**TRƯỜNG ĐẠI HỌC TÀI NGUYÊN VÀ MÔI TRƯỜNG TP.HCM**

A logo with a person holding a book

AI-generated content may be incorrect.**KHOA: CÔNG NGHỆ THÔNG TIN**

**ĐỒ ÁN CHUYÊN NGÀNH**

**NGÀNH CÔNG NGHỆ PHẦN MỀM**

**ỨNG DỤNG HỌC SÂU VÀ EXPLAINABLE ARTIFICIAL INTELLIGENCE TRONG PHÂN TÍCH ĐẶC TRƯNG**

**VÀ THIẾT KẾ PEPTIDE**

|  |  |
| --- | --- |
| Giảng viên hướng dẫn: | **ThS. Phạm Trọng Huynh** |
| Sinh viên thực hiện: | **Lâm Thị Phương Thảo** |
| Mã số sinh viên: | **1050080118** |
| Lớp: | **10\_ĐH\_CNPM2** |
| Niên khóa: | **2021 - 2025** |

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

**TRƯỜNG ĐẠI HỌC TÀI NGUYÊN VÀ MÔI TRƯỜNG TP.HCM**

A logo with a person holding a book

AI-generated content may be incorrect.**KHOA: CÔNG NGHỆ THÔNG TIN**

**ĐỒ ÁN CHUYÊN NGÀNH**

**NGÀNH CÔNG NGHỆ PHẦN MỀM**

**ỨNG DỤNG HỌC SÂU VÀ EXPLAINABLE ARTIFICIAL INTELLIGENCE TRONG PHÂN TÍCH ĐẶC TRƯNG**

**VÀ THIẾT KẾ PEPTIDE**

|  |  |
| --- | --- |
| Giảng viên hướng dẫn: | **ThS. Phạm Trọng Huynh** |
| Sinh viên thực hiện: | **Lâm Thị Phương Thảo** |
| Mã số sinh viên: | **1050080118** |
| Lớp: | **10\_ĐH\_CNPM2** |
| Niên khóa: | **2021 - 2025** |

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

**LỜI CẢM ƠN**

Với lòng biết ơn sâu sắc và sự tri ân chân thành, em xin được gửi lời cảm ơn đến Ban Giám hiệu Trường Đại học Tài nguyên và Môi trường TP.HCM, đã kiến tạo môi trường học thuật chuyên nghiệp, tạo điều kiện thuận lợi và những cơ hội quý báu cho em trong suốt quá trình học tập, nghiên cứu và hoàn thiện bản thân. Những giá trị mà nhà trường đã vun đắp là nền tảng vững chắc cho sự phát triển học thuật và nghề nghiệp của em.

Em xin bày tỏ lòng kính trọng và biết ơn đến tập thể quý thầy cô Khoa Công Nghệ Thông Tin - những người đã không chỉ truyền đạt kiến thức chuyên môn mà còn truyền lửa đam mê, khơi dậy tinh thần khoa học, và đặc biệt là nuôi dưỡng tư duy sáng tạo và tinh thần không ngừng học hỏi trong em. Những bài giảng uyên bác, những lời chỉ dạy tận tâm và sự đồng hành không mệt mỏi của quý thầy cô đã trở thành kim chỉ nam quý giá trên hành trình học thuật của em.

Với sự tôn kính đặc biệt, em xin gửi lời tri ân sâu sắc đến thầy Ths. Phạm Trọng Huynh - người hướng dẫn tận tụy đã dành nhiều tâm huyết, thời gian và công sức để chỉ bảo, định hướng và truyền cảm hứng cho em trong suốt quá trình thực hiện đồ án. Những buổi thảo luận chuyên sâu, những góp ý sắc bén và tầm nhìn học thuật rộng mở của thầy đã giúp em vượt qua nhiều thách thức, hoàn thiện đồ án một cách khoa học và chuyên nghiệp. Sự tận tâm và trách nhiệm cao cả của thầy sẽ mãi là tấm gương sáng để em noi theo trong sự nghiệp học thuật và nghề nghiệp tương lai.

Trong quá trình thực hiện đồ án chuyên ngành, mặc dù đã nỗ lực hết mình nhưng em nhận thức rõ rằng bản thân không thể tránh khỏi những hạn chế và thiếu sót. Em chân thành mong nhận được những ý kiến đóng góp quý báu từ Hội đồng đánh giá và quý thầy cô để em có thể tiếp tục hoàn thiện không chỉ đồ án này mà còn cả kiến thức chuyên môn và phương pháp nghiên cứu khoa học trong tương lai.

