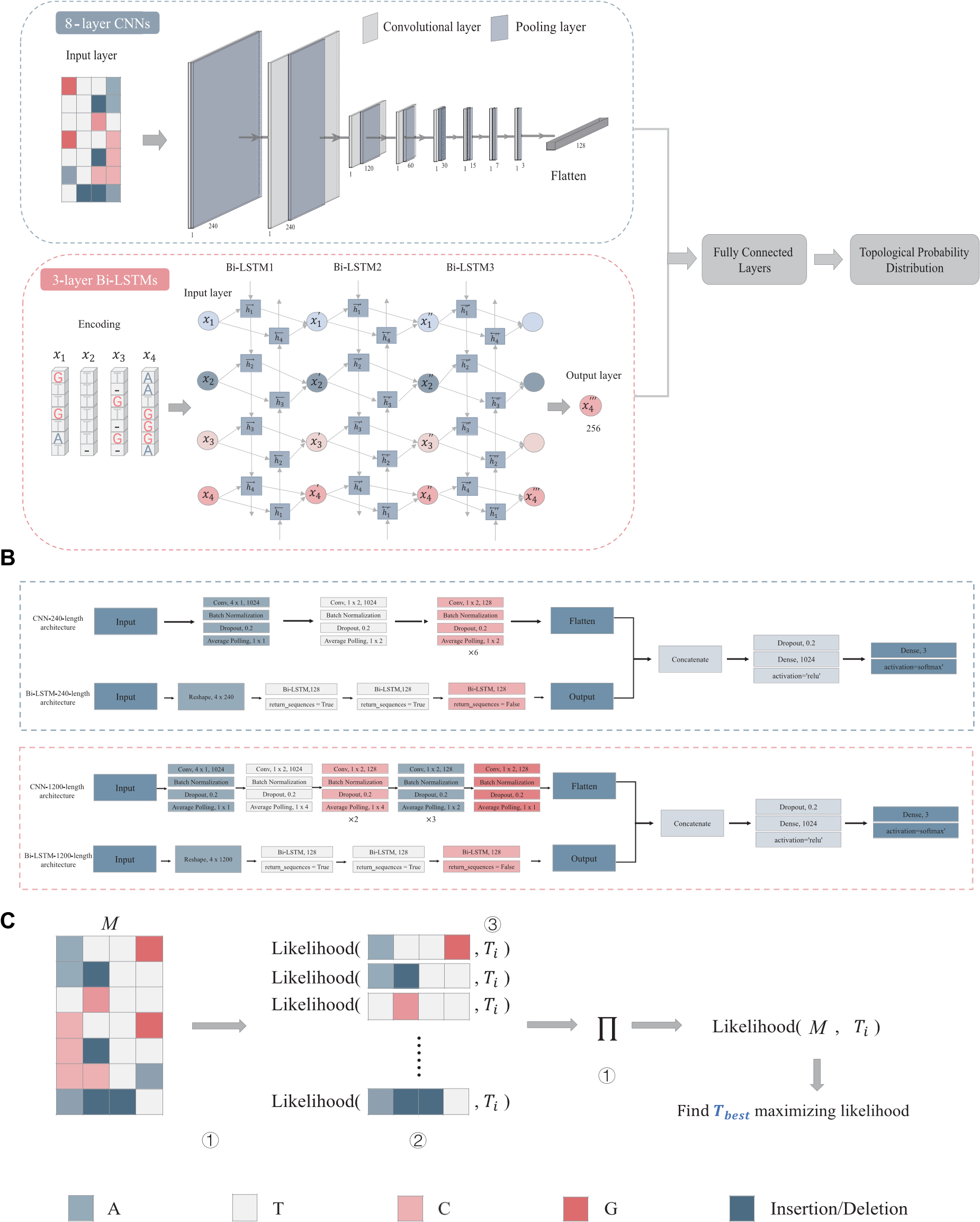


* + Trong đó  (mặc định) là siêu tham số quyết định số lượng cửa sổ trung bình để xuất hiện một nucleotide base.
* Kết quả: thu được ma trậntrong đó 

## Pha 2.Mô hình Deep Learning trên cây quartet độ dài cố định

### 1. Dự đoán với mô hình DL

* Huấn luyện 2 mô hình riêng biệt cho hai trường hợp  và .



