**TRƯỜNG ĐẠI HỌC TÀI NGUYÊN VÀ MÔI TRƯỜNG THÀNH PHỐ HỒ CHÍ MINH**

**KHOA CÔNG NGHỆ THÔNG TIN**



**ĐỒ ÁN CHUYÊN NGÀNH  
NGÀNH CÔNG NGHỆ THÔNG TIN**

**PHÁT TRIỂN MÔ HÌNH HỌC SÂU KẾT HỢP  
EXPLAINABLE AI ĐỂ DỰ ĐOÁN VÀ PHÂN TÍCH ĐẶC TRƯNG CHUỖI PEPTIDE**

|  |  |  |
| --- | --- | --- |
| Giảng viên hướng dẫn | : | **ThS. Phạm Trọng Huynh** |
| Sinh viên hướng dẫn | : | **Hồ Tuấn Khanh** |
| Mã số sinh viên | : | **1050080103** |
| Lớp | : | **10\_ĐH\_CNPM2** |
| Khoá | : | **2021 - 2025** |

***TP. Hồ Chí Minh, tháng 5 năm 2025***

**TRƯỜNG ĐẠI HỌC TÀI NGUYÊN VÀ MÔI TRƯỜNG THÀNH PHỐ HỒ CHÍ MINH**

**KHOA CÔNG NGHỆ THÔNG TIN**



**ĐỒ ÁN CHUYÊN NGÀNH**

**NGÀNH CÔNG NGHỆ THÔNG TIN**

**PHÁT TRIỂN MÔ HÌNH HỌC SÂU KẾT HỢP  
EXPLAINABLE AI ĐỂ DỰ ĐOÁN VÀ PHÂN TÍCH ĐẶC TRƯNG CHUỖI PEPTIDE**

|  |  |  |
| --- | --- | --- |
| Giảng viên hướng dẫn | : | **ThS. Phạm Trọng Huynh** |
| Sinh viên hướng dẫn | : | **Hồ Tuấn Khanh** |
| Mã số sinh viên | : | **1050080103** |
| Lớp | : | **10\_ĐH\_CNPM2** |
| Khoá | : | **2021-2025** |

***TP. Hồ Chí Minh, tháng 5 năm 2025*LỜI CẢM ƠN**

Em xin được gửi lời tri ân sâu sắc tới Ban Giám Hiệu cùng toàn thể cán bộ, giảng viên của nhà trường – những người đã kiến tạo nên một môi trường học tập lý tưởng, đầy đủ điều kiện để em có thể học tập, rèn luyện và phát triển toàn diện trong suốt quãng thời gian gắn bó tại đây. Những năm tháng đồng hành cùng mái trường đại học không chỉ giúp em trang bị nền tảng kiến thức chuyên môn vững chắc, mà còn trao cho em những bài học quý báu về đạo đức nghề nghiệp, kỹ năng sống cũng như tinh thần tự học, tự nghiên cứu. Sự tận tụy và tâm huyết của các thầy cô là nguồn động lực to lớn, giúp em nhận ra rằng việc học không chỉ là tiếp thu tri thức mà còn là hành trình rèn luyện bản thân, vượt qua thử thách để trưởng thành mỗi ngày. Những giá trị ấy sẽ luôn là hành trang vững chắc, theo em trên chặng đường phát triển phía trước.

Em cũng xin gửi lời cảm ơn chân thành đến Khoa Công nghệ Thông tin – nơi đã đặt những viên gạch đầu tiên trong hành trình chinh phục lĩnh vực công nghệ mà em đam mê. Các thầy cô trong khoa không chỉ truyền đạt kiến thức lý thuyết sâu rộng mà còn không ngừng khơi dậy tinh thần sáng tạo, khám phá trong từng giờ học. Những môn học được thiết kế bài bản, mang tính ứng dụng cao đã giúp em tiếp cận thực tiễn công việc và trang bị đầy đủ kỹ năng để đáp ứng yêu cầu của ngành nghề. Em đặc biệt trân trọng sự tận tâm, đồng hành và dẫn dắt của các thầy cô trong suốt quá trình học tập – đó chính là nguồn cảm hứng giúp em duy trì ngọn lửa đam mê và luôn nỗ lực không ngừng trên con đường mình đã chọn.

Đặc biệt, em xin bày tỏ lòng biết ơn sâu sắc đến Thạc sĩ Phạm Trọng Huynh – người thầy đã không chỉ tận tình hướng dẫn mà còn chia sẻ những kiến thức quý báu và kinh nghiệm thực tiễn trong suốt quá trình em học tập và rèn luyện. Sự đồng hành và những lời chỉ bảo chân thành của thầy không chỉ giúp em nâng cao chuyên môn mà còn rèn giũa những kỹ năng thiết yếu trong môi trường làm việc sau này. Nhờ có sự dẫn dắt của thầy, em đã có cơ hội tiếp cận và hiểu rõ hơn về cách vận hành của các mô hình công nghệ thông tin trong môi trường chuyên môn – điều mà em tin chắc sẽ là nền tảng quan trọng cho sự phát triển sự nghiệp của bản thân trong tương lai.

Một lần nữa, em xin kính gửi lời cảm ơn chân thành và sâu sắc nhất đến toàn thể thầy cô và nhà trường. Kính chúc quý thầy cô luôn dồi dào sức khỏe, thành công trong sự nghiệp trồng người và chúc nhà trường ngày càng phát triển mạnh mẽ, tiếp tục đóng góp tích cực cho công cuộc giáo dục và đào tạo nguồn nhân lực chất lượng cho xã hội.

**Sinh viên thực hiện**

**Họ và Tên**

**LỜI CAM ĐOAN**

Toàn bộ nội dung trong báo cáo này được thực hiện dựa trên quá trình học tập, tìm hiểu và nghiên cứu một cách nghiêm túc, có tham khảo các nguồn tài liệu rõ ràng, minh bạch. Các số liệu, kết quả, hình ảnh và nội dung trình bày trong báo cáo đều trung thực và chưa từng được sử dụng trong bất kỳ công trình nào khác.

Nếu có bất kỳ sai phạm nào liên quan đến tính trung thực hoặc vi phạm bản quyền, hoàn toàn chịu trách nhiệm trước quy định của nhà trường và pháp luật.

Kính mong sự xem xét và đánh giá từ quý thầy cô.

**Sinh viên thực hiện**

(ký tên và ghi rõ họ tên)

**NHẬN XÉT**

**(giảng viên hướng dẫn)**

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

**Kết luận: (Đồng ý hoặc Không đồng ý cho sinh viên nộp báo cáo)**

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

TP. Hồ Chí Minh, ngày… tháng….năm 2025

**GIẢNG VIÊN HƯỚNG DẪN**

(Ký và ghi rõ học hàm, học vị họ tên)

**ThS. Phạm Trọng Huynh**

**TS. Dương Thị Thúy Nga**

**NHẬN XÉT**

**(giảng viên phản biện)**

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

**Kết luận: (Đồng ý hoặc Không đồng ý cho sinh viên nộp báo cáo)**

…………………………………………………………………………………………

…………………………………………………………………………………………

…………………………………………………………………………………………

TP. Hồ Chí Minh, ngày… tháng….năm 2025

**GIẢNG VIÊN PHẢN BIỆN**

(Ký và ghi rõ học hàm, học vị họ tên)

**LỊCH LÀM VIỆC**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tuần** | **Nội dung** | **Địa điểm** | **Nhận xét của GV hướng dẩn** | **Chữ ký của GV hướng dẩn** |
| 1 | Tìm hiểu về dữ liệu | Trường đại học Tài Nguyên Và Môi Trường TPHCM |  |  |
| 2 | Tìm hiểu về mô hình | Trường đại học Tài Nguyên Và Môi Trường TPHCM |  |  |
| 3 | Đọc tài liệu về cách cào dữ liệu | Trường đại học Tài Nguyên Và Môi Trường TPHCM |  |  |
| 4 | Đọc tài liệu về các phương pháp xử lý dữ liệu | Trường đại học Tài Nguyên Và Môi Trường TPHCM |  |  |
| 5 | Meeting với giáo viên hướng dẫn | Trường đại học Tài Nguyên Và Môi Trường TPHCM |  |  |
| 6 | Bắt đầu thực hiện cào dữ liệu và xử lý dữ liệu | Trường đại học Tài Nguyên Và Môi Trường TPHCM |  |  |
| 7 | Xây dựng mô hình | Trường đại học Tài Nguyên Và Môi Trường TPHCM |  |  |
| 8 | Đọc tài liệu về Explainable AI và thực hiện | Trường đại học Tài Nguyên Và Môi Trường TPHCM |  |  |
| 9 | Viết báo cáo | Trường đại học Tài Nguyên Và Môi Trường TPHCM |  |  |
| 10 | Làm slide báo cáo | Trường đại học Tài Nguyên Và Môi Trường TPHCM |  |  |

**TÓM TẮT ĐỒ ÁN**

**1. Vấn đề đã đưa ra giải quyết**

Chuỗi peptide, đặc biệt là của các loài vi sinh vật, đóng vai trò quan trọng trong nhiều quá trình sinh học và có tiềm năng kháng khuẩn. Tuy nhiên, việc dự đoán và phân tích đặc trưng của các chuỗi này gặp nhiều thách thức do độ dài biến thiên và cấu trúc phức tạp. Ngoài yêu cầu dự đoán chính xác khả năng kháng khuẩn, các nhà nghiên cứu còn cần hiểu rõ **nguyên nhân** đằng sau các dự đoán của mô hình. Do đó, việc kết hợp Explainable Artificial Intelligence (XAI) là cần thiết nhằm giải thích cơ sở của các quyết định được đưa ra bởi mô hình học sâu.

**2. Mục tiêu đã đạt được**

Xây dựng mô hình học sâu Transformer để dự đoán khả năng kháng khuẩn của chuỗi peptide.

Kết hợp phương pháp XAI LIME để giải thích các quyết định dự đoán của mô hình.

Sử dụng hai bộ dữ liệu từ NCBI (GenBank và một bộ tổng hợp từ nhiều nguồn), tiến hành so sánh hiệu quả mô hình trên từng bộ và huấn luyện mô hình tổng hợp dữ liệu từ cả hai.

**3. Kỹ thuật sử dụng, thiết kế tổng quát của đề tài, thuật toán được sử dụng**

Kỹ thuật: Mô hình học sâu Transformer, phương pháp XAI LIME.

Hạ tầng: Amazon EC2 sử dụng GPU cho việc thu thập, xử lý dữ liệu và huấn luyện mô hình.

Thiết kế tổng quát: Tiến hành thu thập và xử lý dữ liệu chuỗi peptide với độ dài từ 10–50. Tạo ra các thư viện sinh học dùng để tạo ra các đặc trưng của chuỗi. Các đặc trưng này được đưa vào mô hình Transformer để huấn luyện và dự đoán. Cuối cùng, tích hợp phương pháp LIME để giải thích các dự đoán. Việc huấn luyện được thực hiện trên 2 bộ dữ liệu khác nhau và trên một bộ dữ liệu gộp lại giữa 2 bộ dữ liệu.

Thuật toán chính: Transformer với cơ chế self-attention, kết hợp cùng LIME để giải thích đầu ra mô hình.

**4. Kết quả thực hiện như thế nào**

Mô hình đạt độ chính xác trên 90% trong việc phân loại chuỗi peptide có khả năng kháng khuẩn.

Có thể giải thích rõ ràng các yếu tố ảnh hưởng đến dự đoán thông qua XAI.

Xác định được bộ dữ liệu nào mang lại hiệu quả huấn luyện tốt hơn giữa hai bộ dữ liệu từ NCBI (GenBank và một bộ tổng hợp từ nhiều nguồn).

**5. Kết luận về Đồ án**

Đồ án đã đạt được các mục tiêu đề ra. Trong tương lai, cần tiếp tục mở rộng và bổ sung dữ liệu để tối ưu hiệu quả của mô hình Transformer một kiến trúc đặc biệt mạnh khi được huấn luyện với tập dữ liệu lớn Ngoài ra, có thể nghiên cứu tích hợp thêm các phương pháp giải thích khác nhằm nâng cao độ tin cậy và minh bạch của mô hình.

**MỤC LỤC**

[MỞ ĐẦU 1](#_Toc197562083)

[CHƯƠNG 1: TỔNG QUAN TÀI LIỆU 8](#_Toc197562084)

[1.1. Các dự án tương tự trước đây 8](#_Toc197562085)

[1.1.1. Các mô hình học sâu trong dự đoán chức năng peptide 8](#_Toc197562086)

[1.1.2. Tích hợp kiến trúc Transformer trong sinh học phân tử 9](#_Toc197562087)

[1.1.3. Các nghiên cứu tích hợp Explainable AI (XAI) trong sinh học 10](#_Toc197562088)

[1.1.4. Sự khác biệt và điểm mới của đề tài: 10](#_Toc197562089)

[1.2. Cơ sở lý thuyết 11](#_Toc197562090)

[1.2.1. Lý thuyết 11](#_Toc197562091)

[1.2.2. Kỹ thuật 15](#_Toc197562092)

[CHƯƠNG 2: PHƯƠNG PHÁP THỰC HIỆN 32](#_Toc197562093)

[2.1. Các yêu cầu kĩ thuật 32](#_Toc197562094)

[2.2. Tổng thể về thiết kế phát triển mô hình 33](#_Toc197562095)

[2.2.1. Các thành phần chính 36](#_Toc197562096)

[2.2.2. Những mối liên hệ của các thành phần 38](#_Toc197562097)

[2.3. Chi tiết quá trình thực hiện 39](#_Toc197562098)

[2.4. Sự lựa chọn phương án thiết kế và công nghệ 41](#_Toc197562099)

[2.4.1. Công nghệ 41](#_Toc197562100)

[2.4.2. Phương án thiết kế 41](#_Toc197562101)

[2.4.3. Lý do lựa chọn công nghệ và phương án thiết kế 44](#_Toc197562102)

[2.5. Những thách thức về thuật toán 45](#_Toc197562103)

[2.5.1. Tiền xử lý dữ liệu 45](#_Toc197562104)

[2.5.2. Thuật toán mô hình Transformer 49](#_Toc197562105)

[2.5.3. Thuật toán LIME 52](#_Toc197562106)

[CHƯƠNG 3: CÀI ĐẶT THỬ NGHIỆM, KẾT QUẢ VÀ ĐÁNH GIÁ 55](#_Toc197562107)

[3.1. Các thiết kế mô hình giải pháp 55](#_Toc197562108)

[3.1.1. Thu thập và xử lý dữ liệu 55](#_Toc197562109)

[3.1.2. Xây dựng mô hình học sâu Transformer 61](#_Toc197562110)

[3.1.3. Tích hợp XAI vào mô hình Transformer 67](#_Toc197562111)

[3.1.4. Xây dựng giao diện 69](#_Toc197562112)

[3.2. Các khó khăn và thử thách 70](#_Toc197562113)

[3.3. Kết quả đạt được 71](#_Toc197562114)

[3.3.1. Kết quả huấn luyện 72](#_Toc197562115)

[3.3.2. Kết quả giải thích dự đoán của mô hình Transformer qua Explainable AI bằng phương pháp LIME 81](#_Toc197562116)

[3.3.3. So sánh kết quả huấn luyện của mô hình Transformer với mô hình LSTM 83](#_Toc197562117)

[3.4. Hạn chế và đề xuất cải thiện 84](#_Toc197562118)

[CHƯƠNG 4: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 86](#_Toc197562119)

[DANH MỤC TÀI LIỆU THAM KHẢO 88](#_Toc197562120)

**DANH MỤC SƠ ĐỒ**

[Sơ đồ 2. 1. Sơ đồ tổng thể thiết kế phát triển mô hình 33](#_Toc197538474)

[Sơ đồ 2. 2. Sơ đồ chi tiết quá trình thực hiện 39](#_Toc197538475)

[Sơ đồ 2. 3. Sơ đồ chi tiết phương án thiết kế 42](#_Toc197538476)

[Sơ đồ 2. 4. Sơ đồ kiến trúc mô hình Transformer 49](#_Toc197538477)

[Sơ đồ 2. 5. Sơ đồ kiến trúc mô hình Transformer Encoder 50](#_Toc197538478)

[Sơ đồ 3. 1. Sơ đồ kiến trúc mô hình Transformer tùy chỉnh………………………….62](#_Toc197538472)

[Sơ đồ 3. 2. Sơ đồ cơ chế Multi-Head Attention 64](#_Toc197538473)

**DANH MỤC HÌNH**

[Hình 1. 1. Hình ảnh chuỗi peptide 13](#_Toc197538451)

[Hình 3. 1. Hình ảnh kết quả huấn luyện trên bộ dữ liệu GenBank (accuracy)……... 73](#_Toc197538455)

[Hình 3. 2. Hình ảnh kết quả huấn luyện trên bộ dữ liệu GenBank (loss) 74](#_Toc197538456)

[Hình 3. 3. Hình ảnh kết quả huấn luyện trên bộ dữ liệu GenBank (đánh giá hiệu suất) 74](#_Toc197538457)

[Hình 3. 4. Hình ảnh kết quả huấn luyện trên bộ dữ liệu GenBank (ma trận nhầm lẫn) 75](#_Toc197538458)

[Hình 3. 5. Hình ảnh kết quả huấn luyện trên bộ dữ liệu PDB, RefSeq, UnitProtKB, Swisss-Prot, DDBJ, EMBL, PIR (accuracy) 76](#_Toc197538459)

[Hình 3. 6. Hình ảnh kết quả huấn luyện trên bộ dữ liệu PDB, RefSeq, UnitProtKB, Swisss-Prot, DDBJ, EMBL, PIR (loss) 77](#_Toc197538460)

[Hình 3. 7. Hình ảnh kết quả huấn luyện trên bộ dữ liệu PDB, RefSeq, UnitProtKB, Swisss-Prot, DDBJ, EMBL, PIR (đánh giá hiệu suất) 77](#_Toc197538461)

[Hình 3. 8. Hình ảnh kết quả huấn luyện trên bộ dữ liệu PDB, RefSeq, UnitProtKB, Swisss-Prot, DDBJ, EMBL, PIR (ma trận nhầm lẫn) 78](#_Toc197538462)

[Hình 3. 9. Hình ảnh kết quả huấn luyện trên bộ dữ liệu kết hợp (đánh giá hiệu suất) 79](#_Toc197538463)

[Hình 3. 10. Hình ảnh kết quả huấn luyện trên bộ dữ liệu kết hợp (ma trận nhẫm lẫn) 80](#_Toc197538464)

[Hình 3. 11. Hình ảnh kết quả giải thích dự đoán của mô hình Transformer bằng LIME (1) 81](#_Toc197538465)

[Hình 3. 12. Hình ảnh kết quả giải thích dự đoán của mô hình Transformer bằng LIME (2) 81](#_Toc197538466)

[Hình 3. 13. Hình ảnh kết quả giải thích dự đoán của mô hình Transformer bằng LIME (3) 82](#_Toc197538467)

[Hình 3. 14. Hình ảnh kết quả giải thích dự đoán của mô hình Transformer bằng LIME (4) 82](#_Toc197538468)

[Hình 3. 15. Hình ảnh kết quả giải thích dự đoán của mô hình Transformer bằng LIME (5) 83](#_Toc197538469)

[Hình 3. 16. Hình ảnh kết quả huấn luyện bằng mô hình LSTM (ma trận nhầm lẫn) 83](#_Toc197538470)

[Hình 3. 17. Hình ảnh kết quả huấn luyện bằng mô hình LSTM (đánh giá hiệu suất) 84](#_Toc197538471)

**KÍ HIỆU CÁC CỤM TỪ VIẾT TẮT**

|  |  |
| --- | --- |
| Từ viết tắt | Ý nghĩa |
| XAI | Explainable AI |
| LIME | Local Interpretable Model-agnostic Explanations |
| SHAP | SHapley Additive exPlanations |
| AI | Artificial Intelligence |
| NCBI | National Center for Biotechnology Information |
| DeepACP | Deep Learning for Anticancer Peptide prediction |
| AntiCP | Anticancer Peptide |
| iAMPpred | in silico Antimicrobial Peptide predictor |
| AI4AMP | Artificial Intelligence for Antimicrobial Peptides |
| AMPScanner | Antimicrobial Peptide Scanner |
| CNN | Convolutional Neural Network |
| RNN | Recurrent Neural Network |
| LSTM | Long Short-Term Memory |
| BiGRU | Bidirectional Gated Recurrent Unit |
| NLP | Natural Language Processing |
| TAPE | Tasks Assessing Protein Embeddings |
| ProtBERT | Protein Bidirectional Encoder Representations from Transformers |
| ProtTrans | Protein Transformer |
| UniProt | Universal Protein Resource |
| ProtT5 | Protein Text-to-Text Transfer Transformer |
| Grad-CAM | Gradient-weighted Class Activation Mapping |
| RNA-seq | RNA sequencing |
| DL | Deep Learning |
| MHSA | Multi-Head Self-Attention Mechanism |
| FFNN | Feed-Forward Neural Network |
| SDPA | Scaled Dot-Product Attention |
| Q | Query |
| K | Key |
| V | Value |
| MHA | Multi-Head Attention |
| BERT | Bidirectional Encoder Representations from Transformers |
| GPT | Generative Pre-trained Transformer |
| T5 | Text-To-Text Transfer Transformer |
| PDB | Protein Data Bank |
| RefSeq | Reference Sequence Database |
| DDBJ | DNA Data Bank of Japan |
| EMBL | European Molecular Biology Laboratory |
| PIR | Protein Information Resource |
| AMP | Antimicrobial Peptides |
| NAMP | Non-Antimicrobial Peptides |
| API | Application Programming Interface |
| AWS | Amazon Web Services |
| EC2 | Elastic Compute Cloud |
| CI/CD | Continuous Integration/Continuous Deployment |
| Add & Norm | Residual Connection and Layer Normalization |
| MLP Head | Multi-Layer Perceptron Head |
| FFN | Feed Forward Neural Network |
| ReLU | Rectified Linear Unit |
| GenBank | Genetic Sequence Database |
| UniProtKB | Universal Protein Knowledgebase |
| Swiss-Prot | Swiss Protein Database |

# **MỞ ĐẦU**

**1. Tên đề tài cần thực hiện**

Tên đề tài cần thực hiện là: “Phát triển mô hình học sâu kết hợp Explainable AI để dự đoán và phân tích đặc trưng chuỗi peptide”. Cụ thể, mô hình học sâu Transformer sẽ được áp dụng để phân tích và học chuỗi peptide của các loài vi sinh vật, trong khi Explainable AI sẽ sử dụng kỹ thuật LIME (Local Interpretable Model-agnostic Explanations) để giải thích các quyết định của mô hình.

**2. Cơ sở khoa học và tính thực tiễn của đề tài**

**2.1. Cơ sở lý luận.**

Học sâu (Deep Learning) đã và đang là một công cụ đắc lực trong phân tích dữ liệu sinh học, đặc biệt là các chuỗi sinh học như DNA, RNA và peptide. Trong đó, mô hình Transformer – ban đầu phát triển cho xử lý ngôn ngữ tự nhiên – đã chứng minh hiệu quả vượt trội trong việc học biểu diễn từ chuỗi dữ liệu, nhờ vào cơ chế self-attention cho phép mô hình nắm bắt mối liên hệ dài hạn giữa các phần tử trong chuỗi.

Trong bối cảnh phân tích chuỗi peptide – vốn là các chuỗi axit amin ngắn có vai trò sinh học quan trọng – Transformer được xem là kiến trúc lý tưởng để khai thác thông tin thứ tự và ngữ cảnh phức tạp giữa các vị trí trên chuỗi. Các đặc trưng như khả năng kháng khuẩn,… của peptide đều có thể được suy ra từ trình tự và cấu trúc của nó, thông qua việc học biểu diễn ẩn hiệu quả.

Tuy nhiên, một hạn chế lớn của các mô hình học sâu, đặc biệt là Transformer, là tính khó giải thích – thường được coi là “hộp đen”. Điều này đặc biệt quan trọng trong lĩnh vực sinh học, nơi mà các quyết định cần được minh bạch và có thể truy vết.

Để giải quyết vấn đề này, phương pháp LIME (Local Interpretable Model-agnostic Explanations) được sử dụng như một kỹ thuật giải thích mô hình độc lập (model-agnostic). LIME giúp xác định những đặc trưng đầu vào nào ảnh hưởng mạnh nhất đến kết quả dự đoán, bằng cách tạo ra các mẫu nhiễu cục bộ quanh dữ liệu và huấn luyện một mô hình đơn giản để giải thích kết quả dự đoán của mô hình gốc. Việc tích hợp LIME (Local Interpretable Model-agnostic Explanations) vào mô hình Transformer giúp cung cấp cái nhìn trực quan và định lượng về vai trò của từng axit amin trong chuỗi peptide đối với kết quả dự đoán.

**2.2. Cơ sở thực tiễn**

Các nghiên cứu gần đây đã chỉ ra rằng mô hình học sâu có khả năng vượt trội trong việc phân loại và dự đoán đặc trưng của peptide, tuy nhiên vẫn chưa phổ biến việc kết hợp với các kỹ thuật giải thích để tăng tính minh bạch. Trong khi đó, yêu cầu giải thích là yếu tố then chốt để các mô hình AI được chấp nhận rộng rãi trong các nghiên cứu y sinh, thiết kế thuốc, và ứng dụng lâm sàng.

LIME (Local Interpretable Model-agnostic Explanations) được sử dụng hiệu quả trong nhiều lĩnh vực để làm rõ đóng góp của từng phần tử đầu vào đến đầu ra của mô hình, và đang được mở rộng sang cả lĩnh vực sinh học phân tử. Kết hợp Transformer và LIME trong bài toán dự đoán peptide có kháng khuẩn hay không vừa tận dụng sức mạnh biểu diễn của học sâu, vừa nâng cao khả năng giải thích dự đoán của mô hình.

**2.3. Các nghiên cứu liên quan**

Một số mô hình điển hình như ProtTrans (Transformer-based model cho protein), AMPlify, hay DeepAmPEP đã thể hiện tiềm năng trong việc dự đoán các đặc tính sinh học từ chuỗi peptide, nhưng chủ yếu tập trung vào độ chính xác mà ít chú trọng đến khả năng giải thích. Về mặt XAI, LIME đã được áp dụng thành công trong nhiều bài toán phức tạp như chẩn đoán bệnh, phân loại văn bản, và gần đây đang được khai thác trong phân tích trình tự sinh học.

Tuy nhiên, việc tích hợp có mô hình giữa Transformer và LIME để vừa dự đoán vừa phân tích vai trò đặc trưng của chuỗi peptide là một hướng đi còn mới và tiềm năng, góp phần giải quyết thách thức về độ tin cậy và khả năng ứng dụng trong thực tế của các mô hình AI trong y sinh học.

**3. Tính thực tiễn của đề tài**

Đề tài mang tính thực tiễn rất cao cao, xuất phát từ nhu cầu cấp thiết trong nhiều lĩnh vực khoa học và ứng dụng thực tiễn như y học, công nghệ sinh học, và phát triển thuốc.

Hiện nay, chuỗi peptide giữ vai trò quan trọng trong các quá trình sinh học như miễn dịch, chống vi khuẩn, dẫn truyền tín hiệu tế bào, và là cơ sở để thiết kế các loại thuốc điều trị bệnh lý nguy hiểm như ung thư, bệnh truyền nhiễm, tự miễn... Tuy nhiên, việc phân tích chức năng và đặc trưng của peptide theo phương pháp truyền thống (in vitro, in vivo hoặc mô hình sinh học thủ công) thường tốn nhiều thời gian, công sức, chi phí cao và khó mở rộng trên quy mô lớn. Do đó, nhu cầu ứng dụng các công nghệ tiên tiến để hỗ trợ và tự động hóa quá trình này là rất cấp thiết.

Về mặt công nghệ, sự phát triển mạnh mẽ của trí tuệ nhân tạo, đặc biệt là mô hình học sâu Transformer, đã mở ra một hướng đi mới trong việc xử lý dữ liệu chuỗi sinh học. Transformer có khả năng học các mối quan hệ phức tạp trong chuỗi peptide mà không cần phải thiết kế đặc trưng thủ công như các phương pháp truyền thống. Đây là phương pháp nghiên cứu mới, hiện đại, đang dần thay thế các kỹ thuật cũ nhờ khả năng học tự động, độ chính xác cao và khả năng xử lý dữ liệu lớn.

Tuy nhiên, hạn chế lớn của học sâu là tính “hộp đen”, tức là khó giải thích được lý do vì sao mô hình đưa ra dự đoán. Điều này là rào cản trong các lĩnh vực như y học và sinh học, nơi tính minh bạch và khả năng kiểm chứng là yếu tố then chốt. Chính vì vậy, đề tài đã kết hợp thêm phương pháp Explainable AI (XAI), cụ thể là LIME (Local Interpretable Model-Agnostic Explanations) – một kỹ thuật tiên tiến giúp giải thích rõ ràng các quyết định của mô hình bằng cách chỉ ra yếu tố đầu vào nào ảnh hưởng đến đầu ra. Nhờ đó, mô hình không chỉ dự đoán chính xác mà còn giúp các nhà nghiên cứu hiểu rõ bản chất của chuỗi peptide và hiểu rõ quyết định của mô hình học sâu, phục vụ cho việc xác nhận sinh học hoặc thiết kế nghiên cứu tiếp theo.