Với tất cả sự chân thành và lòng biết ơn sâu sắc, Em xin trân trọng cảm ơn!

**Sinh viên thực hiện**

**Lâm Thị Phương Thảo**

**LỜI CAM ĐOAN**

Tôi xin cam đoan rằng đồ án chuyên ngành này là công trình nghiên cứu của riêng tôi. Các số liệu, kết quả nêu trong đồ án là trung thực và chưa từng được công bố trong bất kỳ công trình nào khác.

Tôi đã thực hiện đồ án này một cách độc lập, không sao chép hay sử dụng ý tưởng, kết quả nghiên cứu của người khác mà không trích dẫn nguồn. Những thông tin tham khảo từ các tài liệu, công trình nghiên cứu khác đều được ghi rõ nguồn gốc trong phần tài liệu tham khảo của đồ án.

Tôi hoàn toàn chịu trách nhiệm về tính chính xác và trung thực của những nội dung trình bày trong đồ án này.

**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**

**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 dẫn** | **Chữ ký của GV hướng dẫn** |
| Tuần 1 | Khảo sát và thu thập nguồn dữ liệu từ hệ thống NCBI, nghiên cứu đặc tính dữ liệu | Trường đại học Tài Nguyên Và Môi Trường TP Hồ Chí Minh. |  |  |
| Tuần 2 | Nghiên cứu chuyên sâu về kiến trúc Transformer và công nghệ XAI | Trường đại học Tài Nguyên và Môi Trường TP Hồ Chí Minh. |  |  |
| Tuần 3 | Thực hiện quy trình tiền xử lý dữ liệu và khởi tạo mô hình Transformer | Trường đại học Tài Nguyên và Môi Trường TP Hồ Chí Minh. |  |  |
| Tuần 4 | Điều chỉnh tham số và huấn luyện mô hình | Trường đại học Tài Nguyên và Môi Trường TP Hồ Chí Minh. |  |  |
| Tuần 5 | Tiếp tục quá trình tinh chỉnh và huấn luyện mô hình | Trường đại học Tài Nguyên và Môi Trường TP Hồ Chí Minh. |  |  |
| Tuần 6 | Tối ưu hóa các siêu tham số của mô hình | Trường đại học Tài Nguyên và Môi Trường TP Hồ Chí Minh. |  |  |
| Tuần 7 | Tích hợp XAI | Trường đại học Tài Nguyên và Môi Trường TP Hồ Chí Minh. |  |  |
| Tuần 8 | Hoàn thiện tích hợp XAI và nâng cao chất lượng trực quan hóa | Trường đại học Tài Nguyên và Môi Trường TP Hồ Chí Minh. |  |  |
| Tuần 9 | Biên soạn tài liệu báo cáo đồ án và chuẩn bị slide thuyết trình | Trường đại học Tài Nguyên và Môi Trường TP Hồ Chí Minh. |  |  |
| Tuần 10 | Hoàn thiện báo cáo đồ án và slide trình bày | Trường đại học Tài Nguyên và Môi Trường TP Hồ Chí Minh. |  |  |

**TÓM TẮT ĐỒ ÁN**

Đề tài “Ứng dụng học sâu và Explainable Artificial Intelligence trong phân tích đặc trưng và thiết kế peptide” hướng đến giải quyết vấn đề dự đoán đặc tính sinh học của peptide, một thành phần then chốt trong nhiều cơ chế sinh học và có tiềm năng ứng dụng rộng rãi trong dược phẩm, vaccine, và điều trị bệnh lý. Phương pháp truyền thống để nghiên cứu peptide tốn nhiều thời gian, chi phí và hiệu quả không cao, vì vậy việc ứng dụng các kỹ thuật hiện đại như học sâu và AI có thể tối ưu hóa quá trình này.

*Mục tiêu* chính của đề tài là xây dựng mô hình học sâu có khả năng dự đoán các đặc tính của peptide như tính kháng khuẩn, độc tính, độ ổn định và khả năng thấm màng, đồng thời sử dụng các kỹ thuật XAI để giải thích quá trình dự đoán và xác định các đặc trưng quan trọng ảnh hưởng đến hoạt tính peptide.