* Input: Một ma trận vector (4, w) là một MSA con.
* Với CNN: Sau qua 8 lớp CNN và flatten được một vector .
* Với mô hình Bi-LSTM: Sau khi reshape và qua 3 lớp Bi-LSTM được 1 vector 
* Concatenate 2 vector đầu ra trên tạo 1 vector 
* Cho vector này qua 1 lớp Fully Connected được kết quả cuối cùng là một vector  là phân phối xác suất cho 3 topology : ((1,2),3,4)→0, ((1,3),2,4)→1, ((1,4),2,3)→2

### 2. Tích hợp kết quả

* Reshape đầu ra thành .
* Tính trung bình theo chiều , thu được ma trận  là xác suất thuộc ba loại topology cho từng cây quartet.
* Như vậy, sự kết hợp Mô hình Deep Learning trên cây quartet độ dài cố định với Kỹ thuật sliding window ở pha 1 là đền tảng trong việc xây dựng cây tổng quát ở pha 3.

## Pha 3. Beam search và xây dựng cây đa trình tự với độ dài thay đổi

### 1. Khởi tạo cây quartet

* Chọn cây quartet có xác suất cao nhất làm cây ban đầu .
* Gán score  xác suất topology của .

### 2. Beam search kết hợp stepwise addition

* Thiết lập beam size  (mặc định ). Mỗi beam là  là (Cây ứng viên, Xác suất topology)
* Duyệt lần lượt các nút còn lại (thứ tự đã được xếp sẵn):

1. Lấy nút  kế tiếp.
2. Với mỗi cây ứng viên  trong beam hiện tại, thử gắn  lên từng nhánh của  và tạo ra các cây ứng viên mới .
3. Pruning \*: trước khi chọn top, có thể cắt các nhánh “vô ích” dựa trên xác suất quartet (mask edges) để giảm số ứng viên.
4. Tính điểm cho :
   *  là tổng log-xác suất các quartet sinh ra từ C.
5. Giữ lại  cây  có score cao nhất làm beam cho bước tiếp.

* Khi đã thêm đủ  nút, cây trong beam cuối cùng mà có score tối đa là kết quả cần tìm.

#### Chi tiết về prunning

## Pha 4. Đánh giá

### 1. Độ chính xác phân loại

Độ chính xác phân loại của Fusang trên cây quartet trong 12 bộ dữ liệu mô phỏng [SCN][1–2][UG]. Công thức độ chính xác:



trong đó:

* TP là số trường hợp dương tính đúng
* TN là số trường hợp âm tính đúng
* FP là số trường hợp dương tính sai
* FN là số trường hợp âm tính sai.

So sánh trực tiếp với Tree\_learning, IQ-TREE và RAxML trong hai kịch bản chuẩn S2U và S2G, thực hiện trên MSA 4 trình tự và độ dài cố định 240 hoặc 1200.

### 2. **RF distance**

* Thử nghiệm trên dữ liệu MSA với ít hơn 40 trình tự: Tổng cộng 27.200 MSA, với 29 kịch: 4–30, 35 và 40 trình tự.
* Kết hợp kết quả trên Fusang với NJMerge-2 để xây dựng cây 100 trình tự để:
  + Kiểm chứng khả năng mở rộng (scalability) của Fusang
  + So sánh kết quả với IQ-TREE và RaxML sử dụng RF distance.

### 3. Kiểm thử trên dữ liệu sinh học thực

* Thử nghiệm trên dữ liệu từ các cơ sở dữ liệu công khai UCSC
* Thử nghiệm trên cây thật được tạo ra bằng đột biến nhân tạo trong phòng thí nghiệm.

### 4. Tốc độ tính toán

* Đo thời gian chạy Fusang, IQ-TREE và RAxML trên bốn tập dữ liệu điển hình (10, 20, 30, 40 loài).