Từ những lý do trên, đề tài không chỉ có ý nghĩa về mặt khoa học – khi áp dụng phương pháp nghiên cứu mới để giải quyết một bài toán sinh học quan trọng – mà còn có giá trị thực tiễn cao, khi đóng góp vào quá trình phát triển công cụ hỗ trợ phân tích peptide phục vụ cho nghiên cứu, y học và dược phẩm.

**4. Mục tiêu cần đạt được**

Mục tiêu đầu tiên của đề tài là xây dựng một mô hình học sâu sử dụng kiến trúc Transformer để dự đoán khả năng kháng khuẩn của các chuỗi peptide. Mô hình này sẽ giúp phân tích và phát hiện các đặc trưng quan trọng trong chuỗi peptide có liên quan đến khả năng kháng khuẩn, từ đó nâng cao hiệu quả trong việc dự đoán tính kháng khuẩn của peptide.

Mục tiêu thứ hai là áp dụng phương pháp Explainable AI (XAI), cụ thể là LIME (Local Interpretable Model-agnostic Explanations), để giải thích các quyết định của mô hình học sâu. Việc sử dụng LIME sẽ giúp tăng cường tính minh bạch của mô hình, cung cấp thông tin chi tiết về các yếu tố ảnh hưởng đến kết quả dự đoán và giúp người dùng hiểu rõ hơn về quá trình ra quyết định của mô hình.

Cuối cùng, đề tài sẽ sử dụng hai bộ dữ liệu từ NCBI, bao gồm GenBank và một bộ dữ liệu tổng hợp từ nhiều nguồn khác nhau, để huấn luyện và đánh giá mô hình. Mô hình sẽ được kiểm tra hiệu quả trên từng bộ dữ liệu riêng biệt, và sau đó sẽ huấn luyện một mô hình tổng hợp dữ liệu từ cả hai bộ dữ liệu nhằm cải thiện độ chính xác trong việc dự đoán khả năng kháng khuẩn của chuỗi peptide.

**5. Phương pháp nghiên cứu**

Đề tài này áp dụng phương pháp nghiên cứu học sâu (Deep Learning) kết hợp với Explainable AI (XAI) nhằm dự đoán và phân tích khả năng kháng khuẩn của chuỗi peptide. Cụ thể, sẽ sử dụng mô hình Transformer, một kiến trúc mạnh mẽ trong học sâu để xử lý và dự đoán các đặc trưng của chuỗi peptide. Mô hình này sẽ giúp phân tích các mối quan hệ phức tạp trong dữ liệu chuỗi, từ đó đưa ra những dự đoán chính xác về khả năng kháng khuẩn của peptide.

Để tăng cường tính minh bạch của mô hình và giải thích các quyết định dự đoán của mô hình, phương pháp Explainable AI sẽ được áp dụng thông qua LIME (Local Interpretable Model-agnostic Explanations). LIME giúp giải thích lý do tại sao mô hình đưa ra các dự đoán cụ thể, từ đó giúp người dùng hiểu rõ hơn về các yếu tố quan trọng ảnh hưởng đến khả năng kháng khuẩn của peptide.

Để huấn luyện và kiểm tra mô hình, chúng tôi sẽ sử dụng hai bộ dữ liệu sinh học từ NCBI, bao gồm GenBank và một bộ dữ liệu tổng hợp từ nhiều nguồn khác nhau. Mô hình sẽ được huấn luyện trên từng bộ dữ liệu riêng biệt để so sánh độ hiệu quả của hai bộ dữ liệu và sau đó tổng hợp các bộ dữ liệu này để tối ưu hóa hiệu quả dự đoán. Việc sử dụng các bộ dữ liệu đa dạng giúp đảm bảo tính tổng quát và độ chính xác cao trong việc dự đoán khả năng kháng khuẩn của chuỗi peptide.

**6. Nội dung công việc tương ứng với các phương pháp nghiên cứu đã được đề ra**

**6.1. Thu thập dữ liệu:**

Dữ liệu được thu thập từ API của NCBI, thông qua việc cào thông tin từ GenBank và các bộ dữ liệu tổng hợp từ nhiều nguồn khác nhau. Các chuỗi peptide được lấy từ cơ sở dữ liệu này, đảm bảo cung cấp dữ liệu phong phú và đa dạng về các peptide và khả năng kháng khuẩn của chúng. Việc thu thập dữ liệu này giúp tạo ra một tập dữ liệu đa dạng, từ đó huấn luyện và kiểm tra mô hình dự đoán.

**6.2. Xử lý dữ liệu:**

Dữ liệu thu thập được sẽ được xử lý để loại bỏ các chuỗi peptide không đủ chất lượng hoặc có độ dài không phù hợp. Các chuỗi peptide có độ dài lớn hơn 50 sẽ được tách nhỏ lại, mục tiêu là đưa các chuỗi về độ dài từ 10-50 amino acids, vì đây là độ dài ổn định và hiệu quả nhất của peptide trong nghiên cứu kháng khuẩn. Việc này giúp cải thiện khả năng huấn luyện và tính chính xác của mô hình.

**6.3. Sinh đặc trưng của chuỗi:**

Để giúp mô hình học sâu hiểu rõ hơn về các chuỗi peptide, việc tạo ra thư viện chuyên dụng về sinh học là cần thiết để sinh ra các đặc trưng của từng chuỗi peptide. Những đặc trưng này bao gồm thông tin về cấu trúc, tính chất điện tích, hydrophobicity, và các yếu tố khác ảnh hưởng đến hoạt tính kháng khuẩn. Các đặc trưng này sẽ được dùng làm đầu vào cho mô hình học sâu, giúp mô hình học được mối quan hệ giữa cấu trúc và hoạt tính của peptide.

**6.4. Xây dựng mô hình Transformer:**

Mô hình học sâu Transformer sẽ được xây dựng để dự đoán khả năng kháng khuẩn của chuỗi peptide. Transformer, với khả năng xử lý chuỗi dữ liệu mạnh mẽ, sẽ giúp nhận diện các mẫu phức tạp trong cấu trúc peptide và liên kết chúng với khả năng kháng khuẩn. Mô hình này sẽ được huấn luyện trên tập dữ liệu đã qua xử lý, sử dụng các đặc trưng đã được sinh ra từ bước trước.

**6.5. Sử dụng XAI (Explainable AI):**

Để làm cho mô hình dễ hiểu và minh bạch hơn, chúng tôi sử dụng LIME (Local Interpretable Model-agnostic Explanations), một phương pháp XAI, để giải thích các quyết định mà mô hình Transformer đưa ra. LIME giúp chỉ ra những đặc trưng quan trọng nào của peptide đã ảnh hưởng đến quyết định của mô hình, từ đó giải thích lý do tại sao một peptide có thể có khả năng kháng khuẩn cao hoặc thấp. Việc này không chỉ giúp người dùng hiểu rõ hơn về mô hình mà còn làm tăng tính tin cậy của các dự đoán.

**6.6. Đánh giá mô hình:**

Mô hình sẽ được đánh giá thông qua các chỉ số như accuracy, precision, recall, và F1-score để đảm bảo tính chính xác và hiệu quả của dự đoán. Chúng tôi sẽ kiểm tra mô hình trên các bộ dữ liệu kiểm tra và so sánh hiệu quả dự đoán giữa các bộ dữ liệu khác nhau (GenBank và bộ dữ liệu tổng hợp). Việc đánh giá này sẽ giúp chúng tôi xác định những điểm mạnh và hạn chế của mô hình và có các điều chỉnh cần thiết.

**6.7. Tối ưu hóa mô hình:**

Sau khi đánh giá, mô hình sẽ được tối ưu hóa bằng cách điều chỉnh các tham số học của Transformer, đồng thời thử nghiệm các phương pháp cải tiến khác để nâng cao hiệu quả dự đoán. Việc này giúp cải thiện độ chính xác và khả năng tổng quát của mô hình khi áp dụng vào các dữ liệu thực tế.

**7. Đối tượng và phạm vi nghiên cứu**

Đối tượng nghiên cứu của đề tài là các chuỗi peptide có hoặc không có khả năng kháng khuẩn, được thu thập từ các cơ sở dữ liệu sinh học như NCBI GenBank và các nguồn tổng hợp khác. Trong đó, các peptide sẽ được xử lý để có độ dài từ 10 đến 50 acid amin nhưng vẫn giữ được đặc trưng của chuỗi, vì đây là độ dài phổ biến và ổn định trong nghiên cứu liên quan đến hoạt tính sinh học của peptide [25].

Phạm vi nghiên cứu tập trung vào việc phát triển một mô hình học sâu dạng Transformer để dự đoán khả năng kháng khuẩn của chuỗi peptide. Bên cạnh đó, đề tài cũng hướng đến việc giải thích kết quả của mô hình thông qua kỹ thuật Explainable AI (LIME), nhằm tăng tính minh bạch và khả năng tích hợp trong các lĩnh vực như nghiên cứu y sinh học, phát triển thuốc và công nghệ sinh học.

Ngoài ra, đề tài cũng giới hạn phạm vi ở mức dữ liệu in silico (dữ liệu máy tính), chưa mở rộng sang các thử nghiệm sinh học thực tế. Tuy nhiên, kết quả của nghiên cứu này có thể được dùng làm tiền đề để phát triển các công cụ hỗ trợ nghiên cứu chuyên sâu hơn trong tương lai.

**8. Đề xây dựng cho đối tượng nào**

Đề tài này được thực hiện dưới hình thức tự tìm hiểu và nghiên cứu học thuật, không thuộc yêu cầu hoặc đơn đặt hàng cụ thể từ một đơn vị hay doanh nghiệp nào. Tuy nhiên, nội dung và hướng nghiên cứu của đề tài có tính ứng dụng thực tiễn cao, có thể góp phần hỗ trợ các viện nghiên cứu, phòng thí nghiệm, và công ty trong lĩnh vực công nghệ sinh học, y dược và phát triển thuốc trong việc phân tích, đánh giá và thiết kế các chuỗi peptide tiềm năng.

**9. Cấu trúc đồ án**

**MỞ ĐẦU**

**CHƯƠNG 1: TỔNG QUAN TÀI LIỆU**

* 1. Các dự án tương tự trước đây
  2. Cơ sở lý thuyết

**CHƯỜNG 2: PHƯƠNG PHÁP THỰC HIỆN**

2.1. Các yêu cầu kĩ thuật

2.2. Tổng thể về thiết kế phát triển mô hình

2.3. Chi tiết quả trình thực hiện

2.4. Sự lựa chọn phương án thiết kế và công nghệ

2.5. Những tách thức về thuật toán

**CHƯƠNG 3: CÀI ĐẶT THỬ NGHIỆM, KẾT QUẢ VÀ ĐÁNH GIÁ**

3.1. Các thiết kế mô hình giải pháp

3.2. Các khó khăn và thử thách

3.3. Kết quả đạt được

3.4. Hạn chế và đề xuất cải thiện

**CHƯƠNG 4: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN**

**TÀI LIỆU THAM KHẢO**

# **CHƯƠNG 1: TỔNG QUAN TÀI LIỆU**

## **Các dự án tương tự trước đây**

Trong những năm gần đây, cùng với sự phát triển mạnh mẽ của trí tuệ nhân tạo, đặc biệt là học sâu (Deep Learning), nhiều nghiên cứu đã tập trung vào việc khai thác các mô hình máy học để dự đoán chức năng sinh học và các đặc trưng quan trọng trong chuỗi peptide. Dưới đây là một số công trình nổi bật có liên quan trực tiếp đến hướng nghiên cứu của đề tài:

### **Các mô hình học sâu trong dự đoán chức năng peptide**

Trong những năm gần đây, cùng với sự bùng nổ của trí tuệ nhân tạo, đặc biệt là lĩnh vực học sâu (Deep Learning), nhiều mô hình máy học hiện đại đã được áp dụng vào các bài toán trong sinh học phân tử. Một trong những hướng nghiên cứu nổi bật là dự đoán chức năng sinh học của các chuỗi peptide – các phân tử ngắn được cấu thành từ một số lượng nhỏ axit amin và đóng vai trò quan trọng trong nhiều quá trình sinh lý, như kháng khuẩn, điều hòa miễn dịch, hoặc chống ung thư.

Các công trình như DeepACP (Deep Learning for Anticancer Peptide prediction) [47], AntiCP (Anticancer Peptide) [40], iAMPpred (in silico Antimicrobial Peptide predictor) [26], AI4AMP (Artificial Intelligence for Antimicrobial Peptides) [23], AMPScanner (Antimicrobial Peptide Scanner) [43] là những ví dụ tiêu biểu cho việc ứng dụng các mạng nơ-ron sâu trong nhận diện các peptide có chức năng đặc biệt. Các mô hình này thường sử dụng mạng nơ-ron tích chập CNN (Convolutional Neural Network) để phát hiện các mẫu cục bộ trong chuỗi hoặc mạng nơ-ron hồi tiếp RNN (Recurrent Neural Network) để xử lý tính tuầxn tự và bối cảnh trong dữ liệu sinh học. Một số mô hình cũng thử nghiệm với các kiến trúc kết hợp như CNN (Convolutional Neural Network)-RNN (Recurrent Neural Network) hoặc LSTM (Long Short-Term Memory)-BiGRU (Bidirectional Gated Recurrent Unit).

Tuy nhiên, một điểm chung trong hầu hết các nghiên cứu này là chúng phụ thuộc vào các đặc trưng đầu vào được xây dựng thủ công như mã hóa one-hot, đặc trưng sinh hóa (như tính kỵ nước, khối lượng, điện tích), hoặc sử dụng các vector embedding đơn giản để đại diện cho từng axit amin. Dù đạt được độ chính xác tương đối cao, các mô hình vẫn gặp phải một số vấn đề hạn chế, như khó học được mối quan hệ ngữ nghĩa dài hạn giữa các vị trí trong chuỗi peptide, thiếu khả năng tổng quát hóa tốt cho các loại peptide chưa từng gặp trong tập huấn luyện, và đặc biệt là thiếu tính minh bạch – người dùng cuối khó hiểu được lý do mô hình đưa ra kết quả như vậy. Đây chính là một khoảng trống mà đề tài này hướng đến nhằm cải thiện.

### **Tích hợp kiến trúc Transformer trong sinh học phân tử**

Kiến trúc Transformer, lần đầu tiên được giới thiệu trong lĩnh vực xử lý ngôn ngữ tự nhiên NLP (Natural Language Processing) với mô hình Attention is All You Need [42] vào năm 2017, đã nhanh chóng tạo ra bước tiến vượt bậc trong nhiều lĩnh vực liên quan đến dữ liệu chuỗi. Với cơ chế attention giúp mô hình có khả năng nắm bắt các mối quan hệ xa trong chuỗi, Transformer đã chứng minh được sức mạnh vượt trội so với các mô hình RNN (Recurrent Neural Network) truyền thống [12].

Trong sinh học phân tử, một số nghiên cứu tiêu biểu đã khai thác kiến trúc này để học biểu diễn chuỗi protein hoặc peptide. Mô hình TAPE (Tasks Assessing Protein Embeddings) là một trong những công trình đầu tiên huấn luyện Transformer trên tập dữ liệu protein lớn để tạo ra các biểu diễn đặc trưng phục vụ cho nhiều tác vụ như dự đoán cấu trúc, vị trí phân tử trong tế bào, và tương tác giữa các protein. Sau đó, các mô hình như ProtBERT (Protein Bidirectional Encoder Representations from Transformers) [5], ProtTrans (Protein Transformer) [7] tiếp tục mở rộng quy mô, huấn luyện trên hàng trăm triệu chuỗi protein từ các cơ sở dữ liệu như UniProt (Universal Protein Resource) để tạo ra các embedding giàu ngữ nghĩa sinh học.

Tuy vậy, một điểm đáng chú ý là phần lớn các mô hình kể trên được thiết kế và huấn luyện nhằm phục vụ cho chuỗi protein dài và phức tạp, chứ không chuyên biệt hóa cho peptide – vốn có độ dài ngắn hơn và các đặc trưng sinh học riêng biệt. Hơn nữa, hầu hết các nghiên cứu đều dừng lại ở việc tạo ra embedding để phục vụ các bài toán downstream mà chưa kết hợp cùng các phương pháp Explainable AI để giải thích vì sao mô hình lại đưa ra kết quả như vậy. Ngoài ra, các mô hình như ProtBERT (Protein Bidirectional Encoder Representations from Transformers) hay ProtT5 (Protein Text-to-Text Transfer Transformer) [15] thường yêu cầu tài nguyên tính toán lớn, khó triển khai trong các mô hình nghiên cứu có điều kiện hạn chế.

### **Các nghiên cứu tích hợp Explainable AI (XAI) trong sinh học**

Explainable AI [46] còn được gọi là XAI (Explainable Artificial Intelligence) là một nhánh của trí tuệ nhân tạo tập trung vào việc tạo ra các mô hình không chỉ chính xác mà còn có thể giải thích được – nghĩa là người dùng có thể hiểu được lý do đằng sau mỗi quyết định của mô hình. Trong bối cảnh các mô hình học sâu ngày càng phức tạp, XAI (Explainable Artificial Intelligence) trở thành một yếu tố quan trọng, đặc biệt trong các lĩnh vực nhạy cảm như y học, sinh học, hoặc pháp lý.

Trong sinh học, các kỹ thuật như SHAP (SHapley Additive exPlanations) [27], LIME (Local Interpretable Model-Agnostic Explanations) [6], Grad-CAM (Gradient-weighted Class Activation Mapping) [33] đã được áp dụng cho một số bài toán như phân tích biểu hiện gen, chẩn đoán hình ảnh y học, hoặc phân tích dữ liệu RNA-seq (RNA sequencing) [34]. Ví dụ, trong một số nghiên cứu về ung thư, LIME (Local Interpretable Model-Agnostic Explanations) đã được sử dụng để xác định các gene quan trọng nhất ảnh hưởng đến phân loại tế bào ác tính.

Tuy nhiên, việc áp dụng XAI (Explainable Artificial Intelligence) trong phân tích chuỗi peptide còn khá mới và chưa được khai thác rộng rãi. Một phần nguyên nhân đến từ việc các kỹ thuật như Grad-CAM (Gradient-weighted Class Activation Mapping) vốn phát triển cho dữ liệu ảnh, không phù hợp với chuỗi. SHAP (SHapley Additive exPlanations) thì thường dùng với mô hình dạng cây hoặc hồi quy tuyến tính. Trong khi đó, LIME (Local Interpretable Model-Agnostic Explanations) – với khả năng hoạt động như một “hộp đen mở” cho bất kỳ mô hình nào – hứa hẹn là lựa chọn phù hợp để áp dụng trong bài toán giải thích kết quả dự đoán của mô hình Transformer trong lĩnh vực peptide. Tuy nhiên, hiện chưa có nhiều nghiên cứu thực sự kết hợp XAI (Explainable Artificial Intelligence) với các mô hình học sâu hiện đại trong xử lý chuỗi peptide, mở ra cơ hội lớn cho hướng nghiên cứu của đề tài này.

### **Sự khác biệt và điểm mới của đề tài:**

So với các công trình đã nêu, đề tài này mang lại sự kết hợp mới mẻ giữa mô hình học sâu hiện đại (Transformer) và phương pháp giải thích mô hình LIME (Local Interpretable Model-Agnostic Explanations) để giải quyết bài toán dự đoán và phân tích đặc trưng chuỗi peptide. Mục tiêu không chỉ dừng lại ở việc đạt độ chính xác cao trong dự đoán chức năng sinh học của peptide, mà còn cung cấp một công cụ có thể giải thích rõ ràng các yếu tố ảnh hưởng đến quyết định của mô hình. Điều này đặc biệt quan trọng trong bối cảnh nghiên cứu peptide cho mục đích điều trị, nơi các nhà khoa học cần biết được những vị trí axit amin nào là thiết yếu cho chức năng sinh học.

Đề tài tập trung xây dựng một mô hình Transformer chuyên biệt cho chuỗi peptide [25], được huấn luyện trên tập dữ liệu đặc trưng nhằm học được mối quan hệ dài hạn và cấu trúc bối cảnh phức tạp. Sau đó, sử dụng LIME (Local Interpretable Model-Agnostic Explanations) như một công cụ để phân tích ảnh hưởng của từng đặc trưng đầu vào (các vị trí axit amin, đặc trưng sinh hóa) đến đầu ra dự đoán. Không chỉ cải thiện hiệu năng và tính giải thích, đề tài còn có ý nghĩa thực tiễn khi cung cấp nền tảng cho các nghiên cứu phát triển thuốc, khám phá peptide tiềm năng trong điều trị bệnh.

Như vậy, đề tài không chỉ kế thừa các thành tựu trước đó mà còn cải tiến bằng cách tích hợp hai lĩnh vực tiên tiến: học sâu và trí tuệ nhân tạo có thể giải thích, mở ra một hướng tiếp cận toàn diện, hiệu quả và minh bạch hơn trong nghiên cứu chuỗi peptide.

## **Cơ sở lý thuyết**

### **Lý thuyết**

#### **1.2.1.1. Cơ sở lý thuyết kế thừa**

##### **1.2.1.1.1 Học sâu và mô hình Transformer**

Học sâu [20] là một nhánh quan trọng của học máy (Machine Learning) trong đó các mô hình có thể học từ dữ liệu thông qua các mạng nơ-ron nhân tạo (Neural Networks). Học sâu giúp các máy tính tự động nhận dạng các mẫu và đặc trưng trong dữ liệu phức tạp mà không cần sự can thiệp của con người. Trong đề tài này, mô hình học sâu Transformer sẽ được sử dụng. Transformer là một kiến trúc mạng nơ-ron tiên tiến, đặc biệt hữu ích trong việc xử lý các chuỗi dữ liệu dài như chuỗi peptide. Khác với các mô hình RNN (Recurrent Neural Network - Mạng nơ-ron hồi quy) truyền thống [24], Transformer có khả năng xử lý dữ liệu song song và học được các mối quan hệ dài hạn giữa các phần tử trong chuỗi mà không cần phải xử lý tuần tự.

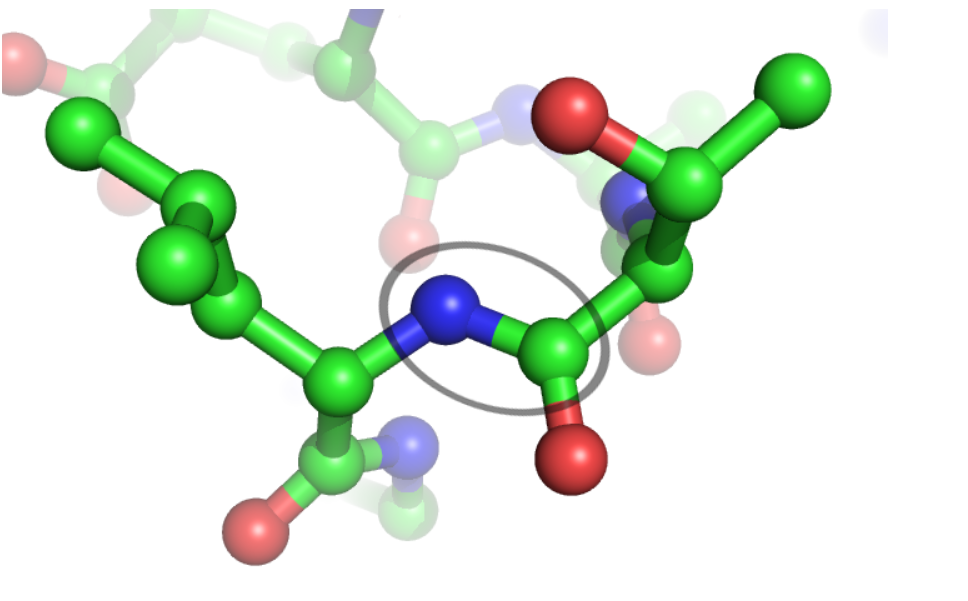
Transformer [14] sử dụng cơ chế Attention [45], cho phép mô hình chú trọng vào các phần quan trọng của chuỗi mà không bị giới hạn bởi chiều dài chuỗi. Điều này rất hữu ích trong việc phân tích chuỗi peptide, nơi các vị trí trong chuỗi có thể có quan hệ mạnh mẽ với nhau dù cách nhau rất xa.

##### **1.2.1.1.2. Explainable AI**

Explainable AI (XAI) là một lĩnh vực trong trí tuệ nhân tạo nhằm phát triển các mô hình có thể giải thích được quá trình đưa ra quyết định của mô hình [13]. Điều này đặc biệt quan trọng trong các ứng dụng y tế, tài chính, hoặc phân tích sinh học, nơi các quyết định của mô hình cần phải có sự minh bạch và giải thích rõ ràng. Trong đề tài này, LIME (Local Interpretable Model-Agnostic Explanations), một phương pháp trong XAI, sẽ được sử dụng để giải thích các dự đoán từ mô hình Transformer.

LIME (Local Interpretable Model-Agnostic Explanations) hoạt động bằng cách tạo ra các mô hình giải thích đơn giản và có thể hiểu được dựa trên các dự đoán của mô hình phức tạp. LIME (Local Interpretable Model-Agnostic Explanations) sẽ giúp người dùng hiểu rõ lý do mà mô hình học sâu đưa ra một dự đoán cụ thể, từ đó giúp người dùng tin tưởng và sử dụng mô hình trong các quyết định quan trọng. Phương pháp này rất phù hợp khi áp dụng vào các chuỗi peptide, vì các dự đoán về các đặc trưng của chuỗi cần có sự giải thích hợp lý.

##### **1.2.1.1.3 Giới thiệu về chuỗi peptide**



Hình 1. 1. Hình ảnh chuỗi peptide

Chuỗi peptide là một dãy các amino acid liên kết với nhau thông qua các liên kết peptide [28]. Mỗi amino acid trong chuỗi peptide có một nhóm amino (-NH₂) ở đầu và một nhóm carboxyl (-COOH) ở cuối. Khi các amino acid kết hợp lại, nhóm carboxyl của một amino acid sẽ kết hợp với nhóm amino của amino acid tiếp theo, tạo thành liên kết peptide và giải phóng nước.

Chuỗi peptide có thể bao gồm từ vài đến hàng nghìn amino acid và là thành phần cơ bản cấu thành các protein. Chúng có thể thực hiện nhiều chức năng sinh học quan trọng trong cơ thể, như tham gia vào các quá trình chuyển hóa, cấu trúc tế bào, tín hiệu sinh học và miễn dịch.

Các chuỗi peptide có thể có dạng không gian phức tạp, với các đoạn chuỗi xoắn hoặc cuộn lại theo một cấu trúc ba chiều nhất định, đóng vai trò quan trọng trong chức năng của chúng. Vì vậy, việc phân tích và hiểu rõ cấu trúc của chuỗi peptide rất quan trọng trong nghiên cứu sinh học phân tử và y học.

Các chuỗi peptide có thể tương tác với nhau và với các phân tử khác trong cơ thể, tạo thành các phức hợp quan trọng hoặc ảnh hưởng đến quá trình sinh lý. Chính vì vậy, việc nghiên cứu và phân tích chuỗi peptide giúp cung cấp cái nhìn sâu sắc về cơ chế hoạt động của các sinh vật sống và các bệnh lý liên quan đến chúng.

#### **1.2.1.2. Phần mới xây dựng**

##### **1.2.1.2.1. Tích hợp Transformer trong phân tích chuỗi peptide**

Mô hình Transformer sẽ được áp dụng trong nghiên cứu này để phân tích và dự đoán các đặc trưng của chuỗi peptide. Đặc trưng chuỗi peptide là một yếu tố quan trọng trong việc hiểu các quá trình sinh học và chức năng của các chuỗi amino acid. Mô hình Transformer sẽ được huấn luyện để nhận diện các mối quan hệ giữa các amino acid trong chuỗi và dự đoán các đặc tính như khả năng tương tác với các phân tử khác, tính ổn định của chuỗi peptide, và các đặc trưng sinh học quan trọng khác.

Một trong những lợi thế lớn của việc sử dụng Transformer là khả năng xử lý chuỗi dài và học được các mối quan hệ xa giữa các phần tử chuỗi, điều mà các mô hình học sâu truyền thống như RNN (Recurrent Neural Network) [12] hoặc CNN (Convolutional Neural Network) [21] không thể làm được. Transformer có cơ chế Attention, giúp mô hình chú trọng vào các phần quan trọng của chuỗi mà không bị giới hạn bởi chiều dài chuỗi. Điều này rất hữu ích khi phân tích chuỗi peptide, vì các mối quan hệ giữa các amino acid có thể xa nhau trong chuỗi nhưng lại có ảnh hưởng mạnh mẽ đến các đặc tính sinh học của peptide.

##### **1.2.1.2.2. Tích hợp LIME để giải thích mô hình Transformer**

Sau khi mô hình Transformer được huấn luyện, LIME (Local Interpretable Model-Agnostic Explanations) sẽ được sử dụng để giải thích các dự đoán của mô hình. Việc sử dụng LIME (Local Interpretable Model-Agnostic Explanations) sẽ giúp người sử dụng có thể hiểu rõ hơn về lý do tại sao mô hình đưa ra các dự đoán cụ thể về chuỗi peptide. LIME (Local Interpretable Model-Agnostic Explanations) hoạt động bằng cách tạo ra các mô hình giải thích đơn giản và có thể hiểu được từ các mô hình phức tạp. Điều này giúp xác định các yếu tố quan trọng nhất trong chuỗi peptide ảnh hưởng đến kết quả dự đoán, chẳng hạn như các amino acid nào trong chuỗi đóng vai trò quan trọng nhất trong việc quyết định tính chất sinh học của peptide.