*Các kỹ thuật sử dụng gồm có:*

* Tiền xử lý dữ liệu bằng cách cắt chuỗi peptide về độ dài chuẩn 10–50 amino acid.
* Mã hóa trình tự bằng phương pháp embedding.
* Áp dụng tokenizer để xử lý chuỗi đầu vào.
* Xây dựng mô hình học sâu Transformer bằng PyTorch.
* Triển khai mô hình trên AWS EC2 sử dụng GPU, nhằm tăng tốc quá trình huấn luyện do dữ liệu lớn và mô hình phức tạp, vượt quá khả năng xử lý của máy tính cá nhân.
* Sử dụng các kỹ thuật XAI như SHAP để giải thích kết quả mô hình.
* Các thư viện Python được sử dụng bao gồm: PyTorch, SHAP, Scikit-learn, NumPy và Matplotlib

*Thiết kế tổng quát* gồm 3 phần chính: tiền xử lý dữ liệu và biểu diễn peptide, xây dựng và huấn luyện mô hình học sâu, và áp dụng XAI để phân tích kết quả. Các mô hình được đánh giá bằng các chỉ số như độ chính xác, F1-score và khả năng diễn giải.

*Kết quả thực hiện* cho thấy mô hình đạt độ chính xác cao trong việc phân loại peptide kháng khuẩn và không kháng khuẩn, đồng thời các biểu đồ SHAP làm rõ các axit amin hoặc motif có ảnh hưởng mạnh đến dự đoán, từ đó hỗ trợ thiết kế peptide mới.

*Kết luận*, đề tài đã đạt được mục tiêu đề ra cả về mặt kỹ thuật lẫn ứng dụng. Tuy nhiên, cần mở rộng phạm vi nghiên cứu trên các bộ dữ liệu lớn hơn, đa dạng hơn để cải thiện khả năng tổng quát của mô hình. Ngoài ra, việc tích hợp thêm dữ liệu cấu trúc không gian 3D của peptide có thể giúp mô hình đưa ra dự đoán chính xác và thực tế hơn trong tương lai.

**MỤC LỤC**

[MỞ ĐẦU xviii](#_Toc197555473)

[CHƯƠNG I: TỔNG QUAN TÀI LIỆU 1](#_Toc197555474)

[1.1 Lý thuyết 1](#_Toc197555475)

[1.1.1 Tóm tắt cơ sở lý thuyết sử dụng giải quyết vấn đề 1](#_Toc197555476)

[1.1.2 Các công trình nghiên cứu trước đây về phân tích và thiết kế peptide 2](#_Toc197555477)

[1.1.3 Sự khác biệt của đề tài hiện tại 3](#_Toc197555478)

[1.2 Kỹ thuật 4](#_Toc197555479)

[1.2.1 Tổng quan về Anaconda Navigator 4](#_Toc197555480)

[1.2.2 Ngôn ngữ Python 5](#_Toc197555481)

[1.2.3 Tensorflow 6](#_Toc197555482)

[1.2.4 Trung tâm Thông tin Công nghệ Sinh học Quốc gia (NCBI) 7](#_Toc197555483)

[1.2.5 Cấu trúc chuỗi peptide 8](#_Toc197555484)

[1.2.6 Cloud Amazon Web Service (AWS) 10](#_Toc197555485)

[1.2.7 Nghiên cứu kiến trúc mô hình Transformer 12](#_Toc197555486)

[1.2.8 Nghiên cứu Explainable Artificial Intelligence 13](#_Toc197555487)

[CHƯƠNG II: PHƯƠNG PHÁP THỰC HIỆN 19](#_Toc197555488)

[2.1 Yêu cầu kỹ thuật 19](#_Toc197555489)

[2.1.1 Yêu cầu về dữ liệu 19](#_Toc197555490)

[2.1.2 Yêu cầu về mô hình học sâu 19](#_Toc197555491)

[2.1.3 Yêu cầu về Explainable AI (XAI) 19](#_Toc197555492)

[2.2 Sơ đồ phương pháp thực hiện 20](#_Toc197555493)

[2.2.1 Ý tưởng nghiên cứu 20](#_Toc197555494)

[2.2.2 Nguồn dữ liệu 21](#_Toc197555495)

[2.2.3 Thu thập và lưu trữ dữ liệu 21](#_Toc197555496)

[2.2.4 Xử lý dữ liệu 21](#_Toc197555497)

[2.2.5 Mô hình triển khai 22](#_Toc197555498)

[2.2.6 Đánh giá và tối ưu 22](#_Toc197555499)

[2.2.7 Phát triển mô hình 22](#_Toc197555500)