# Kết quả

## 1. Độ chính xác phân loại trên cây quartet

* So sánh trong 2 kịch bản Standard

|  |  |  |
| --- | --- | --- |
| Phương pháp | S2U (240 bp) | S2G (1200 bp) |
| **Fusang** | **97.30%** | **90.70%** |
| Tree\_learning | **97.30%** | 90.68% |
| IQ-TREE | 89.51% | 82.05% |
| RAxML | 89.27% | 81.68% |

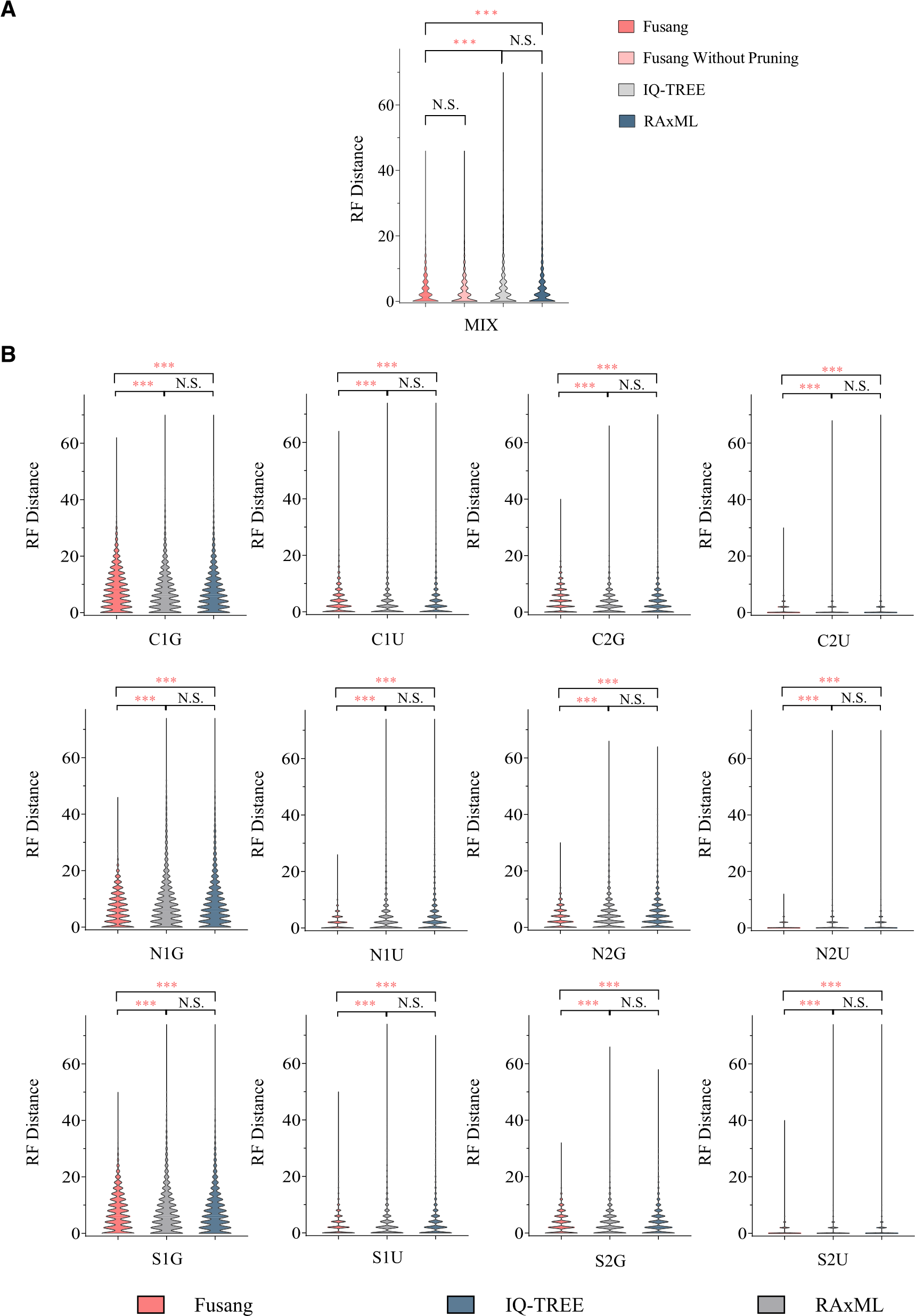
* Sử dụng thông tin indel:
* Train Fusang, Tree\_learning cùng các mô hình học máy trên cùng S2G, với/không indel.
* Mô hình DL thêm indel cải thiện độ chính xác đáng kể; các phương pháp ML chỉ tăng rất ít vì thường coi indel là ký tự không xác định.
* CNN của Fusang giống y hệt với CNN của Tree\_learning, tuy nhiên Fusang sử dụng thêm Bi-LSTM. Do đó, khi thí nghiệm 100 trường hợp cây quartet S2G, xét toàn bộ 24 hoán vị chuỗi thì Fusang sai ít hơn Tree\_learning 10/100 trường hợp.
* Discussion:
  + Sự quan trọng của thông tin indel
  + Lợi thế của BiLSTM trong việc trích xuất thông tin tuần tự.