Phương pháp này không chỉ giúp tăng cường tính minh bạch của mô hình học sâu mà còn giúp các nhà khoa học và chuyên gia trong lĩnh vực sinh học phân tử (Molecular Biology) hiểu rõ hơn về các quá trình sinh học liên quan đến chuỗi peptide. LIME (Local Interpretable Model-Agnostic Explanations) cung cấp cái nhìn sâu sắc và có thể giải thích được về mô hình, điều này rất quan trọng trong các ứng dụng khoa học và nghiên cứu.

### **1.2.2. Kỹ thuật**

**Ngôn ngữ lập trình**: python.

**Mô hình và kĩ thuật tối ưu mô hình**:

* Mô hình: Transformer,
* Kĩ thuật tối ưu mô hình: Keras Tuner

**Phương pháp XAI (Explainable AI)**: dùng kĩ thuật LIME (Local Interpretable Model-Agnostic Explanations).

**Dữ liệu:**

* 1 bộ của GenBank.
* 1 bộ gồm nhiều nguồn khác nhau được đăng trên NCBI (National Center for Biotechnology Information).

**Tiền xử lý dữ liệu:**

* Cào dữ liệu trên trang NCBI (National Center for Biotechnology Information) bằng thư viện BioPython.
* Tách chuỗi peptide theo điều kiện
* Sinh ra các đặc trưng của chuỗi pepdite

**Phần mềm sử dụng:**

* Môi trường tương tác: Jupyter.
* Anaconda: Phân phối phần mềm.

**Framework:** Tensorflow, Flask.

**Phần cứng:**

* Máy tính cá nhân.
* Máy ảo trên nền tảng dịch vụ đám mây AWS (Amazon Web Services) dùng EC2 (Elastic Compute Cloud).

**Công cụ hỗ trợ:** Git, Github, Kaggle

#### **1.2.2.1. Ngôn ngữ lập trình Python**

##### **1.2.2.1.1. Giới thiệu chung**

Python là một ngôn ngữ lập trình bậc cao, thông dịch, được phát triển bởi Guido van Rossum và ra mắt lần đầu vào năm 1991 [30]. Với triết lý thiết kế đề cao sự rõ ràng, đơn giản và dễ đọc, Python nhanh chóng trở thành một trong những ngôn ngữ phổ biến nhất trên thế giới. Cú pháp của Python gần với ngôn ngữ tự nhiên, giúp lập trình viên – kể cả người mới bắt đầu – dễ dàng tiếp cận và làm quen.

Python không chỉ thích hợp với người học mà còn được sử dụng rộng rãi trong các ứng dụng thực tế như phát triển web, xử lý dữ liệu lớn, trí tuệ nhân tạo, học máy, tự động hóa và nhiều lĩnh vực công nghệ khác. Ngôn ngữ này hỗ trợ nhiều mô hình lập trình như hướng đối tượng, hàm và thủ tục.

##### **1.2.2.1.2. Ưu điểm của Python**

Một trong những ưu điểm lớn nhất của Python là cú pháp đơn giản, gọn gàng và dễ hiểu. Cấu trúc của ngôn ngữ này cho phép người viết tập trung vào việc giải quyết vấn đề thay vì tốn nhiều thời gian để xử lý cú pháp. Điều này giúp tăng năng suất lập trình và giảm thiểu lỗi trong quá trình phát triển phần mềm.

Python có hệ sinh thái thư viện và framework phong phú, giúp rút ngắn thời gian phát triển dự án. Chẳng hạn, trong lĩnh vực học máy và trí tuệ nhân tạo, các thư viện như TensorFlow [4], PyTorch [17], Keras [18], Scikit-learn [19] hỗ trợ đầy đủ các thuật toán và công cụ cần thiết. Trong phát triển web, các framework như Django [39] và Flask [11] cung cấp cấu trúc tổ chức mã mạnh mẽ và dễ mở rộng.

Ngôn ngữ này còn có khả năng tương thích tốt với nhiều hệ điều hành như Windows, Linux, và macOS. Python hỗ trợ quản lý bộ nhớ tự động, giúp lập trình viên không phải lo lắng về cấp phát và thu hồi bộ nhớ thủ công như một số ngôn ngữ khác.

Ngoài ra, Python có một cộng đồng lớn mạnh và rất năng động. Điều này đồng nghĩa với việc người dùng có thể dễ dàng tìm thấy tài liệu, ví dụ minh họa, cũng như nhận được sự hỗ trợ từ cộng đồng khi gặp vấn đề kỹ thuật.

##### **1.2.2.1.3. Nhược điểm của Python**

Tuy mang nhiều lợi thế, Python vẫn có một số hạn chế cần lưu ý. Trước tiên, do là ngôn ngữ thông dịch thay vì biên dịch, Python thường có hiệu suất thấp hơn so với các ngôn ngữ như C, C++ hay Java. Trong các ứng dụng yêu cầu xử lý nhanh, như game engine hoặc hệ thống tài chính thời gian thực, Python không phải là lựa chọn lý tưởng.

Một hạn chế khác là mức tiêu thụ bộ nhớ của Python thường cao. Do có cơ chế quản lý bộ nhớ tự động và cấu trúc trừu tượng, các chương trình Python có thể sử dụng nhiều tài nguyên hơn, điều này không phù hợp với những hệ thống có giới hạn về phần cứng.

Python cũng không phổ biến trong lĩnh vực phát triển ứng dụng di động. Dù có thể sử dụng các công cụ như Kivy hoặc BeeWare để viết ứng dụng di động, nhưng hiệu suất và trải nghiệm người dùng thường không tốt bằng các giải pháp chuyên biệt như Java/Kotlin cho Android hoặc Swift cho iOS.

##### **1.2.2.1.4. So sánh với các ngôn ngữ khác**

So với Java, Python có cú pháp ngắn gọn và dễ học hơn nhiều. Trong khi Java yêu cầu khai báo rõ ràng về kiểu dữ liệu, lớp, phương thức,… thì Python cho phép người dùng viết mã một cách linh hoạt hơn mà vẫn đảm bảo hiệu quả. Điều này giúp Python thường được lựa chọn cho các dự án cần phát triển nhanh, trong khi Java lại thích hợp cho các hệ thống lớn cần tính ổn định và bảo trì dài hạn.

So với C++, Python đơn giản hơn rất nhiều về cú pháp và cách tiếp cận. C++ là ngôn ngữ mạnh về hiệu suất, thích hợp cho các hệ thống nhúng, game hoặc phần mềm yêu cầu tối ưu tài nguyên. Tuy nhiên, C++ đòi hỏi người dùng quản lý bộ nhớ thủ công và xử lý lỗi phức tạp hơn, điều mà Python đã đơn giản hóa đi rất nhiều.

Nói cách khác, nếu như Java và C++ phù hợp với các hệ thống cần độ ổn định, hiệu suất cao và phát triển lâu dài, thì Python lại phát huy sức mạnh trong các dự án cần tính linh hoạt, sáng tạo và triển khai nhanh.

##### **1.2.2.1.5. Ứng dụng thực tiễn của Python**

Python là ngôn ngữ được sử dụng trong rất nhiều lĩnh vực công nghệ hiện nay. Trong trí tuệ nhân tạo và học máy, Python gần như là ngôn ngữ mặc định nhờ vào các thư viện hỗ trợ mạnh mẽ như TensorFlow, Keras, Scikit-learn và PyTorch. Trong khoa học dữ liệu, Python giúp xử lý và phân tích dữ liệu hiệu quả nhờ các công cụ như Pandas và NumPy, đồng thời có thể trực quan hóa dữ liệu bằng thư viện Matplotlib hoặc Seaborn.

Trong phát triển web, Python được sử dụng để xây dựng các ứng dụng web linh hoạt, bảo mật thông qua các framework như Flask hoặc Django. Bên cạnh đó, Python còn được sử dụng rộng rãi trong tự động hóa tác vụ, viết script cho hệ thống, phát triển phần mềm nhúng, hoặc điều khiển các thiết bị IoT như Raspberry Pi.

Sự đa dạng trong ứng dụng và tính dễ sử dụng khiến Python không chỉ phổ biến trong giới kỹ sư phần mềm mà còn trong các lĩnh vực khác như tài chính, sinh học, giáo dục hay nghiên cứu khoa học.

##### **1.2.2.1.6. Lý do lựa chọn Python trong dự án**

Trong bối cảnh hiện nay, khi trí tuệ nhân tạo AI (Artificial Intelligence) ngày càng đóng vai trò quan trọng trong việc tự động hóa, phân tích dữ liệu và đưa ra các quyết định thông minh, việc lựa chọn ngôn ngữ lập trình phù hợp là một yếu tố then chốt quyết định đến chất lượng và hiệu quả của dự án. Python được lựa chọn trong dự án này vì nó là một trong những ngôn ngữ lập trình mạnh mẽ, phổ biến và thân thiện nhất đối với các ứng dụng sử dụng trí tuệ nhân tạo.

Lý do đầu tiên và nổi bật nhất là Python sở hữu một hệ sinh thái thư viện hỗ trợ AI (Artificial Intelligence) rất phong phú và mạnh mẽ. Các thư viện như TensorFlow, PyTorch, Keras, Scikit-learn hay OpenCV đã trở thành những công cụ không thể thiếu cho các kỹ sư dữ liệu và nhà nghiên cứu trong lĩnh vực học máy, thị giác máy tính, xử lý ngôn ngữ tự nhiên và học sâu. Những thư viện này không chỉ dễ sử dụng mà còn được cập nhật liên tục từ cộng đồng mã nguồn mở và các tập đoàn công nghệ hàng đầu.

Bên cạnh đó, cú pháp đơn giản, rõ ràng và gần gũi với ngôn ngữ tự nhiên của Python giúp tăng tốc độ phát triển mô hình và giảm thiểu lỗi lập trình, nhất là trong các bài toán AI (Artificial Intelligence) vốn đã phức tạp và đòi hỏi thử nghiệm nhiều lần. Python còn hỗ trợ tốt cho việc xử lý dữ liệu, một công đoạn quan trọng không thể thiếu trước khi huấn luyện mô hình AI (Artificial Intelligence), nhờ các thư viện như NumPy, Pandas và Matplotlib.

#### **1.2.2.2. Mô hình và kĩ thuật tối ưu mô hình**

##### **1.2.2.2.1. Mô hình Transformer**

###### **1.2.2.2.1.1. Giới thiệu mô hình Transformer**

Mô hình Transformer được giới thiệu lần đầu tiên vào năm 2017 trong bài báo “Attention is All You Need” của Vaswani và các cộng sự. Đây là một bước tiến lớn trong lĩnh vực NLP (Natural Language Processing) và DL (Deep Learning). Transformer đã thay thế kiến trúc RNN (Recurrent Neural Network) bằng cách sử dụng Attention Mechanism (Cơ chế chú ý), cho phép xử lý đồng thời toàn bộ chuỗi đầu vào thay vì tuần tự như trước.

Mục tiêu chính của Transformer là cải thiện hiệu suất huấn luyện và hỗ trợ khả năng song song hóa. Ngoài ra, mô hình còn giúp khắc phục những hạn chế cố hữu của các mô hình dựa trên RNN (Recurrent Neural Network) và LSTM (Long Short-Term Memory) [10] như khó khăn trong việc ghi nhớ thông tin từ các chuỗi dài.

Transformer từ đó đã trở thành nền tảng cho nhiều mô hình lớn hơn và phức tạp hơn như BERT, GPT và T5, góp phần cách mạng hóa lĩnh vực trí tuệ nhân tạo hiện đại.

###### **1.2.2.2.1.2. Cấu trúc tổng quát của Transformer**

Mô hình Transformer bao gồm hai thành phần chính là bộ mã hóa (Encoder) và bộ giải mã (Decoder).

* Bộ mã hóa (Encoder) nhận đầu vào là một chuỗi các vector biểu diễn từ, sau đó trích xuất thông tin và chuyển thành biểu diễn ẩn.
* Bộ giải mã (Decoder) tiếp nhận biểu diễn ẩn này và sinh ra chuỗi đầu ra.

Mỗi Encoder và Decoder đều bao gồm nhiều lớp giống nhau được xếp chồng lên nhau. Một lớp Encoder gồm hai khối thành phần chính:

* MHSA (Multi-Head Self-Attention Mechanism) giúp mô hình học được các mối quan hệ giữa các phần tử trong chuỗi.
* FFNN (Feed-Forward Neural Network) áp dụng độc lập cho từng vị trí trong chuỗi.

Trong khi đó, một lớp Decoder ngoài hai khối trên còn thêm một khối chú ý liên kết với đầu ra của Encoder, giúp Decoder tham chiếu được toàn bộ thông tin mã hóa trước đó.

Để bảo toàn thông tin thứ tự, Transformer còn bổ sung thêm Positional Encoding (Mã hóa vị trí).

###### **1.2.2.2.1.3. Cơ chế chú ý trong Transformer**

Cơ chế cốt lõi trong Transformer là Attention Mechanism (Cơ chế chú ý), đặc biệt là SDPA (Scaled Dot-Product Attention).

Cơ chế này cho phép mô hình “tập trung” vào các phần tử quan trọng trong chuỗi thay vì xử lý theo tuần tự. Cụ thể, mỗi phần tử trong chuỗi sẽ được ánh xạ thành ba vector:

* Q (Query)
* K (Key)
* V (Value)

Đầu tiên, mô hình tính điểm chú ý giữa các Q (Query) và K (Key) thông qua phép nhân ma trận. Sau đó, kết quả được chia cho một hệ số chuẩn hóa (căn bậc hai của kích thước chiều K (Key)) để tránh giá trị quá lớn, rồi áp dụng hàm Softmax nhằm chuẩn hóa thành xác suất. Cuối cùng, mô hình dùng xác suất này để kết hợp các vector V (Value), tạo ra đầu ra đã được “chú ý” phù hợp. Điều này giúp mô hình tự động xác định được đâu là thông tin quan trọng trong toàn bộ chuỗi đầu vào.

###### **1.2.2.2.1.4. Cơ chế chú ý đa đầu**

Để cải thiện khả năng học tập nhiều loại quan hệ khác nhau, Transformer sử dụng MHA (Multi-Head Attention).

Thay vì chỉ có một phép Attention duy nhất, mô hình thực hiện song song nhiều phép Attention (gọi là các “head”), mỗi head học được một kiểu quan hệ khác nhau giữa các phần tử.

Cụ thể:

* Các vector đầu vào được tách thành nhiều phần nhỏ hơn.
* Mỗi phần nhỏ được đưa vào một cơ chế Attention độc lập.
* Sau khi tính toán, các kết quả từ các head được kết hợp lại thông qua phép nối (concatenation) và chiếu lại về kích thước ban đầu.

Nhờ đó, mô hình vừa có thể học các quan hệ ngắn hạn, vừa có thể học các quan hệ dài hạn trong cùng một lần truyền dữ liệu.

###### **1.2.2.2.1.5. Ưu điểm của Transformer**

Mô hình Transformer mang lại nhiều ưu điểm vượt trội:

* Khả năng song song hóa: Do không phải xử lý theo chuỗi như RNN (Recurrent Neural Network), Transformer có thể xử lý toàn bộ chuỗi đầu vào cùng lúc, giúp tăng tốc độ huấn luyện đáng kể.
* Giải quyết hiện tượng quên lâu: Các mô hình truyền thống như RNN (Recurrent Neural Network) và LSTM (Long Short-Term Memory) dễ bị quên mất thông tin xa trong chuỗi dài. Transformer, nhờ Attention Mechanism (Cơ chế chú ý), có thể duy trì và xử lý các mối quan hệ xa một cách hiệu quả.
* Khả năng mở rộng: Transformer dễ dàng mở rộng lên mô hình lớn hơn như BERT, GPT, T5... phục vụ cho nhiều nhiệm vụ khác nhau trong lĩnh vực NLP (Natural Language Processing) và thị giác máy tính.

Nhờ những đặc điểm này, Transformer đã trở thành một trong những mô hình cốt lõi trong nghiên cứu và ứng dụng trí tuệ nhân tạo hiện đại.

##### **1.2.2.2.2. Kĩ thuật tối ưu mô hình Keras Tuner**

###### **1.2.2.2.2.1. Giới thiệu về Keras Tuner**

Keras Tuner là một thư viện của Keras trong TensorFlow được thiết kế để hỗ trợ việc tìm kiếm hyperparameter cho các mô hình học sâu [29]. Việc lựa chọn hyperparameter là một phần quan trọng trong quá trình xây dựng mô hình, vì các tham số này có thể ảnh hưởng lớn đến hiệu quả của mô hình. Keras Tuner giúp tự động hóa quá trình tìm kiếm tham số tối ưu, thay vì phải thử nghiệm thủ công.

###### **1.2.2.2.2.2. Kỹ thuật tối ưu trong Keras Tuner**

Keras Tuner sử dụng ba phương pháp chính để tối ưu hóa mô hình:

* **Random Search:** Trong phương pháp này, các hyperparameter được chọn ngẫu nhiên trong một không gian tìm kiếm được định nghĩa trước. Random Search có thể tìm được các tham số tốt trong một không gian rộng mà không cần thử mọi sự kết hợp của các tham số.
* **Hyperband:** Hyperband là một kỹ thuật tối ưu hóa hiệu quả sử dụng thuật toán multi-armed bandit. Kỹ thuật này phân bổ các tài nguyên tính toán (ví dụ: số lần lặp) một cách thông minh cho các thử nghiệm mô hình để nhanh chóng tìm ra các mô hình tiềm năng nhất. Hyperband sử dụng chiến lược thử nghiệm các mô hình với nhiều cấu hình khác nhau và tinh chỉnh các tham số dựa trên hiệu quả của mỗi mô hình trong các lần huấn luyện.
* **Bayesian Optimization:** Phương pháp Bayesian Optimization giúp mô hình học từ các thử nghiệm trước đó để dự đoán và tìm ra các tham số tối ưu. Phương pháp này thường được sử dụng khi không gian tham số rất lớn và các thử nghiệm có chi phí tính toán cao. Bayesian Optimization tối ưu hóa một hàm mục tiêu dựa trên các quan sát trước đó để quyết định tham số tiếp theo cần thử.

###### **1.2.2.2.2.3. Các thành phần chính của Keras Tuner**

* **HyperModel:** HyperModel là một lớp trong Keras Tuner, đại diện cho mô hình mà bạn muốn tối ưu. Bạn có thể tạo một lớp HyperModel để định nghĩa mô hình của mình và sử dụng các hyperparameter thay vì giá trị cố định.
* **Hyperparameter:** Hyperparameter là các tham số của mô hình, chẳng hạn như số lượng lớp ẩn, số lượng neuron trong mỗi lớp, tỷ lệ học, v.v. Những tham số này cần được tối ưu hóa để đạt hiệu quả cao nhất cho mô hình. Trong Keras Tuner, bạn có thể sử dụng các đối tượng như Hyperband, Random Search, hoặc Bayesian Optimization để tìm các giá trị tối ưu cho các tham số này.
* **Tuner:** Tuner là đối tượng chính trong Keras Tuner chịu trách nhiệm tìm kiếm và tối ưu hóa các hyperparameter của mô hình. Bạn có thể cấu hình Tuner với một phương pháp tối ưu hóa và khởi tạo Tuner. Ví dụ, bạn có thể chọn Random Search, Hyperband, hoặc Bayesian Optimization.

###### **1.2.2.2.2.4. Cách sử dụng Keras Tuner**

Để sử dụng Keras Tuner trong một mô hình học sâu, bạn cần thực hiện các bước cơ bản sau:

* **Định nghĩa mô hình**: Xây dựng mô hình học sâu và thay thế các tham số cố định bằng các tham số có thể thay đổi được, sử dụng các hàm của Keras Tuner như Hyperparameter để định nghĩa phạm vi tìm kiếm.
* **Khởi tạo Tuner**: Lựa chọn phương pháp tối ưu hóa và khởi tạo Tuner. Ví dụ, bạn có thể chọn Random Search, Hyperband, hoặc Bayesian Optimization.
* **Chạy quá trình tối ưu hóa**: Sử dụng tuner.search() để bắt đầu quá trình tìm kiếm các tham số tối ưu cho mô hình. Quá trình này sẽ huấn luyện nhiều mô hình với các cấu hình khác nhau và đánh giá hiệu suất của chúng.
* **Đánh giá kết quả**: Sau khi quá trình tìm kiếm hoàn tất, bạn có thể lấy được mô hình tối ưu với các hyperparameter đã được tối ưu hóa.

###### **1.2.2.2.2.5. Lợi ích của Keras Tuner**

* **Tiết kiệm thời gian**: Thay vì phải thử nghiệm thủ công, Keras Tuner giúp tự động hóa quá trình tối ưu hóa, tiết kiệm thời gian cho người nghiên cứu.
* **Tìm kiếm các tham số hiệu quả hơn**: Với các phương pháp tối ưu hóa như Random Search, Hyperband, hay Bayesian Optimization, Keras Tuner giúp tìm kiếm các tham số phù hợp trong không gian tham số rộng.
* **Dễ dàng sử dụng**: Thư viện dễ dàng tích hợp với Keras, không cần viết mã phức tạp cho các thuật toán tối ưu hóa.

#### **1.2.2.3. Phương pháp XAI (Explainable AI)**

##### **1.2.2.3.1. Giới thiệu**

Explainable Artificial Intelligence (gọi tắt là XAI) là một lĩnh vực nghiên cứu nhằm làm cho các mô hình trí tuệ nhân tạo trở nên dễ hiểu và có thể giải thích được cho con người. Khi các mô hình học sâu (deep learning) ngày càng trở nên phức tạp và mạnh mẽ, nhu cầu giải thích vì sao mô hình đưa ra một dự đoán cụ thể càng trở nên quan trọng, đặc biệt trong các lĩnh vực yêu cầu tính minh bạch cao như y tế, tài chính, và pháp lý.

Explainable Artificial Intelligence giúp:

* Tăng độ tin cậy vào mô hình.
* Phát hiện lỗi hoặc thiên lệch trong mô hình.
* Tuân thủ các quy định pháp lý về minh bạch dữ liệu.

##### **1.2.2.3.2. Các phương pháp Explainable Artificial Intelligence phổ biến**

Một số phương pháp nổi bật trong Explainable Artificial Intelligence gồm:

* **LIME** **(Local Interpretable Model-agnostic Explanations):** phương pháp giải thích cục bộ bằng cách xây dựng các mô hình đơn giản xung quanh từng dự đoán.
* **SHAP (SHapley Additive exPlanations):** phương pháp dựa trên lý thuyết trò chơi để phân tích đóng góp của từng đặc trưng.
* **Grad-CAM** **(Gradient-weighted Class Activation Mapping):** dùng cho các mô hình học sâu trong lĩnh vực thị giác máy tính để trực quan hóa vùng ảnh ảnh hưởng nhiều nhất đến quyết định của mô hình.

##### **1.2.2.3.3. Phương pháp LIME (Local Interpretable Model-agnostic Explanations)**

###### **1.2.2.3.3.1. Mục tiêu của LIME**

LIME nhằm mục đích giải thích từng dự đoán riêng lẻ của mô hình bằng cách:

* Xây dựng một mô hình đơn giản (thường là mô hình tuyến tính) xung quanh một điểm dữ liệu cụ thể.
* Phản ánh cách các đặc trưng đầu vào ảnh hưởng đến quyết định đầu ra của mô hình.

Thay vì cố gắng giải thích toàn bộ mô hình phức tạp, LIME tập trung vào việc giải thích một vùng nhỏ xung quanh từng điểm cụ thể.

###### **1.2.2.3.3.2. Nguyên lý hoạt động**

Quy trình hoạt động của LIME gồm các bước:

* **Chọn điểm cần giải thích**: Một mẫu đầu vào cụ thể được chọn để phân tích.
* **Tạo mẫu dữ liệu xung quanh**: Các biến thể nhỏ (perturbations) của dữ liệu gốc được sinh ra bằng cách thay đổi ngẫu nhiên các giá trị đặc trưng.
* **Gán nhãn cho các mẫu mới**: Sử dụng mô hình gốc để dự đoán nhãn cho các mẫu mới này.
* **Xây dựng mô hình đơn giản**: Huấn luyện một mô hình tuyến tính hoặc cây quyết định nhỏ trên tập dữ liệu đã được gán nhãn, sao cho mô hình này phản ánh được cách mô hình gốc ra quyết định tại khu vực đó.
* **Đưa ra giải thích**: Phân tích trọng số của mô hình đơn giản để xác định đặc trưng nào quan trọng nhất trong quyết định.

###### **1.2.2.3.3.3. Ưu điểm của LIME**

* **Mô hình-agnostic**: LIME không phụ thuộc vào loại mô hình học máy đang sử dụng. Bạn có thể áp dụng nó cho bất kỳ mô hình nào (ví dụ như hồi quy logistic, mạng nơ-ron sâu, cây quyết định...).
* **Giải thích cục bộ**: LIME tập trung vào giải thích dự đoán tại từng điểm cụ thể thay vì giải thích toàn bộ mô hình, nên mang tính cá nhân hóa cao.
* **Hiển thị rõ ràng**: Các giải thích do LIME tạo ra thường dễ hiểu cho con người thông qua các biểu đồ hoặc danh sách đặc trưng quan trọng.

###### **1.2.2.3.3.4. Hạn chế của LIME**

* **Độ tin cậy phụ thuộc vào khu vực lân cận**: Nếu khu vực được sinh mẫu không đại diện đúng cho phân phối dữ liệu thực tế, giải thích có thể bị sai lệch.
* **Tốn thời gian tính toán**: Việc sinh mẫu và huấn luyện mô hình giải thích cho từng điểm dữ liệu có thể tiêu tốn nhiều thời gian nếu tập dữ liệu lớn.
* **Không hoàn toàn chính xác**: Vì chỉ mô phỏng mô hình tại khu vực nhỏ, nên LIME không thể nắm bắt toàn bộ hành vi phức tạp của mô hình gốc.

#### **1.2.2.4. Dữ liệu chuỗi peptide của các loài vi sinh vật**

Dữ liệu chuỗi peptide của các loài vi sinh vật là nguồn tài nguyên quý giá, đóng vai trò quan trọng trong nhiều lĩnh vực như sinh học phân tử, sinh học tiến hóa, y học và công nghệ sinh học. Các peptide này có thể là các đoạn của protein hoặc là protein hoàn chỉnh, đảm nhiệm nhiều chức năng sinh học khác nhau trong cơ thể vi sinh vật, bao gồm enzyme, yếu tố độc lực, hoặc thành phần cấu trúc tế bào.

Trong nghiên cứu, một trong những nguồn dữ liệu đáng tin cậy và được sử dụng rộng rãi là Cơ sở dữ liệu Trung tâm Thông tin Công nghệ Sinh học Quốc gia (National Center for Biotechnology Information – NCBI). Đây là một cơ quan trực thuộc Thư viện Y khoa Quốc gia Hoa Kỳ (U.S. National Library of Medicine), cung cấp kho dữ liệu sinh học đồ sộ và được cập nhật thường xuyên. Các chuỗi peptide từ vi sinh vật được thu thập, chú thích và lưu trữ trong các cơ sở dữ liệu của NCBI.

Việc sử dụng dữ liệu peptide từ Cơ sở dữ liệu Trung tâm Thông tin Công nghệ Sinh học Quốc gia không chỉ giúp đảm bảo độ chính xác và tính toàn vẹn của dữ liệu, mà còn tạo điều kiện thuận lợi cho việc phân tích, so sánh, và khai thác thông tin trong các ứng dụng nghiên cứu chuyên sâu.

Dữ liệu chuỗi peptide của các loài vi sinh vật, được thu thập từ Cơ sở dữ liệu Trung tâm Thông tin Công nghệ Sinh học Quốc gia (National Center for Biotechnology Information – NCBI), bao gồm ba thành phần chính: Sequence, Label, và ID.

* **Sequence**: là chuỗi axit amin đại diện cho peptide hoặc protein. Đây là thông tin cốt lõi, phản ánh cấu trúc tuyến tính của phân tử peptide và là cơ sở để phân tích chức năng sinh học, mối quan hệ tiến hóa, cũng như thiết kế các nghiên cứu tiếp theo.
* **Label**: thể hiện thông tin phân loại hoặc đặc điểm chức năng liên quan đến peptide, ví dụ như phân biệt giữa peptide kháng khuẩn và không kháng khuẩn, hoặc các nhóm chức năng khác nhau trong cơ thể vi sinh vật.
* **ID**: là mã định danh duy nhất được gán cho mỗi chuỗi peptide, giúp truy xuất dữ liệu một cách dễ dàng và đảm bảo tính nhất quán khi tích hợp với các nguồn dữ liệu khác.

Nhờ cấu trúc dữ liệu rõ ràng và được chuẩn hóa từ một nguồn uy tín như NCBI, bộ dữ liệu này trở thành nền tảng đáng tin cậy cho các nghiên cứu về chức năng protein, phát triển thuốc, cũng như phân tích hệ gen và hệ protein của các loài vi sinh vật.