[2.3 Lựa chọn công nghệ và phương án thiết kế 22](#_Toc197555501)

[2.3.1 Công nghệ và phương án đề xuất 22](#_Toc197555502)

[2.3.2 Lý do lựa chọn phương án và công nghệ. 23](#_Toc197555503)

[2.4 Thách thức thuật toán 24](#_Toc197555504)

[2.4.1 Mô tả thuật toán áp dụng 24](#_Toc197555505)

[2.4.2 Khó khăn và giải pháp thay thế 28](#_Toc197555506)

[CHƯƠNG III: CÀI ĐẶT THỰC NGHIỆM, KẾT QUẢ VÀ ĐÁNH GIÁ 30](#_Toc197555507)

[3.1 Quá trình thực hiện và thiết kế mô hình 30](#_Toc197555508)

[3.1.1 Phân tích yêu cầu và lựa chọn giải pháp 30](#_Toc197555509)

[3.1.2 Thiết kế chi tiết 31](#_Toc197555510)

[3.1.3 Triển khai dựa trên mô hình kiến trúc 35](#_Toc197555511)

[3.1.4 Các khó khăn và thách thức trong quá trình thực hiện 44](#_Toc197555512)

[3.2 Kết quả đạt được 44](#_Toc197555513)

[3.2.1 Kết quả triển khai mô hình 44](#_Toc197555514)

[3.2.2 Yêu cầu phần cứng và môi trường cài đặt 45](#_Toc197555515)

[3.3 Đánh giá và phân tích kết quả thử nghiệm 45](#_Toc197555516)

[3.3.1 Phương pháp kiểm tra và đánh giá mô hình 45](#_Toc197555517)

[3.3.2 Phân tích độ chính xác, độ tin cậy của mô hình học sâu 46](#_Toc197555518)

[3.3.3 So sánh kết quả huấn luyện khi kết hợp 2 bộ dữ liệu với kết quả huấn luyện trên từng bộ dữ liệu 55](#_Toc197555519)

[3.3.4 So sánh kết quả huấn luyện của mô hình Transformer với kết quả huấn luyện của mô hình LSTM 57](#_Toc197555520)

[3.3.5 Phân tích khả năng giải thích của mô hình qua Explainable AI SHAP 59](#_Toc197555521)

[3.3.6 Hạn chế và đề xuất cải thiện 60](#_Toc197555522)

[CHƯƠNG IV: KẾT LUẬN 62](#_Toc197555523)

[TÀI LIỆU THAM KHẢO 64](#_Toc197555524)

**DANH MỤC SƠ ĐỒ, HÌNH**

[Sơ đồ 2. 1. Sơ đồ phương pháp thực hiện thiết kế tổng quan hệ thống 20](#_Toc197555445)

[Sơ đồ 2. 2. Luồng thực hiện của mô hình 20](#_Toc197555446)

[Sơ đồ 2. 3. Kiến trúc Transformer 25](#_Toc197555447)

[Sơ đồ 3. 1. Kiến trúc Transformer Encoder 31](#_Toc197555448)

[Sơ đồ 3. 2. Kiến trúc được xây dựng lại từ Transformer Encoder gốc cho phù hợp 32](#_Toc197555449)

[Sơ đồ 3. 3. Cơ chế của Multihead Attention 40](#_Toc197555450)

[Hình 1. 1. Khung đọc mở ORF 9](#_Toc197555451)

[Hình 3. 1. Ví dụ Position Embebđing 38](#_Toc197555452)

[Hình 3. 2. Kết quả huấn luyện mô hình trên bộ dữ liệu Gen Bank 46](#_Toc197555453)

[Hình 3. 3. Kết quả Accuracy được huấn luyện trên bộ dữ liệu Gen Bank 47](#_Toc197555454)

[Hình 3. 4. Kết quả Loss được huấn luyện trên bộ dữ liệu Gen Bank 47](#_Toc197555455)

[Hình 3. 5. Kết quả hiệu năng của mô hình khi được huấn luyện trên bộ dữ liệu Gen Bank 48](#_Toc197555456)

[Hình 3. 6. Ma trận nhầm lẫn khi được huấn luyện trên bộ dữ liệu Gen Bank 49](#_Toc197555457)

[Hình 3. 7. Kết quả huấn luyện trên bộ dữ liệu (PDB, RefSeq, UnitProtKB, Swisss-Prot, DDBJ, EMBL, PIR) 50](#_Toc197555458)