## 2. Hiệu năng xây dựng cây phân loài chung

### 2.1. Đánh giá trên tập hỗn hợp 12 kịch bản

Sử dụng 29 bộ MSA đa lá (4–30, 35, 40 lá), mỗi bộ 5 MSA.

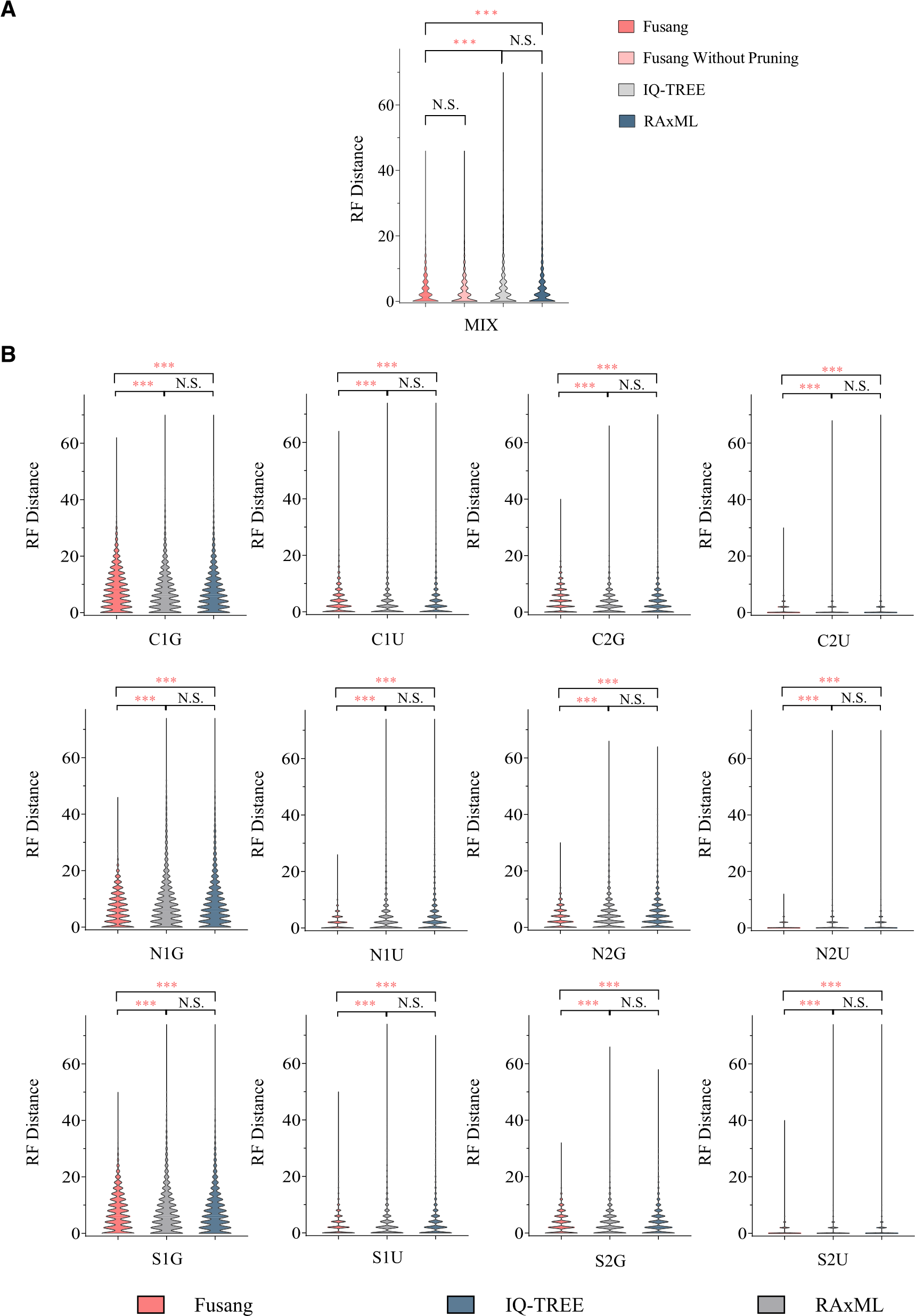
Sử dụng metric khoảng cách Robinson–Foulds (RF) giữa cây suy diễn và cây “thật”