#### **1.2.2.5. Tiền xử lý dữ liệu**

Tiền xử lý dữ liệu là bước quan trọng để đảm bảo rằng nguồn dữ liệu đầu vào cho mô hình học máy hoặc học sâu có chất lượng cao, phù hợp với yêu cầu của bài toán nghiên cứu. Trong nghiên cứu này, dữ liệu chuỗi peptide của các loài vi sinh vật được tiền xử lý qua ba giai đoạn chính: thu thập dữ liệu từ các cơ sở dữ liệu sinh học, phân loại chuỗi peptide theo tiêu chí nhất định, và sinh ra các đặc trưng (feature extraction) từ các chuỗi đã xử lý.

##### **1.2.2.5.1. Thu thập dữ liệu từ NCBI**

Dữ liệu peptide được cào từ trang web NCBI (National Center for Biotechnology Information) [9] thông qua việc sử dụng thư viện BioPython. BioPython là một thư viện mã nguồn mở mạnh mẽ được thiết kế để xử lý các dữ liệu sinh học, đặc biệt là dữ liệu liên quan đến trình tự DNA, RNA và protein. Thư viện này cung cấp giao diện để truy cập và tải dữ liệu từ nhiều cơ sở dữ liệu quốc tế như GenBank, RefSeq (Reference Sequence Database), SwissProt, Protein Data Bank (PDB), DNA Data Bank of Japan (DDBJ), European Molecular Biology Laboratory (EMBL), và Protein Information Resource (PIR).

Việc sử dụng BioPython giúp tự động hóa quy trình tải dữ liệu peptide, hạn chế lỗi thủ công và tăng tốc độ thu thập dữ liệu. Dữ liệu thu thập bao gồm các trình tự peptide cùng với thông tin mô tả liên quan như chức năng sinh học, tính chất hóa học, nguồn gốc sinh vật học, và các chú thích đặc biệt (annotation).

Trong quá trình thu thập, dữ liệu được lọc sơ bộ để loại bỏ các bản ghi không đầy đủ, các trình tự peptide có độ dài quá ngắn hoặc quá dài so với yêu cầu nghiên cứu, đồng thời chỉ giữ lại những chuỗi có thông tin rõ ràng về chức năng hoặc nguồn gốc.

##### **1.2.2.5.2. Tách chuỗi peptide theo điều kiện**

Sau khi thu thập đầy đủ dữ liệu chuỗi peptide, bước tiếp theo là xử lý và tách các chuỗi dựa trên nhãn đã có sẵn và một số điều kiện liên quan đến đặc tính sinh hóa. Mỗi đoạn được tách ra từ chuỗi peptide sẽ được gán nhãn là AMP (peptide có khả năng kháng khuẩn) hoặc nAMP (peptide không có khả năng kháng khuẩn), tùy thuộc vào nhãn ban đầu của chuỗi đó.

Một yêu cầu quan trọng trong quá trình xử lý dữ liệu là mỗi chuỗi peptide đầu vào phải có độ dài từ 10 đến 50 amino acid. Nếu một chuỗi dài hơn 50 amino acid, nó sẽ được chia nhỏ thành các đoạn có độ dài phù hợp trong khoảng này. Sau khi chia nhỏ, nếu đoạn nào đáp ứng các tiêu chí sinh hóa nhất định, thì đoạn đó sẽ được giữ lại để dùng trong quá trình huấn luyện mô hình học sâu [38].

Việc tách và lọc như vậy giúp loại bỏ các đoạn không phù hợp, đồng thời vẫn giữ nguyên nhãn và đặc điểm sinh hóa quan trọng của chuỗi ban đầu. Nhờ đó, dữ liệu sau xử lý trở nên đồng nhất hơn, dễ sử dụng cho mô hình học sâu, và giúp mô hình dự đoán chính xác hơn về khả năng kháng khuẩn của peptide.

##### **1.2.2.5.3. Sinh ra các đặc trưng của chuỗi peptide**

###### **1.2.2.5.3.1. Khái niệm về Global Protein Sequence Descriptor**

Đây là một phương pháp mã hóa các trình tự protein dưới dạng số, dựa trên các đặc tính lý hóa của các amino acid. Mục đích là chuyển đổi thông tin chuỗi peptide thành các giá trị số để các mô hình học sâu có thể phân tích và dự đoán các tính chất đặc trưng của peptide, như khả năng gấp cuộn (folding) hoặc khả năng kháng khuẩn [22].

###### **1.2.2.5.3.2. Ứng dụng**

* Dự đoán cấu trúc và chức năng chuỗi peptide: Phương pháp này có thể giúp các mô hình học sâu dự đoán các tính chất đặc trưng của chuỗi peptide như cách thức gấp cuộn hoặc khả năng tương tác với các phân tử khác.
* Xác định peptide kháng khuẩn (AMPs): Cải thiện khả năng nhận diện các peptide có khả năng kháng khuẩn.
* Phân loại peptide: Hỗ trợ phân loại peptide thành các nhóm có tính chất tương tự.

#### **1.2.2.6. Phần mềm sử dụng**

Trong quá trình nghiên cứu và phát triển các mô hình học máy, học sâu, việc sử dụng các phần mềm và công cụ hỗ trợ là rất quan trọng để đảm bảo hiệu quả và tính chính xác của công việc. Hai phần mềm chính được sử dụng trong môi trường nghiên cứu này là Jupyter và Anaconda.

##### **1.2.2.6.1. Môi trường tương tác Jupyter**

Jupyter là một môi trường phát triển tích hợp (IDE) mã nguồn mở được sử dụng rộng rãi trong các lĩnh vực khoa học dữ liệu và học máy. Điểm mạnh của Jupyter là khả năng hỗ trợ tạo và chia sẻ tài liệu dạng notebook (sổ tay), trong đó có thể chứa mã nguồn lập trình, văn bản mô tả, biểu đồ trực quan và phương trình toán học. Môi trường này đặc biệt hữu ích cho các nhà nghiên cứu và lập trình viên khi họ cần chạy và thử nghiệm mã trực tiếp, đồng thời có thể hiển thị kết quả ngay lập tức. Việc tích hợp các thư viện khoa học dữ liệu phổ biến như pandas, matplotlib hay tensorflow với Jupyter Notebook giúp việc phân tích và trực quan hóa dữ liệu trở nên dễ dàng và hiệu quả hơn.

##### **1.2.2.6.2. Phân phối phần mềm Anaconda**

Anaconda là một phân phối phần mềm mã nguồn mở mạnh mẽ, được thiết kế để hỗ trợ các công việc phân tích dữ liệu và phát triển các mô hình học máy. Phân phối này bao gồm một bộ công cụ phong phú như NumPy, pandas, scikit-learn, và Matplotlib, tất cả đều được tối ưu hóa để thực hiện các phép toán khoa học và xử lý dữ liệu. Một trong những tính năng đặc biệt của Anaconda là conda, một công cụ quản lý gói và môi trường, cho phép người dùng cài đặt, cập nhật, và quản lý các thư viện phần mềm mà không làm ảnh hưởng đến hệ thống chính. Anaconda cũng giúp người dùng dễ dàng quản lý nhiều môi trường lập trình khác nhau, mỗi môi trường có thể chứa các phiên bản thư viện và công cụ khác nhau, điều này rất hữu ích trong các dự án đòi hỏi sự tương thích giữa các gói phần mềm.

#### **1.2.2.7. Framework**

Trong quá trình phát triển các mô hình học máy và học sâu, việc chọn lựa framework phù hợp là rất quan trọng để tối ưu hóa hiệu quả công việc. Hai framework chính được sử dụng trong dự án này là TensorFlow và Flask.

##### **1.2.2.7.1. Tensorflow**

TensorFlow là một framework mã nguồn mở do Google phát triển, chủ yếu được sử dụng trong lĩnh vực học máy và học sâu [4]. TensorFlow hỗ trợ xây dựng và triển khai các mô hình học máy, từ các mô hình đơn giản đến các mô hình phức tạp, bao gồm mạng nơ-ron sâu. Framework này cung cấp các công cụ và thư viện mạnh mẽ, giúp việc xử lý và huấn luyện các mô hình trở nên dễ dàng hơn, đồng thời hỗ trợ việc triển khai chúng trên nhiều nền tảng, từ các máy chủ đến các thiết bị di động

##### **1.2.2.7.2 Flask**

Flask là một micro framework cho Python, được sử dụng để xây dựng các ứng dụng web [11]. Nó được thiết kế để đơn giản và dễ sử dụng, cho phép các nhà phát triển nhanh chóng tạo ra các ứng dụng web nhẹ và có thể mở rộng. Flask cung cấp các công cụ cần thiết để xây dựng các API (Application Programming Interface) và tương tác với cơ sở dữ liệu, nhưng không yêu cầu cấu hình phức tạp hay các quy tắc nghiêm ngặt. Flask thường được sử dụng trong việc xây dựng các dịch vụ web và API (Application Programming Interface), và nó dễ dàng tích hợp với các thư viện học máy như TensorFlow để triển khai các mô hình học máy trực tiếp trong ứng dụng web.

#### **1.2.2.8. Phần cứng**

##### **1.2.2.8.1. Máy tính cá nhân**

Máy tính cá nhân là một thiết bị sử dụng cho cá nhân hoặc hộ gia đình, thường được sử dụng cho các công việc như xử lý văn bản, tính toán, giải trí và các tác vụ khác. Các máy tính này có thể là máy tính để bàn hoặc laptop và là công cụ chủ yếu cho công việc hàng ngày của người sử dụng.

##### **1.2.2.8.2. Máy ảo trên nền tảng dịch vụ đám mây AWS (Amazon Web Services)**

Máy ảo trên nền tảng dịch vụ đám mây AWS (Amazon Web Services) dùng EC2 (Elastic Compute Cloud) là một giải pháp lưu trữ và tính toán đám mây do Amazon cung cấp [3]. AWS là dịch vụ đám mây phổ biến, cung cấp các dịch vụ như lưu trữ, cơ sở hạ tầng tính toán, và các công cụ khác hỗ trợ phát triển mô hình. EC2 (Elastic Compute Cloud) là một dịch vụ của AWS cho phép người dùng tạo và quản lý các máy chủ ảo (gọi là instance) trên đám mây, giúp tiết kiệm chi phí và tăng tính linh hoạt cho việc triển khai mô hình mà không cần phải sở hữu phần cứng vật lý.

#### **1.2.2.9. Công cụ hỗ trợ**

##### **1.2.2.9.1. Git**

Git là một hệ thống quản lý phiên bản phân tán, được sử dụng để theo dõi các thay đổi trong mã nguồn của các dự án phần mềm [16]. Git cho phép lưu trữ và theo dõi lịch sử phát triển của mã nguồn, giúp các lập trình viên có thể làm việc độc lập mà không sợ mất dữ liệu khi thực hiện các thay đổi. Công cụ này rất hữu ích trong việc phát triển phần mềm theo nhóm, vì nó giúp đồng bộ hóa công việc của từng thành viên, đảm bảo mã nguồn luôn được cập nhật và không có xung đột. Git cũng cho phép phục hồi các phiên bản trước đó của dự án khi cần thiết.

##### **1.2.2.9.2. Github**

GitHub là một dịch vụ trực tuyến sử dụng Git để lưu trữ và quản lý mã nguồn [41]. Đây là công cụ giúp lập trình viên lưu trữ mã nguồn của dự án trên nền tảng đám mây, tạo ra không gian cho việc cộng tác giữa các thành viên trong nhóm. GitHub hỗ trợ các tính năng như pull request (yêu cầu hợp nhất mã), issues (quản lý các vấn đề của dự án), và CI/CD (Continuous Integration/Continuous Deployment - Tích hợp liên tục và Triển khai liên tục), giúp cải thiện hiệu quả làm việc nhóm và nâng cao quy trình phát triển phần mềm. GitHub cũng cung cấp các công cụ để người dùng dễ dàng chia sẻ và quản lý các kho mã nguồn của mình, đồng thời có thể kết nối với nhiều công cụ khác nhau để phát triển phần mềm hiệu quả.

##### **1.2.2.9.3. Kaggle**

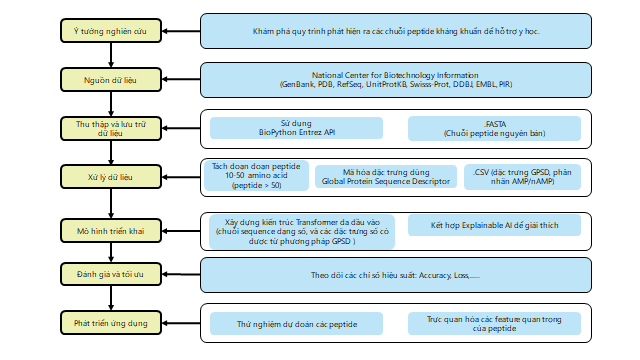
Kaggle là một nền tảng trực tuyến chuyên cung cấp các bộ dữ liệu cho các nhà khoa học dữ liệu và những người đam mê học máy [44]. Trên Kaggle, người dùng có thể lưu trữ và chia sẻ các bộ dữ liệu của mình, tham gia vào các cuộc thi học máy để giải quyết các bài toán thực tế, và học hỏi từ các mô hình và dự án của người khác. Kaggle là công cụ tuyệt vời cho việc nghiên cứu và thử nghiệm các mô hình học máy, đồng thời cung cấp môi trường tính toán trực tuyến để thực hiện các bài toán phân tích dữ liệu mà không cần cài đặt phần mềm cồng kềnh. Nền tảng này cũng hỗ trợ việc phát triển các mô hình học máy bằng cách cung cấp các công cụ, tài liệu học tập, và cộng đồng sẵn sàng giúp đỡ.

# **CHƯƠNG 2: PHƯƠNG PHÁP THỰC HIỆN**

## **2.1. Các yêu cầu kĩ thuật**

* **Thu thập và xử lý dữ liệu:** Dữ liệu chuỗi peptide được thu thập từ NCBI. Chuỗi peptide dài trên 50 axit amin sẽ được tách theo điều kiện sinh hóa nhất định. Sau đó, các đặc trưng chuỗi peptide được trích xuất các đặc trưng bằng Global Protein Sequence Descriptor và chuỗi peptide sẽ được chuyển thành dạng số thông qua phương pháp Tokenizer và cả hai sẽ được làm dữ liệu đầu vào cho mô hình học sâu Transformer.
* **Xây dựng mô hình học sâu Transformer:** Mô hình học sâu Transformer sẽ được sử dụng để phân tích và dự đoán các đặc trưng của chuỗi peptide. Đây là một mô hình mạnh mẽ, thích hợp cho các bài toán chuỗi và phân tích dữ liệu dạng chuỗi như chuỗi peptide.
* **Áp dụng Explainable AI vào mô hình Transformer:** Các phương pháp Explainable AI, như LIME (Local Interpretable Model-agnostic Explanations), sẽ được áp dụng để giải thích kết quả dự đoán của mô hình Transformer. Điều này giúp người dùng có thể hiểu được các yếu tố ảnh hưởng đến quyết định của mô hình, từ đó cải thiện khả năng tin cậy và minh bạch của mô hình học sâu.
* **Xây dựng giao diện:** Một giao diện người dùng trực quan sẽ được thiết kế để người dùng có thể dễ dàng nhập vào chuỗi peptide và nhận kết quả dự đoán từ mô hình học sâu. Giao diện này cũng sẽ hiển thị các giải thích từ mô hình, giúp người dùng hiểu rõ hơn về quá trình ra quyết định của hệ thống.
* **Sử dụng AWS EC2 (Amazon Elastic Compute Cloud):** Mô hình sẽ được triển khai trên dịch vụ điện toán đám mây AWS. Việc sử dụng AWS EC2 giúp chương trình được chạy trên GPU, có khả năng mở rộng, và hiệu suất cao trong quá trình xử lý và dự đoán chuỗi peptide, đáp ứng nhu cầu tính toán lớn của mô hình học sâu.

## **2.2. Tổng thể về thiết kế phát triển mô hình**



Sơ đồ 2. 1. Sơ đồ tổng thể thiết kế phát triển mô hình

Nghiên cứu được thực hiện với mục tiêu khám phá và phát hiện ra các chuỗi peptide có nguồn gốc từ vi sinh vật, nhằm tìm kiếm những chuỗi peptide có tiềm năng kháng khuẩn cao, góp phần hỗ trợ và cải thiện các liệu pháp điều trị trong lĩnh vực y học hiện đại. Trong bối cảnh các loại vi khuẩn ngày càng kháng lại nhiều loại kháng sinh truyền thống, việc tìm ra những peptide mới có khả năng tiêu diệt hoặc ức chế sự phát triển của vi khuẩn đang là một hướng nghiên cứu quan trọng và mang tính cấp thiết. Từ thực tiễn đó, đề tài hướng đến việc tận dụng sức mạnh của trí tuệ nhân tạo, đặc biệt là các mô hình học sâu, để phát hiện những đặc điểm có giá trị sinh học từ dữ liệu peptide, đồng thời tự động hóa quy trình phân tích và dự đoán.

Nguồn dữ liệu sử dụng trong nghiên cứu được lấy từ các cơ sở dữ liệu sinh học nổi tiếng và có độ tin cậy cao, đặc biệt là từ Trung tâm Thông tin Công nghệ Sinh học Quốc gia Hoa Kỳ (National Center for Biotechnology Information). Các cơ sở dữ liệu này đóng vai trò quan trọng trong việc hỗ trợ các nhà khoa học trên toàn cầu trong việc phân tích và nghiên cứu sinh học phân tử. Các cơ sở dữ liệu cụ thể bao gồm: GenBank (GenBank sequence database), một kho dữ liệu lớn chứa các trình tự di truyền của nhiều loài sinh vật, là công cụ cơ bản trong nghiên cứu di truyền học và tiến hóa; Protein Data Bank (Protein Data Bank), nơi lưu trữ thông tin về cấu trúc ba chiều của các phân tử protein, giúp hiểu rõ hơn về chức năng và tương tác của các protein trong cơ thể; RefSeq (Reference Sequence Database), cung cấp các trình tự tham chiếu chuẩn cho các gene và bộ gen, đóng vai trò quan trọng trong việc xác định các biến thể di truyền và nghiên cứu bệnh học; UniProt Knowledgebase (UniProt Knowledgebase), một nền tảng toàn diện cung cấp thông tin chi tiết về các protein, từ cấu trúc đến chức năng sinh học của chúng, hỗ trợ việc phát triển các liệu pháp y tế mới; Swiss-Prot (Swiss Protein Database), một cơ sở dữ liệu chứa các thông tin về protein được chú thích thủ công, đảm bảo tính chính xác và đáng tin cậy của thông tin; Ngân hàng Dữ liệu Trình tự AND Nhật Bản (DNA Data Bank of Japan), một kho dữ liệu quan trọng phục vụ cho nghiên cứu di truyền học và sinh học phân tử ở khu vực Châu Á và trên toàn cầu; Phòng thí nghiệm Sinh học Phân tử châu Âu (European Molecular Biology Laboratory), một tổ chức nghiên cứu lớn cung cấp các cơ sở dữ liệu liên quan đến sinh học phân tử, giúp tiến hành các nghiên cứu về sự tương tác của các phân tử sinh học và các cơ chế bệnh lý; và Cơ sở dữ liệu Thông tin Protein (Protein Information Resource), một nguồn tài nguyên quan trọng cung cấp thông tin chi tiết về các protein và các chức năng của chúng trong các quá trình sinh học. Đây là những kho dữ liệu được giới khoa học toàn cầu tin tưởng và thường xuyên sử dụng cho các nghiên cứu trong lĩnh vực sinh học phân tử. Các cơ sở dữ liệu này không chỉ cung cấp thông tin về cấu trúc và chức năng của các phân tử sinh học mà còn giúp các nhà nghiên cứu giải mã các cơ chế tế bào, từ đó mở ra cơ hội phát triển các phương pháp điều trị và liệu pháp y tế mới. Những dữ liệu này được sử dụng rộng rãi trong các nghiên cứu về di truyền học, protein học, và dược học, đóng góp vào sự tiến bộ của y học và các ngành khoa học liên quan.

Để truy xuất và thu thập dữ liệu từ các nguồn trên, nghiên cứu sử dụng công cụ BioPython Entrez API – một thư viện mạnh mẽ được phát triển dành riêng cho các tác vụ xử lý dữ liệu sinh học. Các chuỗi peptide được lấy về ở định dạng FASTA, là định dạng chuẩn quốc tế trong việc lưu trữ và trao đổi trình tự sinh học dưới dạng văn bản, thuận tiện cho quá trình xử lý và phân tích sau này.

Sau khi dữ liệu được thu thập, quá trình xử lý và tiền xử lý dữ liệu được tiến hành nhằm đảm bảo tính nhất quán và phù hợp cho việc huấn luyện mô hình. Cụ thể, các chuỗi peptide có độ dài từ 10 đến 50 axit amin được lựa chọn để phân tích, vì đây là khoảng độ dài phổ biến và phù hợp với các mô hình học máy [38]. Các chuỗi peptide này được mã hóa đặc trưng bằng phương pháp Global Protein Sequence Descriptor, một phương pháp trích xuất các đặc điểm toàn cục của chuỗi protein, giúp mô hình học được thông tin sinh học tiềm ẩn trong trình tự axit amin [22]. Sau khi mã hóa, dữ liệu được lưu trữ dưới định dạng tệp văn bản có cấu trúc CSV, trong đó mỗi dòng đại diện cho một chuỗi peptide cùng với các đặc trưng của nó, đồng thời kèm theo nhãn phân loại cho biết chuỗi đó có phải là peptide kháng khuẩn (antimicrobial peptide) hay không.

Trên cơ sở dữ liệu đã xử lý, một mô hình học sâu được triển khai để tiến hành học và dự đoán. Mô hình được lựa chọn là mạng nơ-ron Transformer – một kiến trúc hiện đại và mạnh mẽ, vốn nổi bật trong việc xử lý dữ liệu dạng chuỗi như ngôn ngữ tự nhiên, và cũng rất phù hợp để áp dụng vào trình tự sinh học. Đầu vào của mô hình là các chuỗi peptide đã được chuyển đổi thành dạng số hóa, kết hợp với các đặc trưng được trích xuất. Để đảm bảo tính minh bạch và khả năng giải thích trong y sinh, mô hình còn được tích hợp kỹ thuật Trí tuệ nhân tạo có thể giải thích (Explainable Artificial Intelligence – XAI), cho phép phân tích và làm rõ các yếu tố ảnh hưởng đến quyết định của mô hình, giúp các nhà nghiên cứu hiểu rõ hơn về các yếu tố quyết định hoạt tính sinh học của peptide.

Giai đoạn tiếp theo là đánh giá và tối ưu mô hình. Trong quá trình này, các chỉ số đánh giá như độ chính xác (Accuracy), độ mất mát (Loss), và các chỉ số phụ trợ khác được sử dụng để đo lường hiệu quả của mô hình. Dựa trên kết quả đánh giá, mô hình sẽ được điều chỉnh về kiến trúc cũng như các siêu tham số nhằm đạt được hiệu suất dự đoán cao nhất. Quy trình huấn luyện và đánh giá được lặp lại nhiều lần để đảm bảo mô hình có khả năng tổng quát tốt và tránh hiện tượng học thuộc dữ liệu.

Sau khi mô hình được huấn luyện và đánh giá thành công, mô hình sẽ được đưa vào giai đoạn ứng dụng. Cụ thể, mô hình sẽ được sử dụng để dự đoán những chuỗi peptide mới chưa được xác định trước, từ đó phát hiện ra các peptide tiềm năng có hoạt tính kháng khuẩn. Đồng thời, kết quả của mô hình cũng sẽ được trực quan hóa thông qua việc biểu diễn các đặc trưng quan trọng có ảnh hưởng lớn đến quyết định phân loại. Điều này không chỉ giúp xác minh tính đúng đắn của mô hình, mà còn tạo điều kiện cho các nhà nghiên cứu dễ dàng tiếp cận và ứng dụng kết quả vào các nghiên cứu tiếp theo về dược học, y học hoặc công nghệ sinh học.

### **2.2.1. Các thành phần chính**

#### **2.2.1.1 Thu thập và xử lý dữ liệu**

Quá trình thu thập dữ liệu bắt đầu bằng việc lấy các chuỗi peptide từ cơ sở dữ liệu NCBI, nơi các chuỗi peptide đã được gán nhãn sẵn để xác định liệu chúng có khả năng kháng khuẩn hay không. Sau khi thu thập dữ liệu, mỗi chuỗi peptide sẽ được kiểm tra độ dài. Các chuỗi peptide có độ dài lớn hơn 50 amino acid sẽ tiếp tục được xử lý, trong khi các chuỗi có độ dài từ 10 đến 50 amino acid sẽ được giữ nguyên mà không qua bất kỳ bước xử lý nào.

Đối với những chuỗi peptide dài hơn 50 amino acid, chúng sẽ được tách ra thành các đoạn theo các điều kiện sinh hóa cụ thể. Cụ thể, các đoạn chuỗi có điện tích tổng (net charge) lớn hơn hoặc bằng 3 và tỷ lệ amino acid không phân cực (non-polar) lớn hơn hoặc bằng 60% sẽ được chọn vào nhóm peptide kháng khuẩn. Ngược lại, các chuỗi có điện tích tổng nhỏ hơn 3 và tỷ lệ amino acid không phân cực nhỏ hơn hoặc bằng 30% sẽ được phân loại vào nhóm peptide không kháng khuẩn. Những đoạn chuỗi nào trong các chuỗi dài đáp ứng được các điều kiện này sẽ được tách ra và giữ lại để tạo đặc trưng cho quá trình huấn luyện mô hình.

Sau khi các chuỗi peptide qua bước xử lý, chúng sẽ được chuyển hóa thành các đặc trưng số bằng phương pháp Global Protein Sequence Descriptor. Điều này giúp tạo ra các đặc trưng số hóa, từ đó làm dữ liệu đầu vào cho mô hình học sâu. Bên cạnh đó, chuỗi peptide cũng sẽ được chuyển thành dạng số thông qua thư viện Tokenizer để có thể đưa vào mô hình Transformer để học. Vậy chuỗi peptide được chuyển thành dạng số và các đặc trưng số hóa được trích xuất từ chuỗi sẽ là đầu vào cho mô hình Transformer

#### **2.2.1.2 Xây dựng mô hình học sâu Transformer**

Xây dựng mô hình học sâu sử dụng kiến trúc Transformer có đầu vào là các đặc trưng số hóa được trích xuất từ chuỗi peptide bằng phương pháp Global Protein Sequence Descriptor và kèm theo chuỗi peptide đó được chuyển về dạng số. Transformer là một mô hình học sâu phổ biến trong các tác vụ xử lý ngôn ngữ tự nhiên và đã được chứng minh hiệu quả trong việc học các mối quan hệ dài hạn giữa các phần tử trong chuỗi dữ liệu. Trong bài toán này, Transformer sẽ được sử dụng để học các mẫu trong chuỗi peptide, từ đó phân loại chúng thành peptide kháng khuẩn hoặc không kháng khuẩn dựa trên đoạn chuỗi đã qua được bước xử lý sau đó số hóa và các đặc trưng đã được trích xuất từ chuỗi bằng phương pháp Global Protein Sequence Descriptor.

Mô hình Transformer sẽ được huấn luyện trên các chuỗi peptide đã được mã hóa, giúp mô hình học được các mối quan hệ giữa các amino acid và các đặc tính sinh hóa của chuỗi. Quá trình huấn luyện sẽ sử dụng các phương pháp tối ưu hóa như Adam để điều chỉnh các tham số của mô hình và cải thiện độ chính xác trong việc phân loại chuỗi peptide.

#### **2.2.1.3 Áp dụng Explainable AI vào mô hình Transformer**

Để tăng tính minh bạch và khả năng giải thích của mô hình học sâu, một trong những kỹ thuật quan trọng cần được áp dụng là Explainable AI (XAI). Đặc biệt, LIME (Local Interpretable Model-agnostic Explanations) sẽ được sử dụng để giải thích các dự đoán của mô hình Transformer. LIME là một phương pháp XAI giúp cung cấp các giải thích rõ ràng về cách mà mô hình học sâu đưa ra quyết định, bằng cách phân tích sự ảnh hưởng của từng đặc trưng đầu vào đến kết quả dự đoán.

Bằng cách áp dụng LIME, chúng ta có thể hiểu được tại sao mô hình phân loại một chuỗi peptide là kháng khuẩn hay không kháng khuẩn, từ đó giúp người dùng và nhà nghiên cứu có cái nhìn rõ ràng hơn về các yếu tố quan trọng trong chuỗi peptide ảnh hưởng đến kết quả phân loại. Điều này đặc biệt hữu ích trong việc cải thiện độ tin cậy và minh bạch của mô hình trong các ứng dụng thực tế.