[Hình 3. 8. Kết quả Accuracy khi được huấn luyện trên bộ dữ liệu (PDB, RefSeq, UnitProtKB, Swisss-Prot, DDBJ, EMBL, PIR) 51](#_Toc197555459)

[Hình 3. 9. Kết quả Loss khi được huấn luyện trên bộ dữ liệu (PDB, RefSeq, UnitProtKB, Swisss-Prot, DDBJ, EMBL, PIR) 51](#_Toc197555460)

[Hình 3. 10. Kết quả hiệu năng của mô hình khi được huấn luyện trên bộ dữ liệu (PDB, RefSeq, UnitProtKB, Swisss-Prot, DDBJ, EMBL, PIR) 52](#_Toc197555461)

[Hình 3. 11. Ma trận nhầm lẫn khi được huấn luyện trên bộ dữ liệu (PDB, RefSeq, UnitProtKB, Swisss-Prot, DDBJ, EMBL, PIR) 53](#_Toc197555462)

[Hình 3. 12. Kết quả hiệu năng của mô hình khi được huấn luyện trên bộ dữ liệu kết hợp 54](#_Toc197555463)

[Hình 3. 13. Ma trận nhầm lẫn khi được huấn luyện trên bộ dữ liệu kết hợp 55](#_Toc197555464)

[Hình 3. 14. Kết quả hiệu năng khi được huấn luyện bằng mô hình LSTM 57](#_Toc197555465)

[Hình 3. 15. Ma trận nhầm lẫn khi được huấn luyện bằng mô hình LSTM 57](#_Toc197555466)

[Hình 3. 16. Kết quả giải thích của SHAP trên một mẫu peptide được dự đoán 59](#_Toc197555467)

[Hình 3. 17. Kết quả giải thích được đặc trưng quan trọng của mẫu peptide 60](#_Toc197555468)

**DANH MỤC BẢNG**

[Bảng 1. 1. So sánh SHAP với các phương pháp giải thích khác 18](#_Toc197555420)

[Bảng 2. 1. Mô tả dữ liệu 21](#_Toc197555421)

[Bảng 3. 1. Mô tả sinh ra số lượng đặc trưng từ chuỗi peptide 37](#_Toc197555422)

[Bảng 3. 2. So sánh kết quả huấn luyện khi kết hợp 2 bộ dữ liệu với kết quả huấn luyện trên từng bộ dữ liệu 56](#_Toc197555423)

[Bảng 3. 3. So sánh độ chính xác giữa mô hình Transformer và LSTM 57](#_Toc197555424)

[Bảng 3. 4. So sánh chỉ số Precision, Recall, F1-score giữa mô hình Transformer và LSTM 58](#_Toc197555425)

[Bảng 3. 5. Confusion Matrix của mô hình LSTM 58](#_Toc197555426)

[Bảng 3. 6. Confusion Matrix của mô hình Transformer 58](#_Toc197555427)

**KÍ HIỆU CÁC CỤM TỪ VIẾT TẮT**

|  |  |
| --- | --- |
| Từ viết tắt | Ý nghĩa |
| Add & Norm | Residual Connection and Layer Normalization |
| AI | Artificial Intelligence |
| AMP | Antimicrobial Peptides |
| API | Application Programming Interface |
| AWS | Amazon Web Services |
| BERT | Bidirectional Encoder Representations from Transformers |
| CLI | Command Line Interface |
| CNN | Convolutional Neural Network |
| DDBJ | DNA Data Bank of Japan |
| EC2 | Elastic Compute Cloud |
| EMBL | European Molecular Biology Laboratory |
| FFN | Feed Forward Neural Network |
| FFNN | Feed-Forward Neural Network |
| GELU | Gaussian Error Linear Unit |
| GenBank | Genetic Sequence Database |
| GPT | Generative Pre-trained Transformer |
| GUI | Graphical User Interface |
| K | Key |
| LSTM | Long Short-Term Memory |
| MLP Head | Multi-Layer Perceptron Head |
| NAMP | Non-Antimicrobial Peptides |
| NCBI | National Center for Biotechnology Information |
| NIH | National Institutes of Health |
| NLM | National Library of Medicine |
| NLP | Natural Language Processing |
| PDB | Protein Data Bank |
| PIR | Protein Information Resource |
| ProtBERT | Protein Bidirectional Encoder Representations from Transformers |
| Q | Query |
| ReLU | Rectified Linear Unit |
| RefSeq | Reference Sequence Database |
| RNN | Recurrent Neural Network |
| SDPA | Scaled Dot-Product Attention |
| SHAP | SHapley Additive exPlanations |
| Swiss-Prot | Swiss Protein Database |
| UniProt | Universal Protein Resource |
| UniProtKB | Universal Protein Knowledgebase |
| V | Value |
| XAI | Explainable AI |