* Fusang vượt trội phương pháp khác nhờ vào sử dụng dữ liệu indel như đã phân tích ở phần 1.
* Pruning không làm giảm độ chính xác quá nhiều, nhưng có tốc độ chạy nhanh hơn rất nhiều so với không pruning.

### 2.2. Hiệu năng theo từng kịch bản (12 scenarios)

* Sử dụng 12 kịch bản riêng (không hỗn hợp), mỗi kịch bản 27 200 MSA

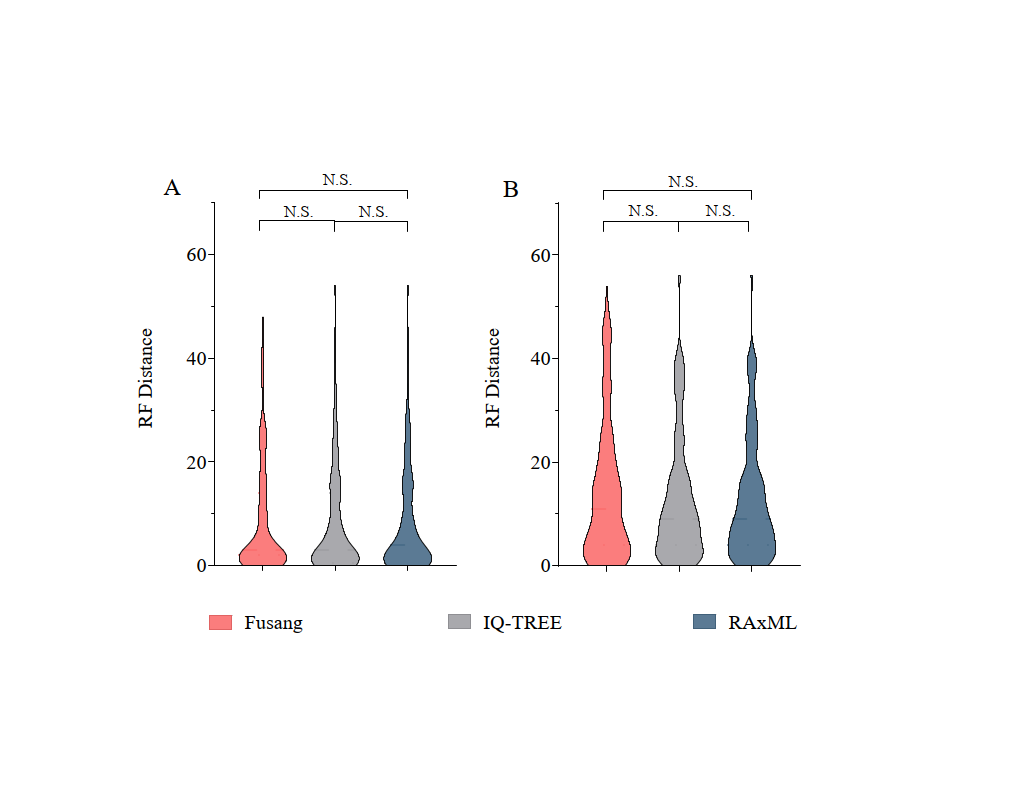


* Fusang luôn có RF thấp hơn các phương pháp trong 12 kịch bản.
* Không có khác biệt đáng kể giữa IQ-TREE và RAxML trong 12 kịch bản.

## 3. Hiệu năng trên dữ liệu sinh học thực tế

### 3.1. Trường hợp “cực đoan” từ Pfam

* Dữ liệu: 9444 MSA (≤ 40 lá; độ dài ≤ 10 000 bp) với 2 trường hợp
  1. 100 MSA có độ lệch chuẩn divergence chuỗi cao nhất
  2. 100 MSA có độ lệch chuẩn divergence loài cao nhất (TimeTree)



### 3.2. Cây 100 trình tự của động vật có xương sống (Chr1, UCSC)

* Dữ liệu: 100 MSA từ whole-genome Chr1 với độ dài 352 245 148 bp
* Thử nghiệm trên 10 cây con 10 trình tự: Fusang RF=12.3 vs IQ-TREE 11.8 vs RAxML 12.0 (Bảng S11)
* Xây cây 100 trình tự: ML methods lỗi RAM; Fusang+NJMerge-2 RF=66 vs Neighbor-Joining RF=90

### 3.3. Whole-genome & Transcriptome

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Dataset** | **Số trình tự** | **Length (bp)** | **Fusang RF** | **IQ-TREE RF** | **RAxML RF** |