#### **2.2.1.4. Xây dựng giao diện**

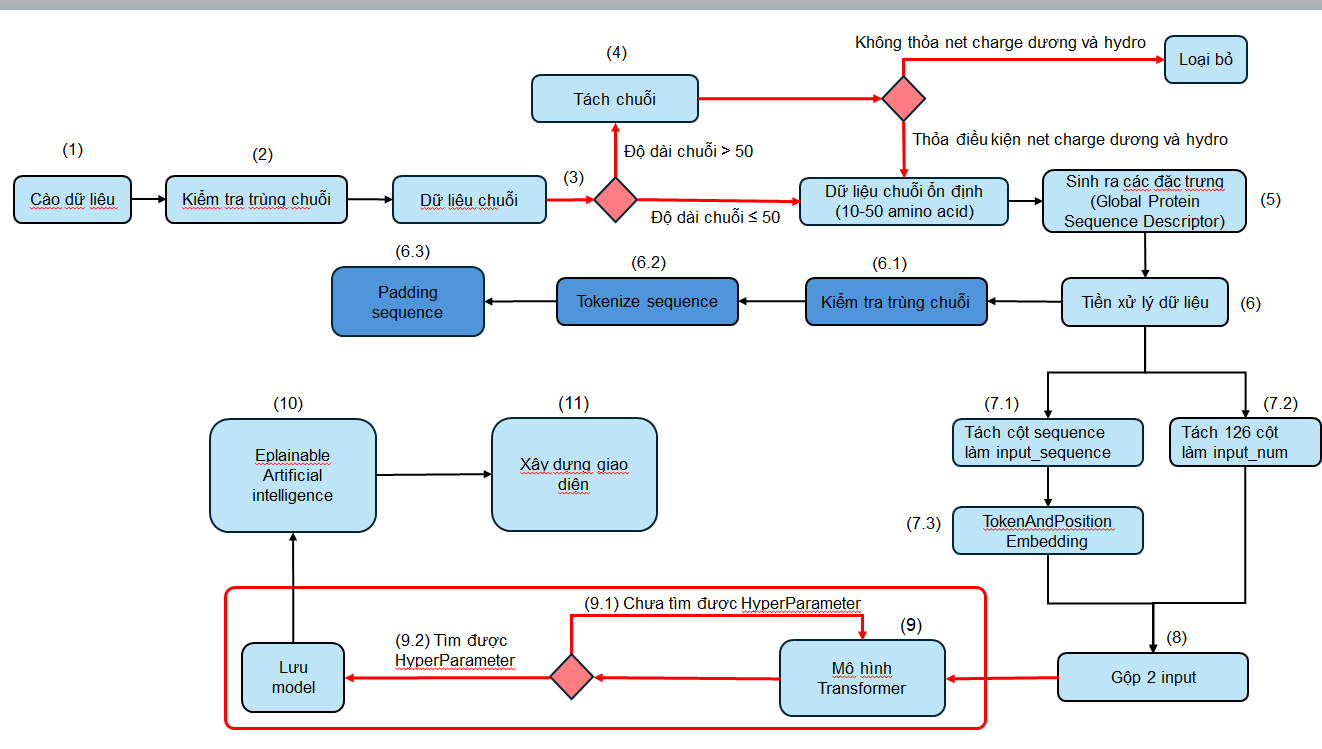
Để người dùng có thể dễ dàng tương tác và sử dụng mô hình, một giao diện người dùng sẽ được xây dựng. Giao diện này sẽ cho phép người dùng tải lên các chuỗi peptide, nhập thông tin liên quan và nhận kết quả phân loại từ mô hình học sâu. Giao diện cũng sẽ cung cấp các giải thích trực quan về kết quả dự đoán thông qua LIME (Local Interpretable Model-agnostic Explanations), giúp người dùng hiểu được lý do tại sao một peptide được phân loại vào nhóm kháng khuẩn hay không kháng khuẩn.

Giao diện người dùng sẽ được thiết kế thân thiện và dễ sử dụng, với các tính năng như: nhập chuỗi peptide, xem kết quả phân loại, và nhận các giải thích chi tiết từ LIME về các yếu tố ảnh hưởng đến quyết định của mô hình. Việc xây dựng giao diện giúp nâng cao trải nghiệm người dùng và tạo ra một công cụ hữu ích cho các nhà nghiên cứu và chuyên gia trong lĩnh vực peptide kháng khuẩn.

### **2.2.2. Những mối liên hệ của các thành phần**

* **Thu thập và xử lý dữ liệu:** cung cấp dữ liệu đầu vào cho mô hình phát triển. Sau khi thu thập các chuỗi peptide từ cơ sở dữ liệu NCBI, các chuỗi được xử lý theo các điều kiện sinh học và chuyển đổi thành các đặc trưng số hóa. Đầu ra của bước này là dữ liệu đã qua xử lý, sẵn sàng để được đưa vào mô hình học sâu.
* **Mô hình học sâu Transformer:** nhận dữ liệu đầu vào từ bước xử lý dữ liệu (chuỗi peptide đã chuyển thành số và các đặc trưng số). Mô hình này thực hiện nhiệm vụ phân loại chuỗi peptide thành nhóm kháng khuẩn hoặc không kháng khuẩn, dựa trên các đặc trưng học được trong quá trình huấn luyện.
* **Explainable AI (LIME):** tương tác trực tiếp với mô hình Transformer để giải thích kết quả phân loại. LIME giúp làm rõ lý do tại sao một chuỗi peptide được phân loại vào nhóm kháng khuẩn hoặc không kháng khuẩn, từ đó cung cấp khả năng giải thích và hiểu biết sâu hơn về quá trình ra quyết định của mô hình.
* **Giao diện người dùng:** kết nối với mô hình học sâu và LIME. Người dùng tương tác với mô hình Transformer qua giao diện này để tải chuỗi peptide, nhận kết quả phân loại và giải thích chi tiết về lý do phân loại của mô hình, giúp tăng cường tính minh bạch và khả năng sử dụng.

## **2.3. Chi tiết quá trình thực hiện**



Sơ đồ 2. 2. Sơ đồ chi tiết quá trình thực hiện

Đề tài được triển khai thông qua một quy trình gồm nhiều bước kế tiếp nhau, nhằm đảm bảo tính chặt chẽ, khoa học và phù hợp với đặc trưng dữ liệu sinh học. Cụ thể, quy trình bao gồm các giai đoạn chính như sau:

Đầu tiên, dữ liệu chuỗi peptide được thu thập từ dữ liệu từ nguồn nCBI (Bước 1). Sau đó, dữ liệu được kiểm tra và loại bỏ các chuỗi trùng lặp nhằm đảm bảo tính đa dạng và tránh hiện tượng học lệch trong mô hình (Bước 2).

Tiếp theo, các chuỗi sẽ được kiểm tra độ dài (Bước 3). Những chuỗi có độ dài vượt quá 50 amino acid sẽ được chia tách thành các đoạn 10 – 50 amino acid theo điều kiện sinh hóa net charge và hydro, trong khi những chuỗi dưới ngưỡng này sẽ được giữ nguyên. Một bước kiểm tra sinh hóa cũng được tiến hành (Bước 4) nhằm loại bỏ các chuỗi không đáp ứng được điều kiện net charge dương và hydro, đảm bảo chỉ những chuỗi có tính ổn định sinh học mới được đưa vào mô hình.

Sau khi lọc dữ liệu, các chuỗi peptide sẽ được chuyển sang bước trích xuất đặc trưng sinh học (Bước 5), sử dụng các phương pháp mô tả toàn cục (Global Protein Sequence Descriptor) để tạo ra một bộ đặc trưng đại diện cho từng chuỗi. Các đặc trưng này có thể bao gồm các thông số như tính kỵ nước, tính phân cực, khối lượng phân tử ,…

Giai đoạn tiếp theo là tiền xử lý dữ liệu (Bước 6). Ở đây, các chuỗi sẽ tiếp tục được kiểm tra trùng lặp, mã hóa thành chuỗi số nguyên (tokenization), và chuẩn hóa độ dài thông qua kỹ thuật padding để đảm bảo tính nhất quán đầu vào cho mô hình.

Trong bước chuẩn bị đầu vào (Bước 7), dữ liệu được chia thành hai phần: một phần là chuỗi số nguyên của peptide (input\_sequence), phần còn lại là 126 đặc trưng sinh học (input\_num). Phần chuỗi sẽ được đưa qua lớp Token and Position Embedding để giữ lại thông tin về vị trí từng amino acid cũng như biểu diễn một kí tự dưới dạng mảng số nhiều chiều.

Hai loại dữ liệu đầu vào này sau đó được gộp lại (Bước 8), tạo thành một đầu vào hợp nhất cho mô hình học sâu. Mô hình được sử dụng là kiến trúc Transformer (Bước 9), một mô hình nổi bật trong xử lý chuỗi nhờ khả năng nắm bắt quan hệ dài hạn giữa các phần tử trong chuỗi.

Trong quá trình huấn luyện, nếu chưa tìm được các siêu tham số tối ưu, mô hình sẽ được chuyển sang bước tìm kiếm siêu tham số (hyperparameter tuning) bằng các công cụ như Keras Tuner (Bước 9.2), sau đó quay lại huấn luyện với các tham số mới.

Khi mô hình đạt được hiệu quả mong muốn, bước tiếp theo là áp dụng trí tuệ nhân tạo có thể giải thích (Explainable Artificial Intelligence – XAI) (Bước 10). Trong đề tài này, phương pháp LIME (Local Interpretable Model-agnostic Explanations) được sử dụng để giải thích kết quả dự đoán, cho phép hiểu rõ các yếu tố sinh học nào đã đóng góp nhiều nhất đến đầu ra của mô hình.

Cuối cùng, mô hình được tích hợp vào một giao diện người dùng (Bước 11), giúp người dùng dễ dàng nhập chuỗi peptide, nhận kết quả dự đoán, và xem các giải thích trực quan về đặc trưng chuỗi – từ đó phục vụ nghiên cứu sinh học, y học và ứng dụng thực tiễn.

## **2.4. Sự lựa chọn phương án thiết kế và công nghệ**

### **2.4.1. Công nghệ**

Mô hình Transformer: Được lựa chọn để xử lý và học biểu diễn từ chuỗi peptide. Transformer nổi bật với khả năng học mối quan hệ dài hạn giữa các thành phần trong chuỗi nhờ cơ chế tự chú ý (self-attention), giúp mô hình hiểu sâu về cấu trúc và tính chất của peptide.

LIME (Local Interpretable Model-agnostic Explanations): Là một kĩ thuật trong Explainable AI, được sử dụng để giải thích các dự đoán của mô hình. LIME tạo ra các giải thích cục bộ giúp hiểu rõ ảnh hưởng của từng đặc trưng peptide đến kết quả dự đoán, từ đó tăng độ minh bạch và độ tin cậy của mô hình.

Python: Ngôn ngữ chính cho toàn bộ quy trình, từ tiền xử lý dữ liệu, xây dựng mô hình, đến phân tích kết quả. Python cung cấp hệ sinh thái thư viện mạnh mẽ cho học sâu (TensorFlow, HuggingFace Transformers) và Explainable AI.

TensorFlow: Framework mạnh mẽ để xây dựng và huấn luyện mô hình học sâu, hỗ trợ tối ưu hóa hiệu suất trên GPU và dễ dàng triển khai mô hình phức tạp như Transformer.

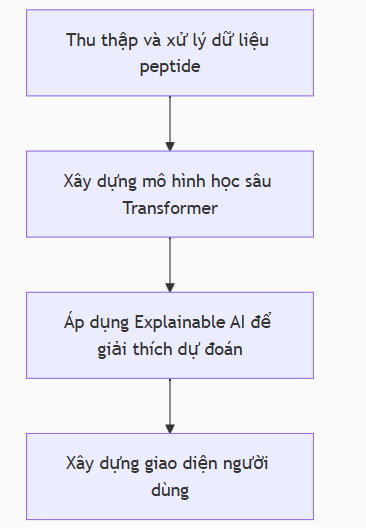
AWS EC2 (Amazon Elastic Compute Cloud) (máy chủ GPU): Được sử dụng để huấn luyện mô hình trên các máy chủ ảo hóa được trang bị GPU chuyên dụng như NVIDIA Tesla V100 hoặc A100. Việc sử dụng EC2 GPU giúp rút ngắn đáng kể thời gian huấn luyện, đảm bảo khả năng thử nghiệm nhiều cấu hình mô hình và bộ siêu tham số khác nhau.

### **2.4.2. Phương án thiết kế**

#### **2.4.2.1. Tổng quan phương án thiết kế**

1. Thu thập và xử lý dữ liệu peptide.
2. Xây dựng mô hình học sâu Transformer.
3. Áp dụng Explainable AI để giải thích dự đoán.
4. Xây dựng giao diện người dùng để vận hành và khai thác mô hình

#### **2.4.2.2. Chi tiết phương án thiết kế**



Sơ đồ 2. 3. Sơ đồ chi tiết phương án thiết kế

1. Thu thập và xử lý dữ liệu

* **Thu thập**: Dữ liệu peptide được lấy từ cơ sở dữ liệu NCBI, với các chuỗi đã được gán nhãn là kháng khuẩn hoặc không kháng khuẩn.
* **Tiền xử lý**:
  + Kiểm tra độ dài chuỗi peptide:
    - Chuỗi dài từ 10–50 amino acid: giữ nguyên.
    - Chuỗi dài >50 amino acid: tách thành các đoạn nhỏ dựa trên tiêu chí sinh hóa.
  + Các đoạn chuỗi được chọn theo hai điều kiện:
    - Điện tích tổng ≥ 3 và tỷ lệ amino acid không phân cực ≥ 60% → Nhóm kháng khuẩn.
    - Điện tích tổng < 3 và tỷ lệ amino acid không phân cực ≤ 30% → Nhóm không kháng khuẩn.
* **Biểu diễn dữ liệu**:
  + Sử dụng phương pháp **Global Protein Sequence Descriptor** để chuyển chuỗi peptide thành đặc trưng số hóa.
  + Đồng thời sử dụng **Tokenizer** để mã hóa trực tiếp chuỗi peptide thành dạng số, phù hợp với đầu vào của mô hình học sâu.

1. Xây dựng mô hình học sâu Transformer

* **Cấu trúc**: Xây dựng mô hình Transformer gồm nhiều lớp encoder để học mối quan hệ giữa các amino acid trong chuỗi peptide.
* **Đầu vào**:
  + Các đặc trưng số hóa từ phương pháp Global Protein Sequence Descriptor.
  + Các chuỗi peptide đã được mã hóa số bằng Tokenizer.
* **Huấn luyện**:
  + Sử dụng thuật toán tối ưu **Adam** để điều chỉnh trọng số.
  + Tối ưu hóa theo hướng phân loại đúng peptide thành nhóm kháng khuẩn hoặc không kháng khuẩn.
* **Đầu ra**: Dự đoán nhãn phân loại cho chuỗi peptide đầu vào.

1. Áp dụng Explainable AI (XAI) cho mô hình Transformer

* **Kỹ thuật sử dụng**:
  + Tích hợp phương pháp **LIME (Local Interpretable Model-agnostic Explanations)** để giải thích quyết định của mô hình.
* **Mục tiêu**:
  + Phân tích ảnh hưởng của từng đặc trưng hoặc thành phần của chuỗi peptide đến kết quả phân loại.
  + Tăng độ minh bạch và độ tin cậy của mô hình trong các mô hình thực tế.

1. Xây dựng giao diện người dùng

* **Chức năng giao diện**:
  + Cho phép người dùng tải lên chuỗi peptide hoặc nhập liệu trực tiếp.
  + Hiển thị kết quả phân loại (kháng khuẩn / không kháng khuẩn).
  + Cung cấp giải thích trực quan từ LIME về các yếu tố quyết định của mô hình.
* **Thiết kế giao diện**:
  + Thân thiện, dễ sử dụng, hỗ trợ các đối tượng người dùng khác nhau (nhà nghiên cứu, chuyên gia sinh học phân tử,...).

### **2.4.3. Lý do lựa chọn công nghệ và phương án thiết kế**

**Lý do chọn mô hình Transformer**: Mô hình **Transformer** được chọn do khả năng vượt trội trong việc xử lý và hiểu các mối quan hệ dài hạn trong chuỗi dữ liệu. Trong trường hợp của chuỗi peptide, các mối quan hệ này rất quan trọng để xác định đặc điểm của peptide và từ đó phân loại chính xác. Cơ chế tự chú ý (self-attention) của Transformer cho phép mô hình tập trung vào các phần quan trọng của chuỗi, giúp hiểu sâu hơn về tính chất của peptide, điều này rất quan trọng khi làm việc với chuỗi sinh học.

**Lý do chọn Explainable AI – LIME: LIME** được chọn để giải thích các quyết định của mô hình vì tính hiệu quả của nó trong việc tạo ra các giải thích cục bộ. Trong các ứng dụng y sinh và nghiên cứu, độ minh bạch và khả năng giải thích mô hình rất quan trọng. Khi mô hình dự đoán một peptide là “kháng khuẩn”, việc cung cấp thông tin giải thích về lý do tại sao mô hình đưa ra dự đoán đó giúp tăng độ tin cậy của mô hình, đặc biệt là trong các lĩnh vực nghiên cứu.

**Lý do chọn Python và TensorFlow: Python** được chọn vì đây là ngôn ngữ phổ biến và mạnh mẽ trong lĩnh vực học máy và học sâu, với thư viện phong phú như **TensorFlow** và **HuggingFace Transformers,** giúp việc xây dựng và huấn luyện mô hình Transformer trở nên dễ dàng và hiệu quả. Python cũng hỗ trợ tốt việc tích hợp các thư viện cho Explainable AI và tiền xử lý dữ liệu. **TensorFlow** được chọn vì đây là framework học sâu mạnh mẽ, hỗ trợ tối ưu hóa hiệu suất trên GPU, giúp việc huấn luyện mô hình phức tạp như Transformer trở nên nhanh chóng và hiệu quả.

**Lý do chọn AWS EC2 (Amazon Elastic Compute Cloud) (máy chủ GPU): AWS EC2** với GPU chuyên dụng như **NVIDIA Tesla V100 hoặc A100** được chọn để huấn luyện mô hình do khả năng tăng tốc quá trình huấn luyện các mô hình học sâu phức tạp. Việc sử dụng GPU giúp giảm thiểu thời gian huấn luyện, đồng thời cho phép thử nghiệm với nhiều cấu hình mô hình và bộ siêu tham số khác nhau.

## **2.5. Những thách thức về thuật toán**

### **2.5.1. Tiền xử lý dữ liệu**

Thuật toán chủ yếu trong tiền xử lý dữ liệu đã có sẵn là “Global Protein Sequence Descriptor” và thuật toán “Tách chuỗi peptide” dài hơn 50 amino acid thành các đoạn 10 – 50 amino acid.

#### **2.5.1.1. Thuật toán tách chuỗi peptide**

##### **2.5.1.1.1. Mô tả**

Các chuỗi peptide từ vi sinh vật có độ dài lớn hơn 50 amino acid sẽ được chia thành các đoạn nhỏ có độ dài từ 10 đến 50 amino acid. Chỉ những đoạn thỏa mãn các điều kiện sinh hóa nhất định mới được giữ lại để sử dụng trong mô hình học sâu.

Các amino acid được chia thành 4 nhóm chính dựa trên đặc tính của nhóm R:

**a. Amino acid phân cực (Polar)**

* **Phân cực tích điện (Acidic/Basic):**
  + Acidic (âm tính): Asp (D), Glu (E)
  + Basic (dương tính): Lys (K), Arg (R), His (H)
* **Phân cực không tích điện (Hydrophilic):**
  + Ser (S), Thr (T), Asn (N), Gln (Q), Tyr (Y), Cys (C)

**b. Amino acid không phân cực (Non-polar/Hydrophobic)**

* Gly (G), Ala (A), Val (V), Leu (L), Ile (I), Met (M), Pro (P), Phe (F), Trp (W)

##### **2.5.1.1.2. Khó khăn**

* Việc tính toán điều kiện trong chuỗi peptide có thể phức tạp khi đối diện với dữ liệu chuỗi dài và nhiều biến thể của amino acid.
* Cần thiết phải tối ưu hóa việc tách chuỗi peptide và tính toán các chỉ số một cách hiệu quả để đảm bảo tính chính xác và tiết kiệm thời gian.

##### **2.5.1.1.3. Giải pháp thay thế**

* Áp dụng các thuật toán phân tích chuỗi như Dynamic Programming để tách chuỗi thành các đoạn với độ dài phù hợp.
* Sử dụng Parallel Processing để tính toán đặc tính hóa học của từng peptide nhanh chóng.

#### **2.5.1.2. Thuật toán tính toán Distribution Descriptor**

##### **2.5.1.2.1. Mô tả**

Tính toán đặc trưng phân bố của các nhóm amino acid trong chuỗi peptide, đặc biệt là nhóm phân loại theo tính chất lý hóa như độ kỵ nước (hydrophobicity). Mỗi amino acid trong chuỗi được phân loại vào ba nhóm: Polar, Neutral, và Hydrophobic.

* **Polar**: R, K, E, D, Q, N
* **Neutral**: G, A, S, T, P, H, Y
* **Hydrophobic**: C, L, V, I, M, F, W

##### **2.5.1.2.2. Khó khăn**

* Phân bố các amino acid có thể không đều trong chuỗi, gây khó khăn khi tính toán và phân tích.
* Việc xử lý các chuỗi dài với hàng nghìn amino acid đòi hỏi thuật toán phải tối ưu để tránh tốn thời gian xử lý.

##### **2.5.1.2.3. Giải pháp thay thế**

* Áp dụng phương pháp Sliding Window để tính toán phân bố của amino acid trong các đoạn nhỏ, giảm độ phức tạp tính toán.
* Sử dụng Approximation Techniques để ước tính các phân bố phân tử lớn, giúp tăng tốc độ tính toán mà vẫn đảm bảo độ chính xác.

#### **2.5.1.3. Khó khăn chung**

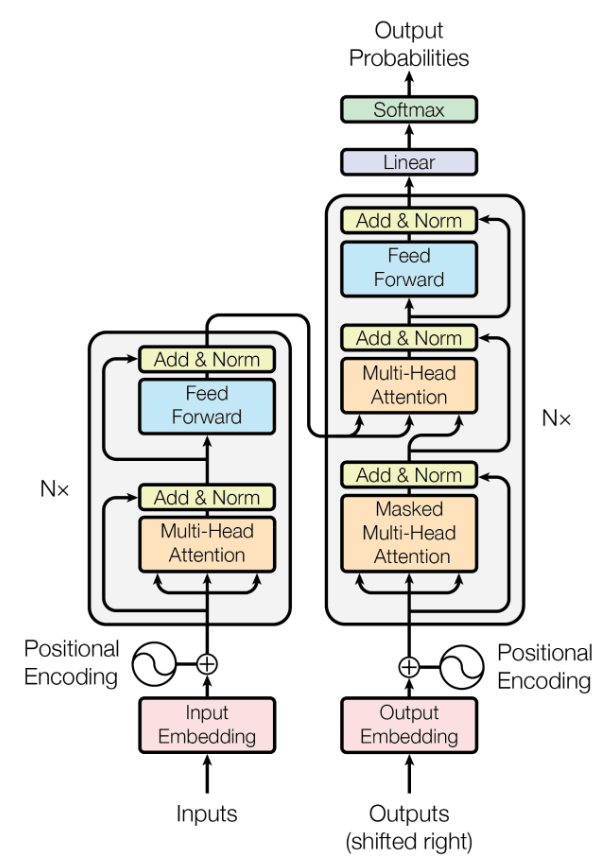
1. **Hiểu biết về đặc tính hóa học và sinh học của amino acid**:
   * Để phân loại các peptide và tính toán các đặc tính hóa học (như điện tích ròng và tỷ lệ amino acid không phân cực), cần nắm vững kiến thức về đặc tính của từng loại amino acid. Mỗi amino acid có những tính chất hóa học khác nhau, ví dụ: tính phân cực, tính kỵ nước (hydrophobicity), khả năng phân cực (polarizability), độ kỵ nước, v.v.
   * Khó khăn: Việc phân nhóm các amino acid và tính toán các chỉ số liên quan yêu cầu kiến thức vững về sinh học phân tử để đảm bảo phân loại đúng và tính toán chính xác.
2. **Tính toán đặc tính của peptide**:
   * Tính toán các đặc tính như net charge (điện tích ròng) và tỷ lệ amino acid không phân cực cần phải phân tích các yếu tố ảnh hưởng đến cấu trúc của peptide. Điều này liên quan đến các đặc tính hóa học của amino acid và cách chúng tương tác trong chuỗi peptide.
   * Khó khăn: Việc áp dụng lý thuyết hóa học để tính toán và phân tích các đặc tính này trong ngữ cảnh của peptide đòi hỏi kiến thức về sinh học phân tử và hóa sinh.
3. **Phân tích chuỗi protein và peptide**:
   * Việc tách các chuỗi peptide và phân loại chúng thành các đoạn có độ dài từ 10-50 amino acid cần có kiến thức về cấu trúc peptide, đặc biệt là các chuỗi dài.
   * Khó khăn: Cần hiểu rõ cách các chuỗi peptide tương tác với môi trường xung quanh (như dung môi, nhiệt độ) và cách thức các đặc tính sinh học ảnh hưởng đến cấu trúc và chức năng của chúng.
4. **Xử lý dữ liệu sinh học lớn**:
   * Dữ liệu chuỗi peptide và protein có thể rất lớn và phức tạp. Việc xử lý các chuỗi dài và tính toán các đặc tính sinh học đòi hỏi phải tối ưu hóa thuật toán để có thể phân tích nhanh chóng và chính xác.
   * Khó khăn: Xử lý dữ liệu sinh học khổng lồ đòi hỏi phải có kiến thức về cách thức sinh học phân tử hoạt động cũng như các kỹ thuật tính toán phức tạp để giảm tải cho mô hình.
5. **Phương pháp máy học trong sinh học**:
   * Các mô hình máy học như học sâu (deep learning) và học máy (machine learning) thường được sử dụng để dự đoán tính chất của peptide (như khả năng kháng khuẩn). Tuy nhiên, việc áp dụng các phương pháp này đòi hỏi phải hiểu rõ cách các đặc tính hóa học của các chuỗi protein ảnh hưởng đến khả năng dự đoán.
   * Khó khăn: Cần hiểu rõ cách thức các đặc điểm sinh học được mã hóa và sử dụng trong các mô hình học máy, cũng như cách các mô hình này có thể áp dụng vào các nghiên cứu sinh học.

#### **2.5.1.4. Giải pháp chung**

* Tăng cường sự hiểu biết về sinh học phân tử, hóa sinh học, và tính toán sinh học để có thể xây dựng thuật toán chính xác.
* Hợp tác với các chuyên gia sinh học để đảm bảo rằng các thuật toán và mô hình tính toán được xây dựng đúng theo nguyên lý sinh học.
* Áp dụng các phương pháp tối ưu như Parallel Processing và Approximation Techniques để xử lý dữ liệu sinh học phức tạp.

### **2.5.2. Thuật toán mô hình Transformer**

#### **2.5.2.1. Mô tả**

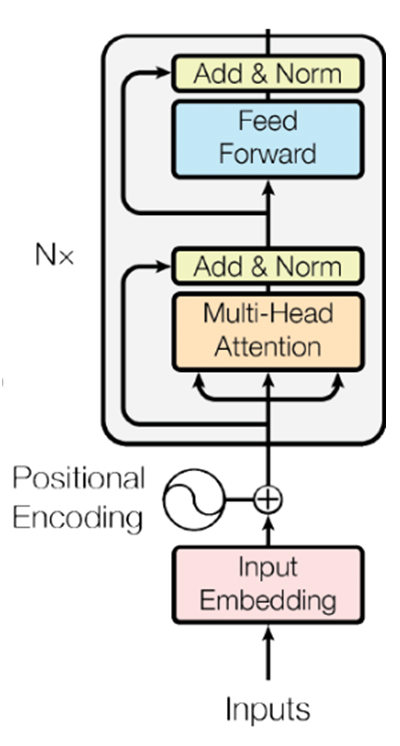


Sơ đồ 2. 4. Sơ đồ kiến trúc mô hình Transformer

Mô hình Transformer là một kiến trúc học sâu, được thiết kế để xử lý các chuỗi dữ liệu như văn bản, âm thanh, hay chuỗi peptide mà không sử dụng cấu trúc tuần tự như trong các mô hình Mạng Nơ-ron Hồi tiếp RNN (Recurrent Neural Network) hay Mạng LSTM (Long Short-Term Memory). Mô hình Transformer được giới thiệu lần đầu bởi Vaswani và các cộng sự vào năm 2017 trong bài báo “Attention is All You Need” và đã trở thành tiêu chuẩn trong các tác vụ như dịch máy, phân loại văn bản, và nhiều bài toán chuỗi khác [42].

Transformer bao gồm hai thành phần chính: Encoder và Decoder. Encoder có nhiệm vụ nhận vào chuỗi đầu vào và mã hóa nó thành các vector đặc trưng (embeddings). Mỗi Encoder bao gồm nhiều lớp, với các thành phần chính là Cơ chế Tự Chú Ý (Self-Attention Mechanism) và Mạng Nơ-ron Hồi tiếp (Feed-Forward Neural Networks). Cơ chế Tự Chú Ý giúp mô hình học được mối quan hệ giữa các phần tử trong chuỗi đầu vào mà không bị giới hạn bởi thứ tự tuyến tính. Trong khi đó, Mạng Nơ-ron Hồi tiếp giúp tăng cường khả năng học của mô hình. Decoder nhận các vector đặc trưng từ Encoder và tạo ra chuỗi đầu ra. Các thành phần chính của Decoder bao gồm Cơ chế Tự Chú Ý, Cơ chế Chú Ý giữa Encoder và Decoder (Encoder-Decoder Attention), và Mạng Nơ-ron Hồi tiếp. Cơ chế Chú Ý cho phép Decoder chú ý đến các vector đặc trưng từ Encoder, kết hợp với các thông tin từ các bước trước đó trong quá trình tạo chuỗi đầu ra. Đề tài tài sẽ được thực hiện 100% chỉ trên encoder.