**MỞ ĐẦU**

Trong bối cảnh toàn cầu hiện nay, khi nhân loại đang phải đối mặt với những thách thức ngày càng gia tăng từ các bệnh truyền nhiễm mới và sự kháng thuốc của vi sinh vật, việc phát triển các phương pháp điều trị mới, đặc biệt là các chất có nguồn gốc sinh học như peptide, trở nên quan trọng hơn bao giờ hết. Đề tài “Ứng dụng học sâu và Explainable Artificial Intelligence trong phân tích đặc trưng và thiết kế Peptide” hướng đến việc kết hợp công nghệ trí tuệ nhân tạo tiên tiến với lĩnh vực sinh học phân tử, nhằm tạo ra những đột phá trong nghiên cứu peptide và phát triển dược phẩm.

*1. Cơ sở khoa học*

Cơ sở khoa học của đề tài dựa trên những bước tiến quan trọng trong khoa học máy tính và sinh học phân tử, đặc biệt là kỹ thuật học sâu với khả năng vượt trội trong việc xử lý các tập dữ liệu phức tạp, bao gồm cả dữ liệu về peptide. Sự kết hợp giữa công nghệ học sâu và AI có khả năng giải thích (XAI) không đơn thuần chỉ gia tăng độ chính xác trong việc dự đoán tính chất peptide, mà còn cung cấp cái nhìn sâu sắc về cách thức hoạt động của mô hình, giúp các chuyên gia khoa học nắm bắt rõ hơn các cơ chế tương tác giữa peptide và hệ thống sinh học.

*2. Tính thực tiễn của đề tài*

Peptide đóng vai trò then chốt trong nhiều cơ chế sinh học cơ bản, từ phản ứng miễn dịch đến các quá trình sinh lý học phức tạp. Khả năng dự đoán chính xác đặc tính của peptide mang lại tiềm năng to lớn trong phát triển dược phẩm, thiết kế vaccine và tìm kiếm phương pháp điều trị mới. Đặc biệt, peptide kháng khuẩn được công nhận là thành phần quan trọng của hệ thống phòng thủ bẩm sinh ở hầu hết các loài, với nhiều đặc tính ưu việt như khả năng tiêu diệt nhanh chóng vi khuẩn kháng thuốc, mức độ phát triển kháng thuốc thấp, và khả năng bảo vệ cơ thể chống lại nhiễm trùng.

Tuy nhiên, việc nghiên cứu và phát triển peptide theo phương pháp truyền thống đòi hỏi thời gian dài, chi phí lớn và hiệu quả không cao.

*3. Mục tiêu của đề tài*

Mục tiêu của đề tài là phát triển mô hình có khả năng dự đoán đặc trưng peptide một cách chính xác và cung cấp giải thích minh bạch về quyết định của mô hình. Kết quả từ nghiên cứu này không chỉ cung cấp một công cụ tính toán hiệu quả cho việc khám phá peptide, mà còn đóng góp vào sự hiểu biết cơ bản về mối quan hệ giữa cấu trúc và chức năng của peptide, từ đó mở ra con đường mới cho việc phát triển thế hệ thuốc peptide tiếp theo.

*4. Phương pháp nghiên cứu*

Để đạt được mục tiêu của đề tài, nghiên cứu áp dụng phương pháp liên ngành, kết hợp giữa khoa học dữ liệu, trí tuệ nhân tạo, và sinh học phân tử. Cụ thể, các phương pháp nghiên cứu chính bao gồm:

* Phương pháp thu thập và tiền xử lý dữ liệu: Dữ liệu peptide được thu thập từ các cơ sở dữ liệu công khai như UniProt, GenBank,... Các thông tin như trình tự axit amin, đặc tính sinh hóa, hoạt tính kháng khuẩn,... sẽ được chuẩn hóa và xử lý để đảm bảo chất lượng đầu vào cho mô hình học máy.
* Phương pháp học sâu (Deep Learning): Áp dụng mô hình Transformer để trích xuất đặc trưng và dự đoán tính chất của peptide dựa trên trình tự axit amin.
* Phương pháp XAI (Explainable Artificial Intelligence): Tích hợp các kỹ thuật như SHAP (SHapley Additive exPlanations) để giải thích các quyết định của mô hình, từ đó giúp các nhà nghiên cứu hiểu rõ mối liên hệ giữa đặc trưng peptide và chức năng sinh học của chúng.
* Phương pháp đánh giá mô hình: Sử dụng các chỉ số đánh giá như độ chính xác (Accuracy), độ nhạy (Recall), độ đặc hiệu (Specificity), và AUC-ROC để đo lường hiệu suất của mô hình. Ngoài ra, so sánh hiệu quả giữa các mô hình học sâu khác nhau sẽ được thực hiện để lựa chọn mô hình tối ưu.
* Phương pháp trực quan hóa dữ liệu: Biểu đồ được sử dụng để trực quan hóa đặc trưng và cấu trúc dữ liệu, hỗ trợ việc phân tích kết quả và hiểu sâu hơn về mối quan hệ giữa cấu trúc và chức năng peptide.

*5. Nội dung công việc với các phương pháp tương ứng*

Nội dung công việc bao gồm thu thập dữ liệu từ NCBI, tiền xử lý các đặc trưng peptide, phát triển và tối ưu hóa mô hình Transformer, ứng dụng XAI để giải thích quyết định mô hình, và đánh giá hiệu suất dự đoán.

*6. Đối tượng và phạm vi nghiên cứu*

Đối tượng nghiên cứu của đề tài là các peptide có nguồn gốc vi sinh vật, đặc biệt là các peptide kháng khuẩn (Antimicrobial Peptides – AMPs), với các thông tin liên quan đến trình tự axit amin, đặc tính sinh hóa và hoạt tính sinh học. Các peptide được lựa chọn có độ dài chuỗi từ 10 đến 50 axit amin [20], vì đây là khoảng độ dài phổ biến và ổn định trong các nghiên cứu về cấu trúc – chức năng của peptide.

Phạm vi nghiên cứu của đề tài tập trung vào:

* Dữ liệu: Thu thập và xử lý dữ liệu peptide từ các cơ sở dữ liệu công khai như UniProt, GenBank, NCBI và APD3.
* Thuật toán: Ứng dụng mô hình học sâu Transformer [6], để trích xuất đặc trưng và dự đoán hoạt tính sinh học của peptide.
* Giải thích mô hình: Tích hợp kỹ thuật Explainable AI (XAI), cụ thể là SHAP [4], để giải thích các quyết định của mô hình và xác định các yếu tố ảnh hưởng đến hoạt tính peptide.

*7. Kết cấu của báo cáo*

Báo cáo đồ án được cấu trúc thành bốn chương chính:

Chương I: Tổng quan đề tài - Trình bày tổng quan về đề tài, bao gồm cơ sở khoa học, tính cấp thiết, mục tiêu nghiên cứu và đối tượng, phạm vi nghiên cứu.

Chương II: Cơ sở lý thuyết - Trình bày chi tiết về các nền tảng lý thuyết của học sâu và Explainable AI, đặc biệt tập trung vào kiến trúc Transformer và phương pháp XAI SHAP, cùng các kiến thức liên quan đến peptide và ứng dụng trong y dược.

Chương III: Cài đặt thực nghiệm - Mô tả quá trình thực nghiệm, bao gồm thu thập và xử lý dữ liệu từ NCBI, xây dựng và huấn luyện mô hình Transformer, ứng dụng các công cụ XAI và đánh giá hiệu suất dự đoán. Trình bày kết quả thực nghiệm và phân tích chi tiết.

Chương IV: Kết luận - Tổng kết các kết quả đạt được, thảo luận về những đóng góp của nghiên cứu và đề xuất hướng phát triển trong tương lai cho việc ứng dụng học sâu và XAI trong nghiên cứu peptide.

# **CHƯƠNG I: TỔNG QUAN TÀI LIỆU**

## **Lý thuyết**

* + 1. ***Tóm tắt cơ sở lý thuyết sử dụng giải quyết vấn đề***

Trong dự án phân tích peptide: khung lý thuyết của dự án này được xây dựng trên ba trụ cột phương pháp luận chính: kiến trúc Transformer, quá trình khai thác và xử lý dữ liệu từ nguồn NCBI, cùng với tích hợp công nghệ AI có khả năng giải thích (XAI).