| Oryza (WG) | 11 | 359 187 370 | 2 | 4 | 3 |
| Bướm (Chr21) | 16 | 13 580 644 | 2 | 5 | 4 |
| Côn trùng (WG) | 4 | 133 202 631 | 0 | 0 | 0 |
| ĐV có xương sống (WG) | 8 | 463 540 596 | 4 | 6 | 5 |

Fusang cho RF rất thấp (≤ 4) so với cây của UCSC.

### 3.4. Phylogeny thực nghiệm

|  |  |
| --- | --- |
| **Phương pháp** | **RF so với ground truth** |
| Fusang custom | 3 |
| Fusang S2G | 8 |
| IQ-TREE | 10 |
| RAxML | 11 |

## 4. Độ phức tạp thời gian

* Thời gian training: 120 tiếng trên Intel(R) Xeon(R) Gold 6140 CPU @ 2.30GHz 2 cores, and 1 Tesla V100 SXM2 32GB cho dữ liệu S2G với 6 triệu mẫu
* Thời gian toàn bộ quá trình suy luận tiến hóa (sử dụng pretrained model ở trên):
  + Các bộ dữ liệu, mỗi bộ 40 MSA:
* D1: 10 loài, trung bình 5.616 bp
* D2: 20 loài, trung bình 6.577 bp
* D3: 30 loài, trung bình 6.750 bp
* D4: 40 loài, trung bình 6.791 bp

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | D1(minutes) | D2(minutes) | D3(minutes) | D4(minutes) |
| IQ-TREE | 0,02 | 0,06 | 0,16 | 0,13 |
| RAxML | 0,02 | 0,17 | 0,36 | 0,89 |
| Fusang | 0,59 | 1,01 | 3,21 | 10,38 |
| IQ-TREE (Bootstrap) | 3,32 | 2,59 | 13,58 | 14,79 |
| RAxML (Bootstrap) | 0,37 | 5,37 | 5,22 | 10,5 |