Sau đây là cấu trúc của Transformer Encoder:



Sơ đồ 2. 5. Sơ đồ kiến trúc mô hình Transformer Encoder

Transformer Encoder [32] bao gồm một loạt các lớp, mỗi lớp này sẽ bao gồm:

* Cơ chế tự chú ý (Self-Attention): Đây là cơ chế chủ đạo của Transformer, cho phép mô hình xác định tầm quan trọng của từng phần tử trong chuỗi so với các phần tử khác. Điều này có thể giúp mô hình học được các mối quan hệ toàn cục giữa các yếu tố trong chuỗi, bất kể khoảng cách giữa chúng.
* Tầng chuỗi đa lớp (Feed-forward network): Sau khi xử lý qua cơ chế tự chú ý, dữ liệu sẽ được chuyển qua một mạng nơ-ron đa lớp để tạo ra các đặc trưng cao cấp hơn.
* Kết hợp thêm vị trí (Positional Encoding): Vì cơ chế tự chú ý không giữ thứ tự chuỗi, Transformer sử dụng vị trí mã hóa để thêm thông tin thứ tự vào các vector đầu vào, giúp mô hình hiểu được các yếu tố trong chuỗi có vị trí nhất định.
* Chuẩn hóa và kết hợp (Normalization & Residual Connections): Để giảm thiểu vấn đề suy giảm gradient và cải thiện khả năng huấn luyện, các lớp này còn sử dụng kỹ thuật chuẩn hóa và kết nối dư thừa (residual connections).

#### **2.5.2.2. Khó khăn**

* Cần một lượng lớn dữ liệu huấn luyện: Mô hình Transformer Encoder yêu cầu một lượng dữ liệu huấn luyện cực kỳ lớn để học được các đặc điểm phức tạp của chuỗi peptide, đặc biệt là trong bài toán phân loại peptide kháng khuẩn và không kháng khuẩn. Dữ liệu phải đủ đa dạng và phong phú để mô hình có thể tổng quát hóa tốt. Thiếu dữ liệu sẽ làm giảm độ chính xác và khả năng khái quát của mô hình.
* Cần tài nguyên tính toán mạnh mẽ (GPU): Mô hình Transformer, đặc biệt là khi sử dụng encoder, có yêu cầu tính toán rất cao. Việc huấn luyện mô hình trên một bộ dữ liệu lớn yêu cầu sử dụng GPU hoặc TPU để giảm thiểu thời gian huấn luyện và cải thiện hiệu suất tính toán. Không có GPU hoặc phần cứng tính toán mạnh mẽ sẽ làm cho việc huấn luyện trở nên tốn thời gian và kém hiệu quả, đặc biệt đối với các mô hình phức tạp như Transformer.

#### **2.5.2.3. Giải pháp thay thế**

* Tăng cường dữ liệu (Data Augmentation): Một giải pháp để khắc phục vấn đề thiếu dữ liệu là tạo dữ liệu giả thông qua các kỹ thuật tăng cường, chẳng hạn như thay đổi các amino acid trong chuỗi peptide hoặc tạo các chuỗi peptide giả từ các mô hình sinh học [36]. Điều này giúp mở rộng bộ dữ liệu mà không cần phải thu thập dữ liệu mới.
* Sử dụng dịch vụ đám mây với GPU: Để giảm bớt gánh nặng về phần cứng, có thể tận dụng các dịch vụ đám mây như Google Cloud, Amazon Web Services (AWS), hoặc Microsoft Azure, những nền tảng này cung cấp các máy chủ có GPU mạnh mẽ cho việc huấn luyện mô hình học sâu [37]. Điều này giúp bạn có đủ tài nguyên tính toán mà không cần phải đầu tư vào phần cứng đắt đỏ.
* Sử dụng học chuyển giao (Transfer Learning): Thay vì huấn luyện mô hình từ đầu, bạn có thể sử dụng các mô hình đã được huấn luyện sẵn trên các bộ dữ liệu lớn [31]. Sau đó, tinh chỉnh mô hình với bộ dữ liệu chuỗi peptide của bạn, giúp giảm bớt nhu cầu về dữ liệu và tài nguyên tính toán.

### **2.5.3. Thuật toán LIME**

#### **2.5.3.1. Mô tả thuật toán LIME**

LIME là một phương pháp giải thích mô hình học máy độc lập với mô hình gốc (model-agnostic), được sử dụng để giải thích các quyết định của bất kỳ mô hình học máy nào, bao gồm các mô hình học sâu như Transformer.

LIME hoạt động dựa trên nguyên lý tạo ra các mẫu dữ liệu giả (perturbed data) gần với mẫu dữ liệu gốc mà mô hình cần giải thích, rồi huấn luyện một mô hình đơn giản, dễ hiểu (như hồi quy tuyến tính) trên tập dữ liệu này. Mục tiêu của LIME là cung cấp các giải thích cục bộ, tức là giải thích cho từng dự đoán cụ thể của mô hình học máy.

Quy trình của LIME bao gồm các bước chính:

1. **Tạo dữ liệu giả:** LIME tạo ra các mẫu dữ liệu giả bằng cách thay đổi nhẹ các đặc tính của mẫu dữ liệu ban đầu (ví dụ, thay đổi một số amino acid trong chuỗi peptide).
2. **Huấn luyện mô hình đơn giản:** Sau khi tạo dữ liệu giả, một mô hình đơn giản (thường là hồi quy tuyến tính hoặc cây quyết định) được huấn luyện trên các dữ liệu giả này.
3. **Giải thích dự đoán của mô hình chính:** Mô hình đơn giản này sẽ được sử dụng để giải thích dự đoán của mô hình học sâu cho mẫu dữ liệu ban đầu..

#### **2.5.3.2. Khó khăn**

* Chi phí tính toán cao: Một trong những khó khăn lớn khi áp dụng LIME là yêu cầu tính toán cao. Để tạo ra các mẫu dữ liệu giả và huấn luyện mô hình đơn giản trên chúng, cần phải tạo nhiều phiên bản dữ liệu và tính toán các dự đoán cho mỗi phiên bản này. Điều này có thể tốn rất nhiều thời gian và tài nguyên tính toán, đặc biệt khi làm việc với các mô hình học sâu phức tạp.
* Khó khăn trong việc chọn mô hình đơn giản: Việc chọn mô hình đơn giản phù hợp với dữ liệu có thể là một thách thức. Mô hình quá phức tạp hoặc không phù hợp có thể không cung cấp được giải thích tốt về các quyết định của mô hình học sâu. Việc lựa chọn mô hình đơn giản và có khả năng giải thích tốt rất quan trọng, nhưng cũng đòi hỏi kiến thức và thử nghiệm.
* Khó khăn trong việc giải thích các tính năng phức tạp: Khi làm việc với các chuỗi peptide, đặc biệt là những chuỗi dài và phức tạp, các tính năng (như các amino acid) có thể có mối quan hệ phức tạp với nhau. Việc giải thích tác động của mỗi tính năng trên dự đoán của mô hình có thể trở nên khó khăn khi dữ liệu có độ phức tạp cao.
* Tính đại diện của các mẫu dữ liệu giả: Các mẫu dữ liệu giả tạo ra trong quá trình giải thích có thể không đại diện đầy đủ cho toàn bộ không gian dữ liệu. Điều này có thể làm giảm tính chính xác và độ tin cậy của giải thích. Nếu các mẫu dữ liệu giả không đủ đa dạng và đại diện, giải thích đưa ra có thể không phản ánh chính xác sự hoạt động của mô hình.

#### **2.5.3.3. Giải pháp thay thế**

* Tối ưu hóa chi phí tính toán: Để giảm thiểu chi phí tính toán khi sử dụng LIME, có thể giới hạn số lượng mẫu dữ liệu giả cần tạo ra bằng cách sử dụng phương pháp chọn mẫu thông minh hoặc giảm bớt số lượng các tính năng cần giải thích. Thêm vào đó, các giải pháp tính toán phân tán hoặc sử dụng các GPU có thể giúp tăng tốc quá trình huấn luyện và tính toán mô hình đơn giản.
* Chọn mô hình đơn giản phù hợp: LIME có thể sử dụng các mô hình đơn giản như hồi quy tuyến tính hoặc cây quyết định để giải thích dự đoán. Tuy nhiên, việc lựa chọn mô hình phù hợp cho dữ liệu của bài toán là rất quan trọng. Một giải pháp là thử nghiệm với nhiều mô hình đơn giản và chọn mô hình mang lại giải thích dễ hiểu và chính xác nhất.
* Tăng cường tính đại diện của mẫu dữ liệu giả: Để đảm bảo rằng các mẫu dữ liệu giả đại diện cho toàn bộ không gian dữ liệu, có thể sử dụng các phương pháp chọn mẫu có trọng số hoặc phương pháp tăng cường dữ liệu để tạo ra các mẫu dữ liệu giả phong phú hơn. Điều này giúp giải thích đúng hơn về các yếu tố ảnh hưởng đến các quyết định của mô hình.
* Cải thiện sự minh bạch của các tính năng phức tạp: Để giải thích rõ ràng hơn về các tính năng phức tạp trong chuỗi peptide, có thể sử dụng công cụ phân tích tính năng tiên tiến để xác định các yếu tố quan trọng trong chuỗi. Các kỹ thuật như tạo ảnh nhiệt (heatmap) hoặc chỉ số tầm quan trọng của tính năng có thể hỗ trợ trong việc hiểu rõ hơn về mối quan hệ giữa các tính năng và kết quả dự đoán.

# **CHƯƠNG 3: CÀI ĐẶT THỬ NGHIỆM, KẾT QUẢ VÀ ĐÁNH GIÁ**

## **3.1. Các thiết kế mô hình giải pháp**

Dựa vào hình ảnh quy trình trong chi tiết quá trình thực hiện thì có sẽ cũng giải pháp nhất định như sau.

### **3.1.1. Thu thập và xử lý dữ liệu**

Phần thu thập và xử lý dữ liệu này bao gồm bước (1) cho đến bước (8).

(1) Để thu thập dữ liệu chuỗi peptide từ các loài vi sinh vật, bao gồm các peptide kháng khuẩn và không kháng khuẩn, có thể sử dụng API (Application Programming Interface) của NCBI (National Center for Biotechnology Information), đặc biệt là thông qua thư viện Biopython [2]. Quy trình thu thập này bao gồm các bước sau:

Trước tiên, bạn cần sử dụng các câu lệnh tìm kiếm trong NCBI (National Center for Biotechnology Information) để xác định dữ liệu cần thiết. Các câu lệnh tìm kiếm có thể được xây dựng dựa trên các từ khóa liên quan đến peptide và các loài vi sinh vật, kết hợp với các bộ cơ sở dữ liệu (DBS) khác nhau như GenBank, RefSeq, PDB, SwissProt, DDBJ, EMBL, và PIR. Bạn có thể chọn lọc các peptide có khả năng kháng khuẩn hoặc không kháng khuẩn bằng cách sử dụng các từ khóa cụ thể như “antimicrobial”, “bacteriocin”, “AMPs”, hoặc “antimicrobial peptide”, phủ định của “antimicrobial”,…. Các câu lệnh tìm kiếm này sẽ giúp lọc dữ liệu từ các loài vi sinh vật như nấm (fungi), vi khuẩn (bacteria), virus, và archaea.

Một khi các câu lệnh tìm kiếm được xác định, bạn có thể sử dụng Biopython để gửi yêu cầu API (Application Programming Interface) tới NCBI Entrez. Câu lệnh tìm kiếm sẽ được gửi tới NCBI (National Center for Biotechnology Information), nơi trả về dữ liệu dưới dạng các bản ghi liên quan đến các peptide theo yêu cầu. Dữ liệu trả về sẽ bao gồm các chuỗi peptide từ các loài vi sinh vật mà bạn quan tâm. Bằng cách thay đổi các tham số trong câu lệnh tìm kiếm (chẳng hạn như thay đổi cơ sở dữ liệu từ GenBank sang PDB, RefSeq, hoặc SwissProt), bạn có thể điều chỉnh nguồn dữ liệu theo nhu cầu của mình.

Các bộ lọc như “Fungi”, “Bacteria”, “Archaea”, và “Viruses” có thể được sử dụng để hạn chế kết quả chỉ trong các loài vi sinh vật mong muốn. Ngoài ra, bạn cũng có thể tích hợp các câu lệnh lọc thêm về các cơ sở dữ liệu cụ thể (như GenBank hoặc PDB) để thu thập các peptide từ các nguồn này. Khi tích hợp các câu lệnh tìm kiếm với Biopython, bạn sẽ thu thập được các chuỗi peptide từ các cơ sở dữ liệu khác nhau mà không cần phải truy cập thủ công vào các trang web của NCBI (National Center for Biotechnology Information). (2) Trong quá trình thu thập dữ liệu, việc kiểm tra và loại bỏ các chuỗi trùng lặp đóng vai trò hết sức quan trọng. Do cơ sở dữ liệu NCBI cho phép bất kỳ ai cũng có thể đăng tải dữ liệu, nên khả năng xuất hiện các chuỗi bị trùng là điều khó tránh khỏi. Nếu không xử lý, các chuỗi lặp sẽ khiến mô hình học máy bị thiên lệch và sinh nhiễu, ảnh hưởng đến chất lượng huấn luyện. Vì vậy, việc kiểm tra trùng lặp và loại bỏ các chuỗi không cần thiết là bước thiết yếu nhằm đảm bảo tính chính xác và độ tin cậy của dữ liệu đầu vào.

(3) Sau khi thu thập dữ liệu chuỗi peptide từ các loài vi sinh vật, bước tiếp theo là tiến hành tách các chuỗi peptide có độ dài hơn 50 amino acid thành các đoạn 10 đến 50 amino acid. (4) Quá trình này được thực hiện nhằm tách các chuỗi peptide dài hơn 50 amino acid thành các đoạn có 10 đến 50 amino acid theo các đặc điểm sinh hóa học cụ thể [38].

Sau đây là các thành phần của chuỗi peptide:

* **Amino acid phân cực (Polar)**
* **Phân cực tích điện (Acidic/Basic)**:
  + Acidic (âm tính -): Asp (D), Glu (E)
  + Basic (dương tính +): Lys (K), Arg (R), His (H)
* **Phân cực không tích điện (Hydrophilic)**:
  + Ser (S), Thr (T), Asn (N), Gln (Q), Tyr (Y), Cys (C)
* **Amino acid không phân cực (Non-polar/Hydrophobic)**
* Gly (G), Ala (A), Val (V), Leu (L), Ile (I), Met (M), Pro (P), Phe (F), Trp (W)

(5) Sau khi tách chuỗi thành các đoạn có độ dài từ 10 – 50 amino acid thì tiếp theo sẽ sinh ra các đặc trưng từ chuỗi peptide bằng kĩ thuật tính toán Global Protein Sequence Descriptor.

Global Protein Sequence Descriptor là một phương pháp được sử dụng để biểu diễn các trình tự protein dưới dạng số, dựa trên các đặc tính lý hóa của các acid amin. Phương pháp này được phát triển nhằm mục đích mã hóa trình tự protein theo các thuộc tính quan trọng, giúp các mô hình machine learning có thể học và dự đoán các tính chất của protein, như khả năng gấp cuộn (folding) hoặc khả năng kháng khuẩn [22].

**Các đặc điểm chính của Global Protein Sequence Descriptor:**

1. **Dựa trên các thuộc tính lý hóa (physicochemical properties):**
   * Phương pháp này mã hóa protein dựa trên các thuộc tính lý hóa của các amino acid như:
     + Độ kỵ nước (hydrophobicity)
     + Thể tích van der Waals chuẩn hóa (normalized van der Waals volume)
     + Độ phân cực (polarity)
     + Khả năng phân cực (polarizability)
     + Điện tích (charge)
     + Cấu trúc thứ cấp (secondary structure)
     + Khả năng tiếp xúc với dung môi (solvent accessibility)
   * Mỗi thuộc tính lý hóa này có thể ảnh hưởng đến cấu trúc và chức năng của protein, do đó, mã hóa chúng giúp phản ánh các đặc điểm sinh học của chuỗi protein.
2. **Phân nhóm các amino acid:**
   * Với mỗi thuộc tính lý hóa, các amino acid được chia thành ba nhóm (classes) dựa trên giá trị của thuộc tính đó.
   * Ví dụ, theo thuộc tính độ kỵ nước (hydrophobicity), các amino acid có thể được chia thành:
     + **Polar** (R, K, E, D, Q, N)
     + **Neutral** (G, A, S, T, P, H, Y)
     + **Hydrophobic** (C, L, V, I, M, F, W)
3. **Mã hóa phân bố vị trí (Distribution descriptors):**
   * Phương pháp này tính toán các chỉ số phân bố của các amino acid thuộc từng nhóm, dựa trên tỷ lệ vị trí của các phần tử đầu tiên, 25%, 50%, 75%, và 100% trong chuỗi protein.
   * Chỉ số phân bố này giúp xác định mẫu phân bố của mỗi nhóm amino acid trong chuỗi. Ví dụ, nếu các amino acid kỵ nước phân bố ở đầu và cuối chuỗi, mô hình học sâu có thể nhận diện và sử dụng thông tin này trong dự đoán.

* **Ứng dụng của Global Protein Sequence Descripto**r
* Dự đoán cấu trúc và chức năng protein: Phương pháp này giúp mã hóa thông tin về trình tự protein, hỗ trợ các mô hình machine learning dự đoán các tính chất phức tạp của protein.
* Xác định peptide kháng khuẩn (AMPs**):** Bằng cách mã hóa các đặc tính của AMPs, ta có thể sử dụng machine learning để phân loại và xác định các peptide có khả năng kháng khuẩn.
* Phân loại protein**:** Global Protein Sequence Descriptor có thể giúp phân loại các protein thành các nhóm dựa trên mẫu phân bố đặc trưng của các thuộc tính lý hóa, giúp hiểu rõ hơn về cơ chế sinh học của chúng.
* **Khái niệm và phương pháp tính toán Distribution Description**

Distribution Descriptor là một loại chỉ số mô tả trong Global Protein Sequence Descriptor dùng để thể hiện phân bố của các nhóm acid amin theo các thuộc tính lý hóa trong một chuỗi protein hoặc peptide. Mỗi nhóm amino acid được xác định dựa trên các tính chất lý hóa cụ thể và có thể chia thành các lớp như Polar, Neutral, và Hydrophobic theo độ kỵ nước. Distribution descriptor được tính toán dựa trên vị trí của các phần tử thuộc các lớp này trong chuỗi protein.

* **Phương pháp tính toán Distribution Descriptor**

1. Lớp tính chất lý hóa: Đầu tiên, các amino acid trong chuỗi protein được phân loại thành các nhóm theo một tính chất cụ thể. Ví dụ, tính chất độ kỵ nước (hydrophobicity) có thể chia amino acid thành các nhóm sau:
   * **Polar (P)**: R, K, E, D, Q, N
   * **Neutral (N)**: G, A, S, T, P, H, Y
   * **Hydrophobic (H)**: C, L, V, I, M, F, W
2. Xác định phân bố của các phần tử**:** Sau đó, chúng ta sẽ tính toán các giá trị phân bố của các phần tử thuộc từng lớp dựa trên các điểm phần trăm:
   * First residue: Vị trí xuất hiện đầu tiên của một amino acid thuộc lớp đó.
   * 25% residue: Vị trí của phần tử thứ 25% trong chuỗi của nhóm amino acid đó.
   * 50% residue: Vị trí của phần tử thứ 50% trong chuỗi của nhóm amino acid đó.
   * 75% residue: Vị trí của phần tử thứ 75%.
   * 100% residue: Vị trí của phần tử cuối cùng của nhóm trong chuỗi/

Dựa trên 2 cách tính trên thì có thể tính được chỉ số đặc trưng của từng chuỗi peptide mà vẫn giữ được tính chắc lý hóa.

Trải qua 5 bước đầu thì thu được tập dữ liệu bao gồm các chuỗi peptide có chiều dài từ 10 đến 50 amino acid. Mỗi chuỗi được đi kèm với các đặc trưng sinh học đã được số hóa bằng phương pháp Global Protein Sequence Descriptor ở bước 5, một phương pháp mô tả toàn cục giúp trích xuất các đặc tính vật lý, hóa học và cấu trúc tiềm ẩn từ chuỗi trình tự. (6) Tiền xử lý dữ liệu là bước quan trọng nhằm đảm bảo chất lượng đầu vào cho mô hình học sâu. Trong quá trình này:

* (6.1) **Loại bỏ chuỗi trùng lặp** để tránh tình trạng mô hình bị overfitting do học lại dữ liệu giống nhau.
* (6.2) **Mã hóa (Tokenizer)** các chuỗi peptide thành mảng số nguyên, đại diện cho các amino acid, giúp mô hình học được cấu trúc trình tự.
* (6.3) **Chuẩn hóa độ dài (padding)** bằng cách thêm số 0 đệm ở cuối chuỗi để tất cả các chuỗi có cùng độ dài, đảm bảo tính tương thích với kiến trúc mạng nơ-ron tuần tự hoặc mạng Transformer.

Tạo đầu vào cho mô hình học sâu trong bài toán xử lý chuỗi peptide là một bước thiết yếu, nhằm đảm bảo rằng mô hình có thể khai thác được cả thông tin trình tự lẫn đặc trưng sinh hóa của dữ liệu. Quá trình này bao gồm hai luồng dữ liệu độc lập nhưng bổ trợ lẫn nhau:

* (7.1) Đầu vào thứ nhất là một vector số nguyên đại diện cho trình tự amino acid. Mỗi chuỗi peptide ban đầu được đưa qua bước tokenizer, chuyển đổi từng ký tự amino acid thành chỉ số số nguyên tương ứng. Sau đó, các chuỗi này được padding để đảm bảo độ dài đồng nhất, phù hợp với yêu cầu của mô hình học sâu. Đây là nguồn dữ liệu chính để mô hình như Transformer học được thông tin liên quan đến thứ tự và mối quan hệ giữa các vị trí trong chuỗi peptide.
* (7.2) Đầu vào thứ hai là một vector đặc trưng số hóa được xây dựng từ các phương pháp mô tả toàn cục (Global Protein Sequence Descriptor). Vector này không phụ thuộc vào trình tự cụ thể của chuỗi mà thay vào đó phản ánh các thuộc tính tổng quát về mặt sinh hóa, chẳng hạn như độ phân cực, tính kỵ nước, khối lượng phân tử,... Nhờ đó, mô hình học sâu không chỉ dựa vào chuỗi token hóa mà còn có thêm kiến thức nền tảng về tính chất sinh học của peptide, giúp cải thiện độ chính xác và khả năng khái quát hóa.
* (7.3) Một trong những nhược điểm nổi bật của cơ chế self-attention trong Transformer là mô hình không tự động biết được vị trí tuyệt đối của các token trong chuỗi. Điều này có thể dẫn đến tình trạng nhầm lẫn hoặc giảm độ chính xác trong việc hiểu cấu trúc trình tự. Để khắc phục điểm yếu này, một lớp đặc biệt có tên TokenAndPositionEmbedding được thiết kế. Lớp này thực hiện hai chức năng quan trọng: thứ nhất là ánh xạ mỗi token (tức là chỉ số số nguyên đại diện cho một amino acid) sang một vector nhiều chiều thông qua lớp Embedding; thứ hai là thêm thông tin vị trí bằng cách cộng embedding vị trí tương ứng. Việc kết hợp thông tin nội dung và vị trí này là vô cùng quan trọng, vì nó giúp mô hình Transformer hiểu được “ai đang ở đâu” trong chuỗi peptide điều kiện tiên quyết để mô hình học được các tương tác phức tạp giữa các vị trí amino acid.

(8) Trong quá trình xây dựng mô hình học sâu với TensorFlow, một yêu cầu kỹ thuật quan trọng là mô hình chỉ có thể nhận một đầu vào duy nhất hoặc một cấu trúc đầu vào được xác định rõ ràng. Tuy nhiên, trong trường hợp đặc biệt như xử lý chuỗi peptide, việc khai thác hiệu quả thông tin từ hai nguồn khác nhau là điều cần thiết để mô hình học sâu có thể đạt hiệu suất cao. Do đó, giải pháp được sử dụng là tạo hai đầu vào độc lập, mỗi đầu vào tương ứng với một loại thông tin đặc thù, sau đó gộp lại thành một thể thống nhất trước khi đưa vào phần chính của mô hình.

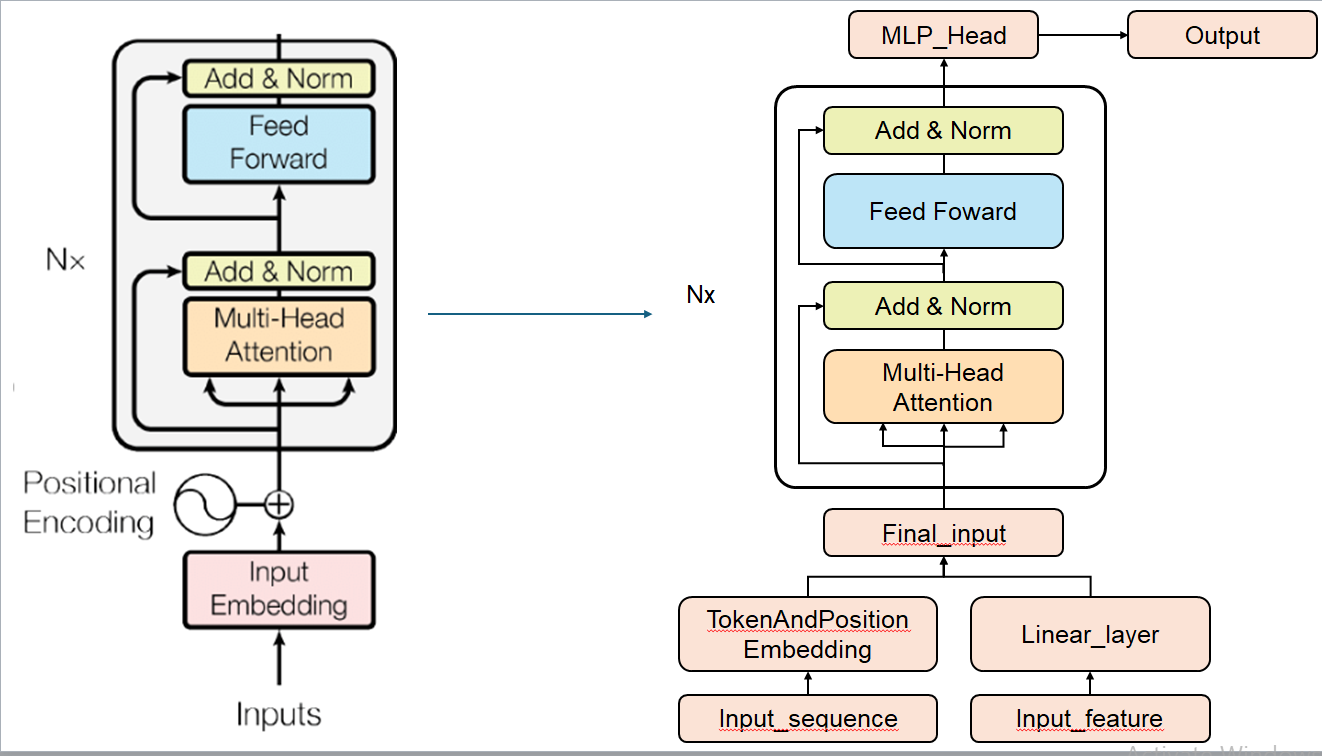
Cụ thể, đầu vào thứ nhất là một vector số nguyên biểu diễn trình tự các amino acid. Các chuỗi peptide ban đầu được chuyển thành vector số nguyên thông qua bước mã hóa bằng tokenizer, trong đó mỗi amino acid được ánh xạ thành một chỉ số duy nhất. Những vector số này sau đó được xử lý bằng phương pháp padding để đảm bảo độ dài đồng nhất, phù hợp với yêu cầu của mô hình học sâu như Transformer. Đây là nguồn dữ liệu trọng yếu để mô hình học được thông tin tuần tự và mối quan hệ giữa các vị trí trong chuỗi.

Đầu vào thứ hai là một vector đặc trưng số học, được trích xuất bằng phương pháp mô tả sinh học toàn cục (Global Protein Sequence Descriptor). Vector này không mang tính chất trình tự mà phản ánh các đặc tính sinh hóa tổng quát của toàn bộ chuỗi peptide, như độ phân cực, tính kỵ nước, hoặc khối lượng phân tử,... Việc đưa thêm nguồn thông tin này giúp mô hình không chỉ tập trung vào cấu trúc thứ tự mà còn có được kiến thức nền tảng về tính chất hóa học và sinh học, từ đó tăng khả năng khái quát hóa và độ chính xác trong dự đoán.

Để hai đầu vào này có thể cùng tồn tại trong một mô hình, ta thiết kế hai đầu vào (input) riêng biệt trong TensorFlow, sau đó xử lý từng luồng dữ liệu theo đặc trưng riêng và cuối cùng gộp chúng lại. Đây là một kỹ thuật phổ biến trong các mô hình học sâu đa luồng, đảm bảo rằng mô hình có thể tiếp nhận và khai thác tối đa từ nhiều nguồn thông tin đầu vào, đáp ứng được tính phức tạp của bài toán phân tích peptide.

### **3.1.2. Xây dựng mô hình học sâu Transformer**

Phần xây dựng mô hình Transformer này bao gồm các bước (9), (9.1) và sau đó sẽ đi tìm kiếm siêu tham số cho mô hình Transformer ở bước (9.2). Hình bên trái là mô hình Transformer Encoder được đăng chính thức từ google, còn bên phải là một mô hình chỉnh sửa để phù hợp với cấu hình cũng như làm việc trên chuỗi peptide loài vi sinh vật. Sau nhiều năm ra mắt viêc xây dựng mô hình Transformer sẽ dễ dàng hơn vì đã có nhiều nhà nghiên cứu đi trước [1].



Sơ đồ 3. 1. Sơ đồ kiến trúc mô hình Transformer tùy chỉnh

Đầu vào của mô hình Transformer sử dụng encoder chính là đầu vào sau khi gộp lại ở trên bước thu thập và xử lý dữ liệu. (9) Mô hình Transformer và đầu ra của mô hình được cấu tạo nên những thành phần như sau:

* Multi-Head Attention
* Add & Norm (Residual Connection and Layer Normalization)
* Feed Forward
* MLP Head (Multi-Layer Perceptron Head)

#### **3.1.2.1. Multi-Head Attention**

Multi-Head Attention là một trong những thành phần cốt lõi của kiến trúc Transformer. Nó cho phép mô hình học được thông tin từ nhiều khía cạnh khác nhau của chuỗi đầu vào bằng cách sử dụng nhiều cơ chế attention song song, mỗi cái được gọi là một “head”. Ta có các công thức sau:

**Attention (Q, K, V) = softmax( ) x V**

##### **3.1.2.1.1. Self-Attention**

Self-Attention là một phương pháp cho phép mô hình học được mối quan hệ giữa các phần tử khác nhau trong một chuỗi dữ liệu, bất kể khoảng cách giữa chúng [35]. Đây là nền tảng của kiến trúc Transformer, giúp mô hình hiểu được ngữ cảnh toàn cục thay vì chỉ dựa vào ngữ cảnh cục bộ như RNN (Recurrent Neural Networks).

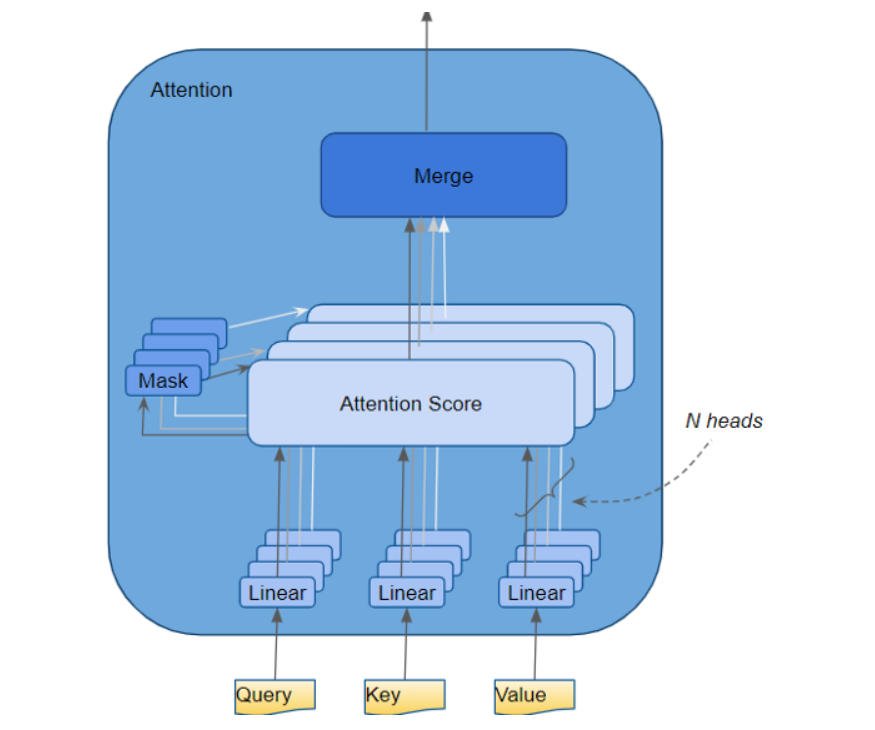
Cơ chế Attention có thể được hiểu như là sự kết hợp giữa một Query và một cặp Key-Value để cho ra một Output. Công thức tính như sau:

**Attention (Q, K, V) = softmax( ) x V**

Trong đó,  là hệ số tỷ lệ. là số chiều của Key.

##### **3.1.2.1.2. Multi-Head Self-Attention**

Trong kiến trúc của Transformer, mỗi một Self-Attention được gọi là một Head. Việc sử dụng nhiều heads đồng thời gọi là Multi-Head.



Sơ đồ 3. 2. Sơ đồ cơ chế Multi-Head Attention

Mỗi Head nhận vào một Input x (Token Embedding và Positional Decoding) và cho ra:

Trong đó, = (*trong trường hợp một Head*) /*(số lượng Head)*.

Sau khi có được các output của từng Head, ta sẽ tổng hợp chúng lại thành 1 Output duy nhất.

 là ma trận có chiều rộng bằng với chiều rộng của ma trận Input, mục đích sử dụng của nó là để đưa kích thước của Output về bằng với kích thước của Input.

#### **3.1.2.2. Add & Norm**

Trong kiến trúc Transformer, thành phần Add & Norm là một bước không thể thiếu, được sử dụng sau mỗi lớp con (sub-layer) như Self-Attention hoặc Feed Forward Mục tiêu của bước này là duy trì tính ổn định của mô hình trong quá trình huấn luyện, đồng thời giúp việc lan truyền thông tin và gradient hiệu quả hơn.

##### **3.1.2.2.1. Residual Connection**

Residual Connection là kỹ thuật giúp mô hình giữ lại thông tin đầu vào ban đầu bằng cách cộng đầu vào x với đầu ra của lớp con. Kỹ thuật này giúp giảm thiểu hiện tượng mất mát thông tin (information degradation) khi qua nhiều lớp sâu.

**Residual Output= x + Sublayer(x)**

Trong đó:

* x là đầu vào của lớp con,
* Sub Layer là đầu ra sau khi xử lý bởi lớp con (ví dụ: Attention hoặc Feed Forward).

##### **3.1.2.2.2. Layer Normalization**

Sau khi cộng kết quả từ Residual Connection, toàn bộ vector đầu ra sẽ được đưa qua Layer Normalization để đảm bảo các đặc trưng có phân phối ổn định. Layer Normalization được áp dụng theo từng vector đầu vào, không phụ thuộc vào batch size như Batch Normalization.

**Output = LayerNorm (x + Sublayer(x))**

##### **3.1.2.2.3. Vai trò của Add & Norm**

* Duy trì thông tin gốc: Residual giúp giữ lại những thông tin quan trọng từ đầu vào.
* Giúp gradient lan truyền tốt hơn: Tránh hiện tượng gradient biến mất khi mô hình có nhiều lớp.
* Ổn định quá trình học: LayerNorm chuẩn hóa phân phối đầu ra, giúp mô hình hội tụ nhanh và hiệu quả hơn.

#### **3.1.2.3. Feed Forward**

Feed Forward là một mạng nơ-ron gồm hai lớp tuyến tính (Linear) với một hàm kích hoạt phi tuyến thường là ReLU (Rectified Linear Unit) và sẽ nằm giữa. Điều quan trọng là FFN (Feed Forward Neural Network) áp dụng riêng biệt và giống nhau cho từng vị trí (token) trong chuỗi, tức là không có tương tác giữa các từ tại bước này, mà chỉ biến đổi thông tin tại từng từ một cách sâu sắc hơn.

**Trong đó:**

* x là đầu vào (vector biểu diễn một token),
* là các ma trận trọng số của hai lớp tuyến tính,
* là các vector hệ số điều chỉnh (bias),
* ReLU là hàm kích hoạt phi tuyến: ReLU(z) = max (0, z)

1. Thông tin đầu vào x được biến đổi qua một lớp tuyến tính đầu tiên (), như một phép suy luận ban đầu.
2. Áp dụng ReLU, giúp loại bỏ những giá trị âm – chỉ giữ lại tín hiệu mạnh mẽ (dương tính).
3. Kết quả được đưa qua lớp tuyến tính thứ hai () để tạo ra biểu diễn mới, sâu sắc hơn, sẵn sàng cho bước tiếp theo.

#### **3.1.2.4. MLP Head**

MLP Head (Multi-Layer Perceptron Head) là phần mạng nơ-ron nhiều lớp nằm ở đầu ra của một mô hình học sâu như Transformer, ViT, hoặc BERT. Nó đóng vai trò như một khâu ra quyết định cuối cùng, nơi thông tin trừu tượng được mô hình rút ra trước đó sẽ được biến đổi thành kết quả cụ thể chẳng hạn như nhãn phân loại, điểm số, hoặc xác suất. Sau khi đi qua lớp MLP Head (Multi-Layer Perceptron Head) là thì sẽ đi qua 1 lớp tuyến tính đầu ra là 1 để có được output

Một MLP Head thường bao gồm:

* Một hoặc nhiều lớp tuyến tính (Linear layers) – còn gọi là fully-connected layers.
* Hàm kích hoạt phi tuyến (ReLU, GELU, v.v.) để tăng khả năng biểu đạt phi tuyến tính.
* Lớp Softmax (nếu dùng cho phân loại) để chuyển đổi đầu ra thành phân phối xác suất.

Mỗi lớp trong MLP thực hiện phép biến đổi tuyến tính trên đầu vào và áp dụng hàm phi tuyến để trích xuất đặc trưng sâu hơn.

Trong đó:

* x: vector đầu vào (thường là đầu ra của encoder/Transformer),
* ​: ma trận trọng số của các lớp tuyến tính,
* : các vector bias,
* ϕ: hàm kích hoạt phi tuyến (ví dụ: ReLU, GELU).

### **Tích hợp XAI vào mô hình Transformer**

(10) Mô hình Transformer sẽ được tích hợp Explainable AI với kĩ thuật LIME (Local Interpretable Model-agnostic Explanations) tìm một mô hình tuyến tính g (thuộc tập các mô hình dễ hiểu G) sao cho g xấp xỉ tốt nhất mô hình f quanh điểm x [8]. Bài toán tối ưu được định nghĩa như sau:

Chi tiết hơn

**Với và =**

Trong đó:

* : Hàm mất mát đo lường sự khác biệt giữa mô hình phức tạp f và mô hình giải thích g trên các mẫu z được tạo ra từ phân phối .
* :Phân phối xác suất của các điểm dữ liệu xung quanh điểm cần giải thích x. Thông thường, trong LIME, phân phối này là phân phối Gaussian, tạo ra các điểm dữ liệu giả (samples) gần điểm. Gaussian kernel: với D (x, z) là khoảng cách (thường dùng cosine hoặc Euclidean).
* : Một hàm ràng buộc (regularization) đối với mô hình giải thích g. Nó đảm bảo rằng mô hình giải thích không quá phức tạp. Ví dụ, nếu g là một mô hình hồi quy tuyến tính, hàm này có thể là một ràng buộc L1 (Lasso) hoặc L2 (Ridge) để hạn chế số lượng đặc trưng sử dụng trong mô hình, giúp mô hình dễ hiểu và không bị overfitting.
* f là mô hình phức tạp mà bạn muốn giải thích. Đây có thể là bất kỳ mô hình học máy nào, như mạng nơ-ron, SVM, hoặc random forest.
* g là mô hình đơn giản mà bạn sử dụng để giải thích dự đoán của mô hình phức tạp f, g có thể là một mô hình đơn giản hơn, như hồi quy tuyến tính, cây quyết định, hay k-nearest neighbors.
* Quy trình giải thích mô hình LIME
* Lựa chọn điểm cần giải thích: Trước tiên, ta chọn một điểm dữ liệu cụ thể x, được biểu diễn bởi vector các đặc trưng , ,…, Đây là điểm mà ta muốn tìm hiểu và giải thích quyết định mà mô hình phức tạp f đã đưa ra.
* Tạo các điểm dữ liệu lân cận: Từ điểm x, ta tạo ra một tập hợp các điểm dữ liệu lân cận , ,… Mỗi điểm lân cận này được biểu diễn bởi các đặc trưng , ,… tương ứng với x.
* Tính trọng số cho từng điểm lân cận: Mỗi điểm z trong tập dữ liệu lân cận được gán một trọng số , phản ánh mức độ tương đồng giữa z và điểm gốc x. Trọng số này thường được tính bằng hàm Gaussian kernel, trong đó các điểm càng gần x sẽ có trọng số càng cao. Tham số σ xác định độ rộng của vùng lân cận và ảnh hưởng đến mức độ phân rã của trọng số.
* Trong đó D (x, z) là khoảng cách giữa x và z, còn σ điều chỉnh độ rộng của vùng lân cận.
* Huấn luyện mô hình tuyến tính cục bộ: Dựa trên các điểm lân cận và trọng số đã tính, các mô hình tuyến tính đơn giản g(z) = (mỗi mô hình g(z) có số đặc trưng khác nhau ví dụ mô hình g(z) (1) = mô hình g(z) (2) = + ,..) được huấn luyện sao cho mô phỏng lại quyết định của mô hình gốc f xung quanh điểm x.
* Tối ưu hóa hàm mục tiêu: Cuối cùng, các mô hình g được tối ưu bằng cách cực tiểu hóa hàm mục tiêu, mô hình g(z) nào có số phạt nhỏ nhất thì sẽ được chọn.

Trong đó:

* Thành phần đầu tiên đo sai lệch giữa mô hình tuyến tính g và mô hình gốc f trên các điểm lân cận, có trọng số.
* Thành phần ​ là hàm chính quy hóa (regularization) nhằm giảm số lượng đặc trưng được chọn và giúp mô hình dễ hiểu hơn.
* Sau khi tối ưu xong thì một mô hình g(z) sẽ được chọn ví dụ như g(z) = 0.5 + 0.4. Vậy các chỉ số 0.4, 0.3, -0.5 là mức độ quan trọng của từng đặc trưng mà mô hình f dựa vào để dự đoán.
* **Ý nghĩa:**
* LIME không giải thích toàn bộ mô hình mà chỉ giải thích một điểm cụ thể x.
* Nó tạo ra dữ liệu nhiễu xung quanh x, dự đoán chúng bằng mô hình f, sau đó dùng các điểm đó để huấn luyện mô hình g.
* Kết quả là mô hình g thể hiện trọng số của từng đặc trưng và có thể được diễn giải dễ dàng (như trọng số hồi quy).

### **3.1.4. Xây dựng giao diện**

(11) Một giao diện người dùng dễ sử dụng sẽ được phát triển để người dùng có thể tương tác với mô hình. Giao diện này cần cho phép người dùng nhập vào các chuỗi peptide và nhận được kết quả dự đoán cùng với các giải thích liên quan. Các kết quả dự đoán sẽ được hiển thị một cách rõ ràng, giúp người dùng dễ dàng hiểu được các đặc trưng của chuỗi peptide mà mô hình đã phân tích. Bên cạnh đó, giao diện cũng sẽ cung cấp các công cụ trực quan hóa các giải thích từ mô hình, chẳng hạn như biểu đồ hoặc bảng mô tả các yếu tố quan trọng đã ảnh hưởng đến dự đoán. Điều này không chỉ giúp người dùng hiểu rõ hơn về kết quả mà còn tạo ra một trải nghiệm người dùng tốt, giúp họ có thể áp dụng mô hình một cách hiệu quả. Các tính năng tương tác cũng sẽ được phát triển để người dùng có thể thử nghiệm với các chuỗi peptide khác nhau và quan sát sự thay đổi của kết quả dự đoán trong thời gian thực.

## **3.2. Các khó khăn và thử thách**

1. Độ tin cậy của dữ liệu thu thập: Dữ liệu chuỗi peptide được thu thập thông qua giao diện lập trình ứng dụng (API) từ các cơ sở dữ liệu công khai như NCBI [9]. Phương pháp thu thập tự động này giúp tiết kiệm thời gian và đảm bảo tính hệ thống, tuy nhiên vẫn tồn tại một số vấn đề về độ tin cậy và tính nhất quán của dữ liệu. Do các chuỗi peptide được tổng hợp từ nhiều nguồn khác nhau, một số chuỗi có thể không rõ ràng về mặt sinh học, thiếu thông tin đi kèm hoặc chưa được xác thực. Vì vậy, sau khi thu thập, dữ liệu cần phải được xử lý thêm để lọc bỏ các trường hợp không đạt yêu cầu nhằm đảm bảo chất lượng đầu vào cho mô hình học sâu.
2. Xử lý các chuỗi dài hơn 50 amino acid: Theo yêu cầu của bài toán, chỉ sử dụng các chuỗi peptide có độ dài từ 10 đến 50 amino acid [38]. Đối với các chuỗi có độ dài lớn hơn 50, cần phải cắt tách thành các đoạn nhỏ sao cho vẫn đảm bảo tính sinh học hợp lệ. Tuy nhiên, quá trình tách chuỗi này không thể thực hiện một cách ngẫu nhiên mà cần tuân theo các nguyên tắc sinh hóa nhất định, chẳng hạn như vị trí cắt không phá vỡ cấu trúc chức năng hoặc vùng hoạt động của peptide. Việc xác định các quy tắc này đòi hỏi hiểu biết chuyên sâu về sinh học phân tử, khiến quá trình xử lý dữ liệu trở nên phức tạp và tốn nhiều thời gian để tìm hiểu tài liệu chuyên ngành cũng như thử nghiệm các phương án khác nhau.
3. Trích xuất đặc trưng sinh học (Global Protein Sequence Descriptor): Để phục vụ cho mô hình học sâu, việc chuyển đổi chuỗi amino acid thành các đặc trưng số là bước thiết yếu [22]. Nhóm sử dụng phương pháp Global Protein Sequence Descriptor để trích xuất các đặc trưng thống kê và sinh hóa từ chuỗi peptide. Tuy nhiên, để hiểu và áp dụng đúng các chỉ số mô tả này, cần có kiến thức nền vững chắc về sinh học, đặc biệt là về tính chất vật lý – hóa học của các amino acid. Do không chuyên về lĩnh vực sinh học, quá trình này đòi hỏi thời gian nghiên cứu tài liệu, làm quen với các công cụ trích xuất, và kiểm tra kết quả đầu ra nhằm đảm bảo tính hợp lý của đặc trưng tạo ra.
4. Hạn chế về dữ liệu huấn luyện và yêu cầu của mô hình Transformer: Mô hình Transformer là một kiến trúc mạnh, thường đạt hiệu quả cao trong các bài toán xử lý chuỗi, tuy nhiên lại yêu cầu lượng lớn dữ liệu huấn luyện để có thể phát huy hiệu suất tối đa [42]. Với dữ liệu peptide vốn hạn chế cả về số lượng lẫn tính đa dạng, việc huấn luyện mô hình Transformer gặp nhiều trở ngại. Hơn nữa, quá trình huấn luyện yêu cầu tài nguyên tính toán lớn, đặc biệt là khi số lượng lớp và kích thước đầu vào tăng lên, dẫn đến thời gian huấn luyện kéo dài và dễ gặp lỗi nếu không được tối ưu tốt.
5. Hạn chế về phần cứng và chi phí tính toán: Do máy tính cá nhân không có GPU chuyên dụng và bộ nhớ giới hạn, việc huấn luyện mô hình Transformer gặp khó khăn nghiêm trọng. Để khắc phục, nhóm sử dụng dịch vụ máy ảo AWS EC2 với cấu hình RAM 16GB. Tuy nhiên, việc cấu hình môi trường trên AWS EC2 tiêu tốn rất nhiều thời gian, gặp nhiều lỗi vặt liên quan đến thư viện, trình điều khiển GPU, và truy cập dữ liệu từ máy cục bộ. Bên cạnh đó, quá trình truyền dữ liệu qua lại giữa máy cục bộ và máy ảo phức tạp và chậm, ảnh hưởng đến tiến độ công việc. Mặc dù cấu hình phần cứng đã được nâng cấp, kernel Jupyter trên máy ảo vẫn thường xuyên bị ngắt đột ngột trong quá trình huấn luyện, gây mất tiến trình và buộc phải khởi động lại. Đây là yếu tố tiêu tốn nhiều thời gian nhất của dự án. Ngoài ra, chi phí thuê máy ảo cũng khá cao, gây áp lực về tài chính nếu phải sử dụng lâu dài cho các lần huấn luyện và thử nghiệm khác nhau [3].

## **3.3. Kết quả đạt được**

Thu thập dữ liệu sinh học được thực hiện thông qua API chính thức của NCBI, kết hợp với thư viện BioPython nhằm đảm bảo việc truy xuất dữ liệu peptide được thực hiện một cách tự động, chính xác và có hệ thống. Cách tiếp cận này giúp tiết kiệm thời gian và giảm thiểu sai sót trong quá trình thu thập dữ liệu thô từ các nguồn dữ liệu lớn.

Đối với các chuỗi peptide có độ dài vượt quá 50 amino acid thì sẽ được áp dụng phương pháp tính toán sinh hóa để tách chuỗi thành các đoạn nhỏ hơn và các đoạn nhỏ peptide này phải thỏa điều kiện sinh hóa. Việc phân mảnh chuỗi không chỉ giúp tối ưu hóa đầu vào cho mô hình học sâu mà còn đảm bảo được tính ổn định của chuỗi peptide khi có độ dài khoảng 10 đến 50 amino acid và các đặc tính sinh học quan trọng của peptide được bảo toàn trong quá trình xử lý.

Các đặc trưng số của peptide được xây dựng dựa trên phương pháp Global Protein Sequence Descriptor (GPSD). Phương pháp này cho phép chuyển đổi chuỗi amino acid thành biểu diễn số, tạo ra tập dữ liệu đặc trưng giàu thông tin phục vụ cho việc huấn luyện các mô hình học máy một cách hiệu quả.

Mô hình học sâu Transformer được sử dụng để huấn luyện trên ba tập dữ liệu khác nhau là bộ dữ liệu từ GenBank, bộ tổng hợp gồm các nguồn PDB, RefSeq, UniProtKB, Swiss-Prot, DDBJ, EMBL và PIR, và bộ dữ liệu kết hợp giữa hai nhóm trên. Việc so sánh hiệu quả giữa các bộ dữ liệu giúp đánh giá tính khái quát và hiệu suất của mô hình trong các kịch bản dữ liệu đa dạng.

Tích hợp phương pháp Explainable AI bằng công cụ LIMEE, cho phép giải thích rõ ràng các quyết định dự đoán của Transformer. Nhờ đó, mô hình không chỉ đạt hiệu quả cao mà còn minh bạch, hỗ trợ người dùng hiểu được các yếu tố ảnh hưởng đến kết quả đầu ra.

Toàn bộ mô hình được triển khai trên nền tảng Amazon Elastic Compute Cloud (Amazon EC2) với GPU của NVIDIA. Việc sử dụng môi trường tính toán mạnh mẽ giúp đẩy nhanh quá trình huấn luyện mô hình, đồng thời đảm bảo khả năng xử lý các tập dữ liệu lớn với độ chính xác và hiệu suất cao.

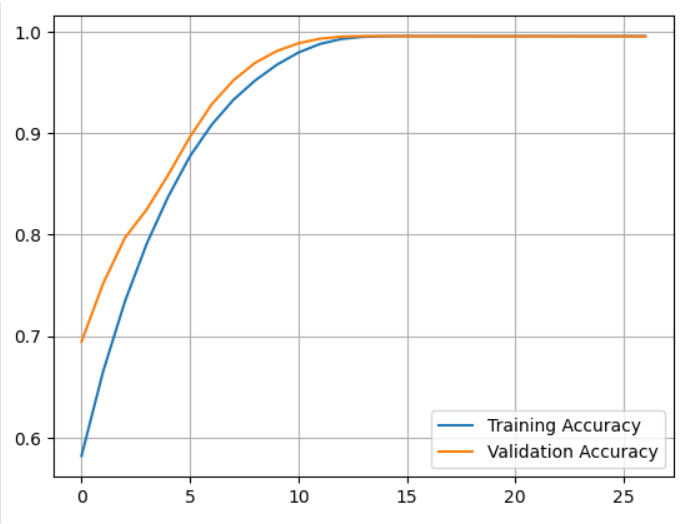
### **3.3.1. Kết quả huấn luyện**

Việc huấn luyện mô hình Transformer có kết quả tổng thể cực kì cao do dữ liệu được thu thập khá nhiều và đã được xử lý chuẩn để mô hình có thể học được dữ liệu một cách dễ dàng hơn.

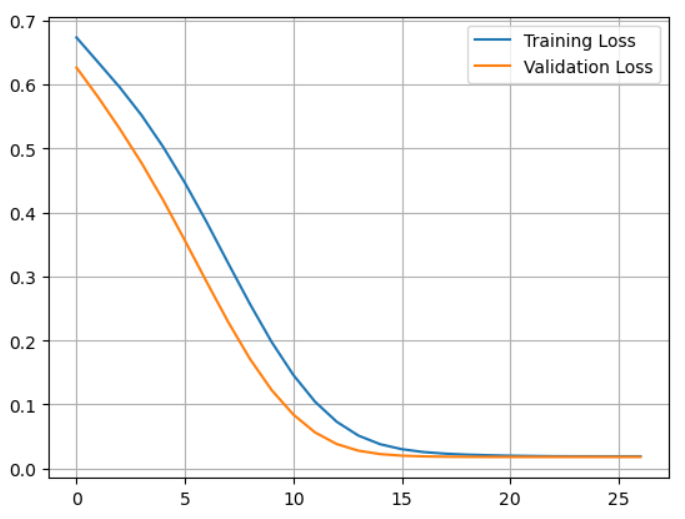
Huấn luyện trên hai bộ dữ liệu, một bộ là GenkBank và bộ còn lại bao gồm PDB, RefSeq, UnitProtKB, Swisss-Prot, DDBJ, EMBL, PIR cho kết quả tổng thể tương đương nhau chênh lệch dường như không có nhưng thời gian huấn luyện bộ dữ liệu thứ hai lâu hơn.

Kết quả huấn luyện trên bộ dữ liệu tổng hợp cả hai bộ cũng cho kết quả tương đương như khi huấn luyện riêng từng bộ nhưng loss có vẻ ít hơn nhưng không đáng kể khi huấn luyện riêng từng bộ.

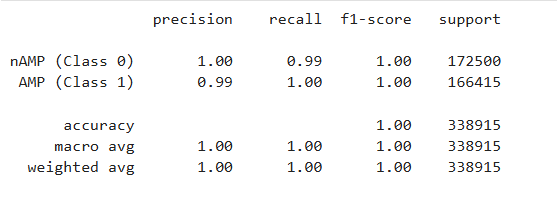
#### **3.3.1.1. Kết quả huấn luyện của mô hình Transformer trên bộ dữ liệu GenBank**



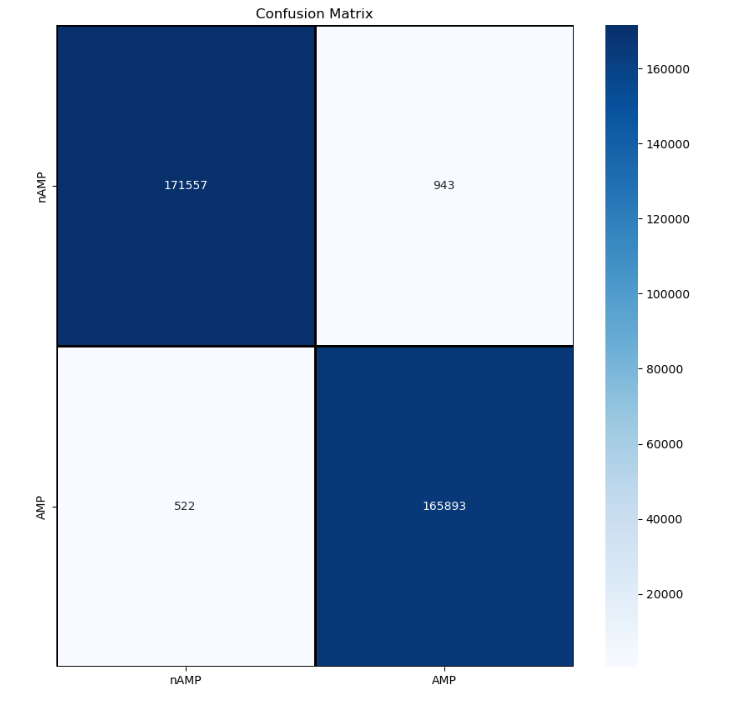
Hình 3. 1. Hình ảnh kết quả huấn luyện trên bộ dữ liệu GenBank (accuracy)



Hình 3. 2. Hình ảnh kết quả huấn luyện trên bộ dữ liệu GenBank (loss)

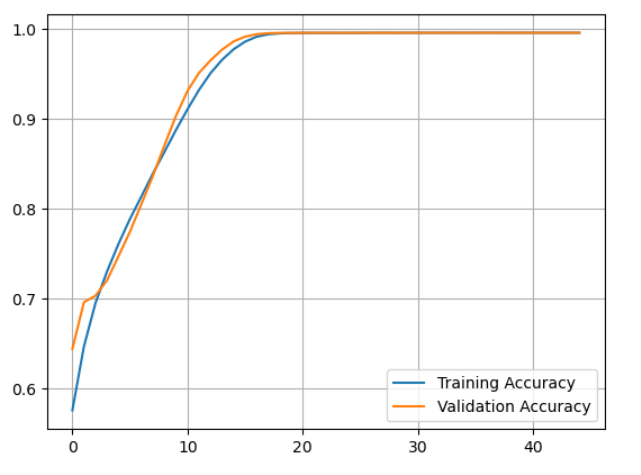


Hình 3. 3. Hình ảnh kết quả huấn luyện trên bộ dữ liệu GenBank (đánh giá hiệu suất)

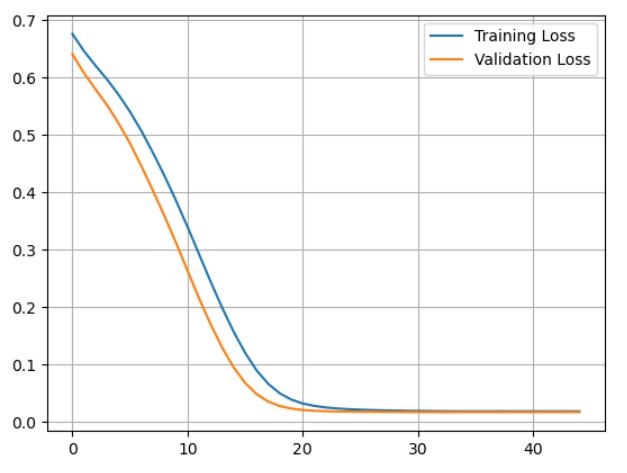


Hình 3. 4. Hình ảnh kết quả huấn luyện trên bộ dữ liệu GenBank (ma trận nhầm lẫn)

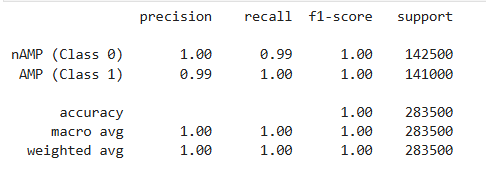
#### **3.3.2.1. Kết quả huấn luyện của mô hình Transformer trên bộ dữ liệu (PDB, RefSeq, UnitProtKB, Swisss-Prot, DDBJ, EMBL, PIR)**



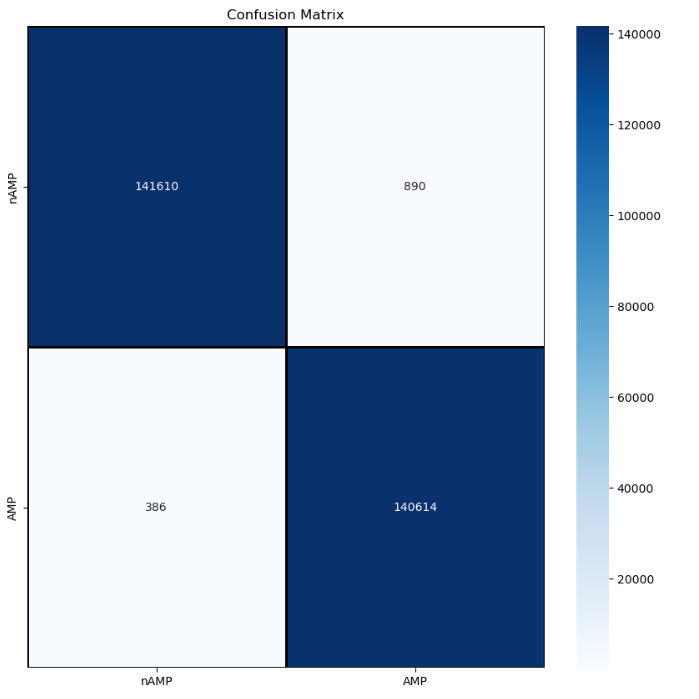
Hình 3. 5. Hình ảnh kết quả huấn luyện trên bộ dữ liệu PDB, RefSeq, UnitProtKB, Swisss-Prot, DDBJ, EMBL, PIR (accuracy)



Hình 3. 6. Hình ảnh kết quả huấn luyện trên bộ dữ liệu PDB, RefSeq, UnitProtKB, Swisss-Prot, DDBJ, EMBL, PIR (loss)

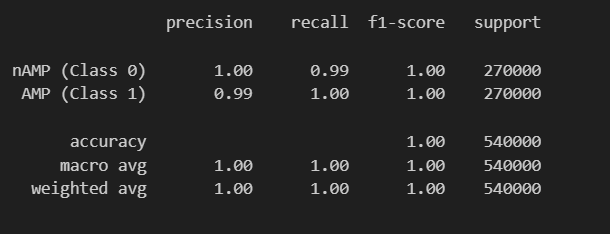


Hình 3. 7. Hình ảnh kết quả huấn luyện trên bộ dữ liệu PDB, RefSeq, UnitProtKB, Swisss-Prot, DDBJ, EMBL, PIR (đánh giá hiệu suất)

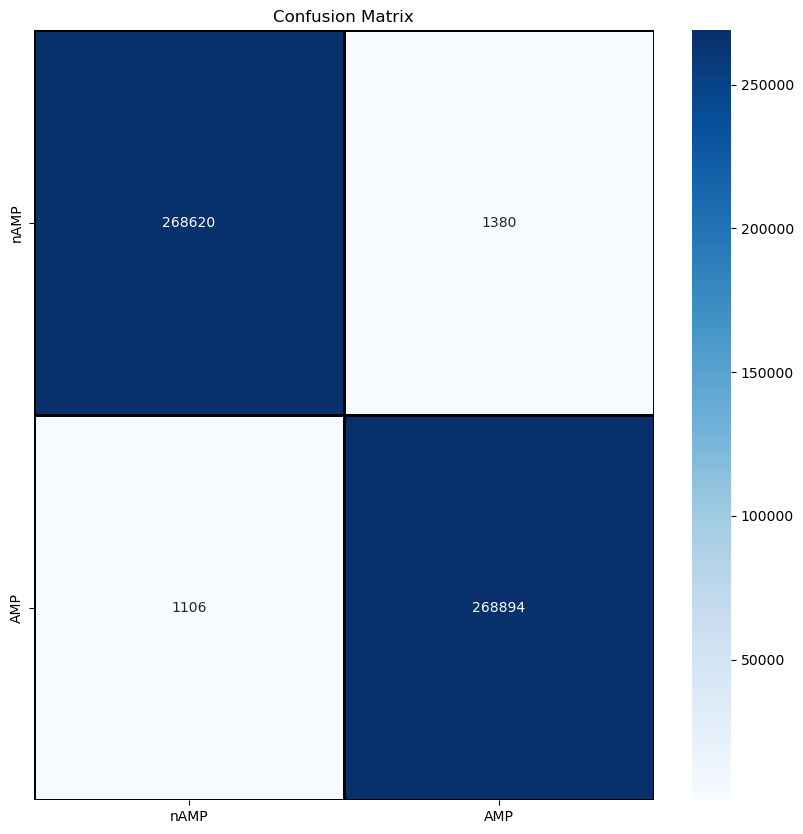


Hình 3. 8. Hình ảnh kết quả huấn luyện trên bộ dữ liệu PDB, RefSeq, UnitProtKB, Swisss-Prot, DDBJ, EMBL, PIR (ma trận nhầm lẫn)

#### **3.3.3.1. Kết quả huấn luyện của mô hình Transformer trên bộ dữ liệu kết hợp**

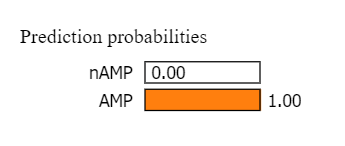


Hình 3. 9. Hình ảnh kết quả huấn luyện trên bộ dữ liệu kết hợp (đánh giá hiệu suất)

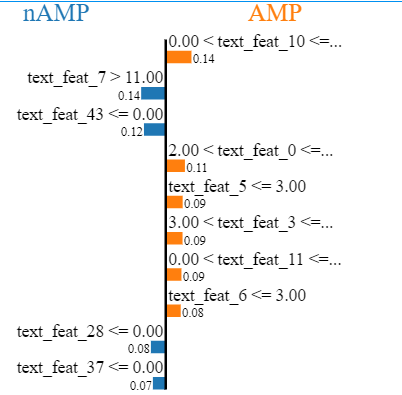


Hình 3. 10. Hình ảnh kết quả huấn luyện trên bộ dữ liệu kết hợp (ma trận nhẫm lẫn)

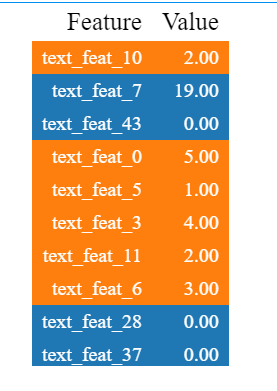
### **3.3.2. Kết quả giải thích dự đoán của mô hình Transformer qua Explainable AI bằng phương pháp LIME**



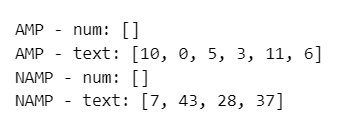
Hình 3. 11. Hình ảnh kết quả giải thích dự đoán của mô hình Transformer bằng LIME (1)



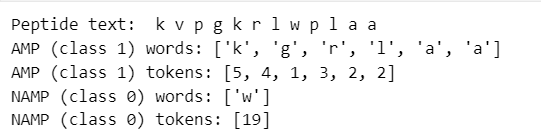
Hình 3. 12. Hình ảnh kết quả giải thích dự đoán của mô hình Transformer bằng LIME (2)



Hình 3. 13. Hình ảnh kết quả giải thích dự đoán của mô hình Transformer bằng LIME (3)



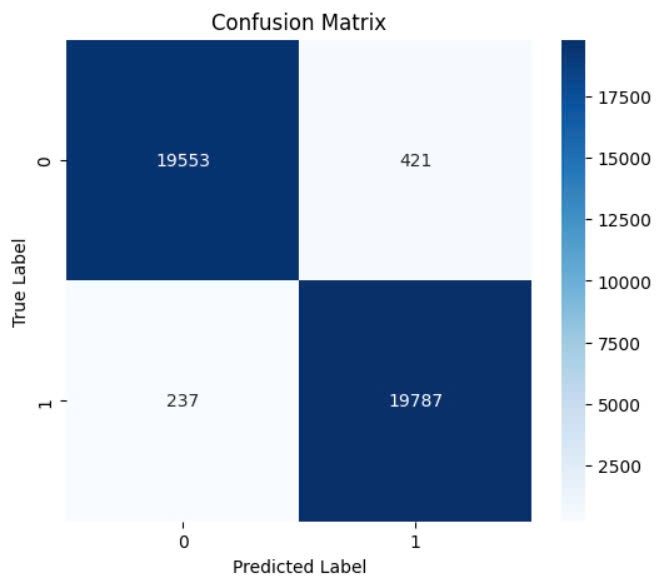
Hình 3. 14. Hình ảnh kết quả giải thích dự đoán của mô hình Transformer bằng LIME (4)



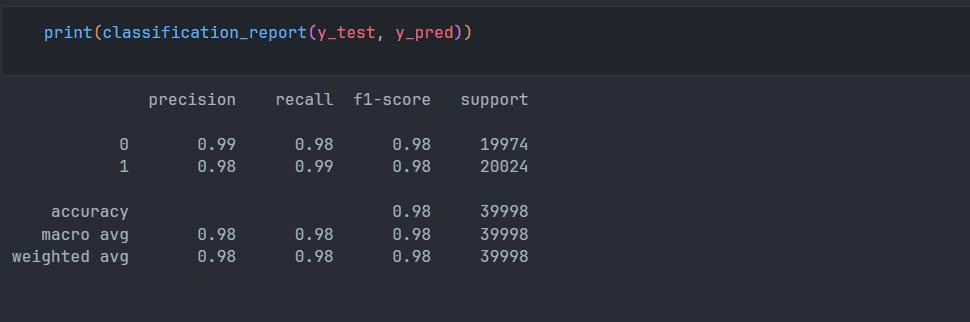
Hình 3. 15. Hình ảnh kết quả giải thích dự đoán của mô hình Transformer bằng LIME (5)

### **3.3.3. So sánh kết quả huấn luyện của mô hình Transformer với mô hình LSTM**

Kết quả huấn luyện của LSTM:



Hình 3. 16. Hình ảnh kết quả huấn luyện bằng mô hình LSTM (ma trận nhầm lẫn)



Hình 3. 17. Hình ảnh kết quả huấn luyện bằng mô hình LSTM (đánh giá hiệu suất)

**Kết luận:** Kết quả huấn luyện của mô hình LSTM kém hơn tổng thể so với mô hình Transformer và độ chính xác (accuracy) đạt ngưỡng 98% kém hơn so với mô hình Transformer trên bộ dữ liệu GenBank và bộ dữ liệu bao gồm PDB, RefSeq, UnitProtKB, Swisss-Prot, DDBJ, EMBL, PIR đạt ở ngưỡng 99.54% và 99.56%. Tuy nhiên số lượng dữ liệu để huấn luyện mô hình LSTM ít hơn khoảng gần 10 lần so với mô hình Transformer.

## **3.4. Hạn chế và đề xuất cải thiện**

Mặc dù Transformer là một mô hình mạnh mẽ và đạt hiệu quả cao trong nhiều bài toán học sâu, nhưng mô hình này vẫn tồn tại một số hạn chế đáng chú ý, đặc biệt trong các ứng dụng yêu cầu tài nguyên tính toán lớn và tập dữ liệu phong phú. Trước hết, Transformer yêu cầu một lượng dữ liệu huấn luyện lớn để có thể học và nhận diện các đặc trưng phức tạp một cách hiệu quả.

Ngoài ra, một yếu tố quan trọng khác là việc tiêu tốn tài nguyên tính toán của Transformer. Với kiến trúc có số lượng tham số lớn và cơ chế attention phức tạp, mô hình yêu cầu phần cứng mạnh mẽ để thực hiện quá trình huấn luyện hiệu quả. Nếu không có phần cứng chuyên dụng như GPU hoặc TPU, việc huấn luyện Transformer trở nên rất tốn thời gian và đòi hỏi nguồn tài nguyên đáng kể. Điều này không chỉ ảnh hưởng đến hiệu suất mà còn có thể làm giảm khả năng ứng dụng mô hình trong các tình huống thiếu thốn về cơ sở hạ tầng.

Để khắc phục những hạn chế này, việc sử dụng các giải pháp điện toán đám mây có thể được coi là một hướng đi khả thi. Trong nghiên cứu này, dịch vụ AWS EC2 (Elastic Compute Cloud) đã được tận dụng để cung cấp môi trường tính toán mạnh mẽ, hỗ trợ quá trình huấn luyện mô hình mà không cần phải đầu tư vào phần cứng đắt đỏ. Các nền tảng điện toán đám mây như AWS, Google Cloud, hay Azure cung cấp sự linh hoạt về tài nguyên, giúp các nhóm nghiên cứu không có khả năng trang bị phần cứng mạnh mẽ có thể tiếp cận với công nghệ tiên tiến mà không gặp phải rào cản lớn về chi phí.

# **CHƯƠNG 4: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN**

Qua quá trình nghiên cứu và thực hiện đồ án “Phát triển mô hình học sâu kết hợp Explainable AI để dự đoán và phân tích đặc trưng chuỗi peptide”, đề tài đã phần nào đáp ứng được các mục tiêu đặt ra trong đề cương ban đầu. Cụ thể, mô hình Transformer đã được xây dựng và huấn luyện thành công để phân loại peptide kháng khuẩn và không kháng khuẩn với độ chính xác rất cao và thành công việc tích hợp Explainable AI công cụ LIME vào mô hình Transformer để giải thích dự đoán. Kết quả thu được cho thấy mô hình có khả năng giải quyết tương đối hiệu quả bài toán phân loại sinh học quan trọng này, đồng thời mở ra tiềm năng ứng dụng trong nghiên cứu y sinh, đặc biệt trong bối cảnh kháng kháng sinh ngày càng trở thành vấn đề cấp thiết.

Trong quá trình thực hiện đồ án, nhiều kiến thức lý thuyết đã được củng cố, đặc biệt là kiến thức chuyên sâu về peptide, các chuỗi amino acid đóng vai trò quan trọng trong sinh học phân tử. Bên cạnh đó, việc nghiên cứu và triển khai mô hình học sâu Transformer đã giúp làm rõ hơn cơ chế hoạt động của mạng nơ-ron tự chú ý (self-attention) trong bài toán xử lý chuỗi sinh học. Ngoài ra, việc ứng dụng LIME để giải thích mô hình cũng góp phần nâng cao hiểu biết về lĩnh vực trí tuệ nhân tạo có thể giải thích được (explainable AI), giúp phân tích rõ hơn vai trò của từng axit amin trong chuỗi peptide.

Về mặt kỹ năng thực hành, quá trình làm đồ án đã giúp rèn luyện thêm nhiều kỹ năng mới như: sử dụng dịch vụ điện toán đám mây Amazon EC2 để huấn luyện mô hình trên dữ liệu lớn, cào dữ liệu từ các nguồn sinh học trực tuyến, xử lý và chuẩn hóa chuỗi peptide theo định dạng tiêu chuẩn trong ngành sinh học. Đây là những kỹ năng quan trọng, không chỉ phục vụ cho đề tài mà còn có giá trị trong công việc thực tiễn sau này.

Một số kinh nghiệm thực tiễn cũng đã được tích lũy trong quá trình thực hiện, như việc hiểu rõ hơn về ứng dụng của các peptide kháng khuẩn trong y học và lý do tại sao một chuỗi peptide có thể hoặc không có khả năng kháng khuẩn. Đề tài đã giúp nhận thức rõ hơn về vai trò của các nghiên cứu liên ngành, nơi Công nghệ Thông tin đóng vai trò hỗ trợ phân tích và dự đoán trong các bài toán sinh học phức tạp.

Tuy đạt được nhiều kết quả khả quan, đề tài vẫn còn một số hạn chế. Dữ liệu peptide hiện tại chưa thực sự phong phú, đặc biệt là thiếu các tập dữ liệu chuẩn hóa chuyên sâu về chức năng sinh học. Bên cạnh đó, do điều kiện cơ sở hạ tầng hạn chế, quá trình huấn luyện mô hình buộc phải thực hiện trên nền tảng đám mây, kéo theo chi phí cao. Đây là yếu tố gây khó khăn nếu việc nghiên cứu được mở rộng trong thời gian dài.

Tổng kết lại, đề tài đã hoàn thành các mục tiêu nghiên cứu quan trọng, xây dựng được mô hình Transformer hiệu quả và áp dụng các kỹ thuật giải thích mô hình hiện đại như LIME để giải thích kết quả dự đoán đầu ra của mô hình Transformer. Qua đó, đề tài không chỉ thể hiện được khả năng ứng dụng học sâu vào lĩnh vực sinh học, mà còn góp phần chứng minh tiềm năng của các mô hình AI trong y học hiện đại.

Từ kết quả thực hiện, có thể đề xuất một số hướng phát triển tiếp theo cho đề tài như: mở rộng tập dữ liệu với nguồn dữ liệu peptide đa dạng hơn; tối ưu hóa mô hình để giảm chi phí tính toán; thử nghiệm với các kiến trúc học sâu khác như BERT, BiLSTM hoặc mô hình lai (hybrid model) và cuối cùng là hướng đến xây dựng một mô hình phân tích peptide trực tuyến tích hợp khả năng giải thích mô hình, nhằm phục vụ cho nghiên cứu và ứng dụng trong y sinh học.

# **DANH MỤC TÀI LIỆU THAM KHẢO**

[1] P. Bhadra, J. Yan, J. Li, S. Fong, and S. W. Siu, “AmPEP: Sequence-based prediction of antimicrobial peptides using distribution patterns of amino acid properties and random forest,” *Scientific reports,* vol. 8, no. 1, p. 1697, 2018.

[2] P. J. A. Cock *et al.*, “Biopython: freely available Python tools for computational molecular biology and bioinformatics,” *Bioinformatics,* vol. 25, pp. 1422 - 1423, 2009.

[3] A. Choudhary, “A walkthrough of Amazon Elastic Compute Cloud (Amazon EC2): A Review,” *International Journal for Research in Applied Science and Engineering Technology,* 2021.

[4] T. Developers, “TensorFlow,” *Zenodo,* 2022.

[5] W. E. Djeddi, K. Hermi, S. Ben Yahia, and G. Diallo, “Advancing drug–target interaction prediction: a comprehensive graph-based approach integrating knowledge graph embedding and ProtBert pretraining,” *BMC bioinformatics,* vol. 24, no. 1, p. 488, 2023.

[6] A. Dowling, J. O'Dwyer, and C. C. Adley, “Lime in the limelight,” *Journal of cleaner production,* vol. 92, pp. 13-22, 2015.

[7] A. Elnaggar *et al.*, “Prottrans: Toward understanding the language of life through self-supervised learning,” *IEEE transactions on pattern analysis and machine intelligence,* vol. 44, no. 10, pp. 7112-7127, 2021.

[8] D. Gaspar, P. Silva, and C. Silva, “Explainable AI for Intrusion Detection Systems: LIME and SHAP Applicability on Multi-Layer Perceptron,” *IEEE Access,* vol. 12, pp. 30164-30175, 2024.

[9] L. Y. Geer *et al.*, “The NCBI BioSystems database,” *Nucleic Acids Research,* vol. 38, no. suppl\_1, pp. D492-D496, 2009, doi: 10.1093/nar/gkp858.

[10] A. Graves and A. Graves, “Long short-term memory,” *Supervised sequence labelling with recurrent neural networks,* pp. 37-45, 2012.

[11] M. Grinberg, *Flask web development*. “ O'Reilly Media, Inc.”, 2018.

[12] S. Grossberg, “Recurrent neural networks,” *Scholarpedia,* vol. 8, no. 2, p. 1888, 2013.

[13] D. Gunning, M. Stefik, J. Choi, T. Miller, S. Stumpf, and G.-Z. Yang, “XAI—Explainable artificial intelligence,” *Science robotics,* vol. 4, no. 37, p. eaay7120, 2019.

[14] K. Han, A. Xiao, E. Wu, J. Guo, C. Xu, and Y. Wang, “Transformer in transformer,” *Advances in neural information processing systems,* vol. 34, pp. 15908-15919, 2021.

[15] X. Hao and L. Fan, “ProtT5 and random forests-based viscosity prediction method for therapeutic mAbs,” *European Journal of Pharmaceutical Sciences,* vol. 194, p. 106705, 2024.

[16] R. J. Hoefen and B. C. Berk, “The multifunctional GIT family of proteins,” *Journal of Cell Science,* vol. 119, no. 8, pp. 1469-1475, 2006, doi: 10.1242/jcs.02925.

[17] S. Imambi, K. B. Prakash, and G. Kanagachidambaresan, “PyTorch,” *Programming with TensorFlow: solution for edge computing applications,* pp. 87-104, 2021.

[18] N. Ketkar, “Introduction to keras,” in *Deep learning with python: a hands-on introduction*: Springer, 2017, pp. 97-111.

[19] O. Kramer and O. Kramer, “Scikit-learn,” *Machine learning for evolution strategies,* pp. 45-53, 2016.

[20] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *nature,* vol. 521, no. 7553, pp. 436-444, 2015.

[21] Z. Li, F. Liu, W. Yang, S. Peng, and J. Zhou, “A survey of convolutional neural networks: analysis, applications, and prospects,” *IEEE transactions on neural networks and learning systems,* vol. 33, no. 12, pp. 6999-7019, 2021.

[22] Z. R. Li, H. H. Lin, L. Y. Han, L. Jiang, X. Chen, and Y. Z. Chen, “PROFEAT: a web server for computing structural and physicochemical features of proteins and peptides from amino acid sequence,” *Nucleic Acids Research,* vol. 34, no. suppl\_2, pp. W32-W37, 2006, doi: 10.1093/nar/gkl305.

[23] T.-T. Lin *et al.*, “AI4AMP: an antimicrobial peptide predictor using physicochemical property-based encoding method and deep learning,” *Msystems,* vol. 6, no. 6, pp. e00299-21, 2021.

[24] L. R. Medsker and L. Jain, “Recurrent neural networks,” *Design and Applications,* vol. 5, no. 64-67, p. 2, 2001.

[25] K. F. Medzihradszky, “Peptide sequence analysis,” *Methods in enzymology,* vol. 402, pp. 209-244, 2005.

[26] P. K. Meher, T. K. Sahu, V. Saini, and A. R. Rao, “Predicting antimicrobial peptides with improved accuracy by incorporating the compositional, physico-chemical and structural features into Chou’s general PseAAC,” *Scientific reports,* vol. 7, no. 1, p. 42362, 2017.

[27] E. Mosca, F. Szigeti, S. Tragianni, D. Gallagher, and G. Groh, “SHAP-based explanation methods: a review for NLP interpretability,” in *Proceedings of the 29th international conference on computational linguistics*, 2022, pp. 4593-4603.

[28] K. Nakao, Y. Ogawa, S.-i. Suga, and H. Imura, “Molecular biology and biochemistry of the natriuretic peptide system. I: Natriuretic peptides,” *Journal of hypertension,* vol. 10, no. 9, pp. 907-912, 1992.

[29] M. Z. A. Pon and K. P. KK, “Hyperparameter tuning of deep learning models in keras,” *Sparklinglight Transactions on Artificial Intelligence and Quantum Computing (STAIQC),* vol. 1, no. 1, pp. 36-40, 2021.

[30] W. Python, “Python,” *Python releases for windows,* vol. 24, 2021.

[31] C. Raffel *et al.*, “Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer,” *J. Mach. Learn. Res.,* vol. 21, pp. 140:1-140:67, 2019.

[32] A. Raza, J. Uddin, A. Almuhaimeed, S. Akbar, Q. Zou, and A. Ahmad, “AIPs-SnTCN: Predicting Anti-Inflammatory Peptides Using fastText and Transformer Encoder-Based Hybrid Word Embedding with Self-Normalized Temporal Convolutional Networks,” *Journal of chemical information and modeling,* 2023.

[33] R. R. Selvaraju, A. Das, R. Vedantam, M. Cogswell, D. Parikh, and D. Batra, “Grad-CAM: Why did you say that?,” *arXiv preprint arXiv:1611.07450,* 2016.

[34] P. A. Sharp, “RNA interference—2001,” *Genes & development,* vol. 15, no. 5, pp. 485-490, 2001.

[35] D. Shi *et al.*, “Spatial-Temporal Self-Attention Transformer Networks for Battery State of Charge Estimation,” *Electronics,* 2023.

[36] C. Shorten and T. M. Khoshgoftaar, “A survey on Image Data Augmentation for Deep Learning,” *Journal of Big Data,* vol. 6, pp. 1-48, 2019.

[37] F. Strati, P. Elvinger, T. Kerimoglu, and A. Klimovic, “ML Training with Cloud GPU Shortages: Is Cross-Region the Answer?,” *Proceedings of the 4th Workshop on Machine Learning and Systems,* 2024.

[38] N. Taniguchi, “Amino acids and proteins,” *Medical Biochemistry, Ed,* vol. 3, pp. 5-21, 2010.

[39] Q. Tarantino, J. Foxx, C. Waltz, and L. DiCaprio, *Django unchained*. Sony Pictures Home Entertainment, 2013.

[40] A. Tyagi *et al.*, “CancerPPD: a database of anticancer peptides and proteins,” *Nucleic acids research,* vol. 43, no. D1, pp. D837-D843, 2015.

[41] S. b. Uzayr, “GitHub,” *Mastering Git,* 2022.

[42] A. Vaswani *et al.*, “Attention is all you need,” *Advances in neural information processing systems,* vol. 30, 2017.

[43] D. Veltri, U. Kamath, and A. Shehu, “Deep learning improves antimicrobial peptide recognition,” *Bioinformatics,* vol. 34, no. 16, pp. 2740-2747, 2018.

[44] A. Y. Wang *et al.*, “What Makes a Well-Documented Notebook? A Case Study of Data Scientists’ Documentation Practices in Kaggle,” *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems,* 2021.

[45] A. Waswani *et al.*, “Attention is all you need,” in *NIPS*, 2017.

[46] F. Xu, H. Uszkoreit, Y. Du, W. Fan, D. Zhao, and J. Zhu, “Explainable AI: A brief survey on history, research areas, approaches and challenges,” in *Natural language processing and Chinese computing: 8th cCF international conference, NLPCC 2019, dunhuang, China, October 9–14, 2019, proceedings, part II 8*, 2019: Springer, pp. 563-574.

[47] L. Yu, R. Jing, F. Liu, J. Luo, and Y. Li, “DeepACP: a novel computational approach for accurate identification of anticancer peptides by deep learning algorithm,” *Molecular Therapy Nucleic Acids,* vol. 22, pp. 862-870, 2020.