Kiến trúc Transformer làm nền tảng xử lý: mô hình Transformer được lựa chọn làm cơ sở kiến trúc chính để phát triển hệ thống học sâu nhằm khám phá và giải mã các đặc tính peptide. Cơ chế tự chú ý (self-attention) đặc trưng của Transformer tạo điều kiện cho việc mô hình hóa các mối tương quan phức tạp giữa các thành tố trong chuỗi peptide với hiệu suất vượt trội. Khác với phương pháp chuyển giao học tập từ các mô hình hiện có, nghiên cứu của chúng tôi chú trọng vào việc thiết kế và xây dựng kiến trúc Transformer từ cơ bản, với các điều chỉnh chuyên biệt để tối ưu hóa cho đặc điểm riêng biệt của dữ liệu peptide. Các tham số kiến trúc như số lượng đầu chú ý (attention heads), chiều sâu mạng (số lớp Transformer), và kích thước véc-tơ nhúng (embedding dimensions) sẽ được thiết kế và hiệu chỉnh một cách có hệ thống dựa trên đặc thù của dữ liệu đầu vào [8].

Chiến lược thu thập và xử lý dữ liệu: nguồn dữ liệu chính cho nghiên cứu được thu thập từ Trung tâm Thông tin Công nghệ Sinh học Quốc gia Hoa Kỳ (NCBI) - một kho lưu trữ dữ liệu sinh học có uy tín và quy mô toàn cầu. Cơ sở dữ liệu hiện đại đại diện cho các trình tự từ hơn 55.000 sinh vật (hơn 4800 vi-rút, hơn 40.000 sinh vật nhân sơ và hơn 10.000 sinh vật nhân chuẩn), từ một bản ghi duy nhất đến toàn bộ bộ gen [16].

Cuối cùng, để cải thiện tính minh bạch của mô hình, phương pháp Explainable AI (XAI) sẽ được áp dụng nhằm giải thích cách Transformer đưa ra dự đoán. Các kỹ thuật như SHAP (SHapley Additive Explanations), LIME (Local Interpretable Model-agnostic Explanations) sẽ giúp đánh giá mức độ ảnh hưởng của từng đặc trưng peptide đến kết quả dự đoán. Ngoài ra, việc sử dụng Attention Visualization giúp phân tích cách Transformer tập trung vào các phần khác nhau của chuỗi peptide trong quá trình học [24]. Nhờ đó, mô hình không chỉ có độ chính xác cao mà còn cung cấp các phân tích trực quan, hỗ trợ các nhà nghiên cứu hiểu rõ hơn về các yếu tố quan trọng trong đặc trưng peptide.

Quá trình thực nghiệm bắt đầu với việc thu thập dữ liệu từ NCBI (National Center for Biotechnology Information). Dữ liệu peptide được lấy bằng thư viện riêng dành cho trang NCBI. Dữ liệu thu thập phải đảm bảo độ chính xác và đầy đủ để làm đầu vào cho quá trình huấn luyện mô hình.

Sau khi thu thập, dữ liệu sẽ trải qua quá trình tiền xử lý, trong đó các trình tự peptide được xử lý bằng tokenizer để tạo ra vocabulary phù hợp. Quá trình này bao gồm việc làm cho peptide thành các mảng số và sẽ phản ánh ra một kí tự được xây dựng từ tokenizer khi nó học từ chuỗi của peptide. Xử lý chuỗi peptide xong thì xử lý các feature dạng số thực khác bằng cách chuẩn hóa giá trị để có thể tăng quá trình học tốt hơn [9].

Tiếp theo, mô hình Transformer sẽ được xây dựng từ đầu thay vì sử dụng mô hình có sẵn. Cấu trúc mô hình bao gồm các thành phần quan trọng như embedding layer để ánh xạ token thành vector, multi-head self-attention để học mối quan hệ giữa các token, và feed-forward layers để xử lý thông tin đặc trưng của peptide. Các yếu tố như hàm kích hoạt, hàm mất mát (loss function), thuật toán tối ưu hóa sẽ được lựa chọn cẩn thận để phù hợp với bài toán phân tích peptide [27].

Quá trình tìm kiếm siêu tham số (Hyperparameter Tuning) sẽ được thực hiện để tối ưu hiệu suất của mô hình, giúp xác định giá trị tối ưu cho learning rate, batch size, số lượng layers, số lượng head nhằm cải thiện độ chính xác và khả năng tổng quát hóa của mô hình.

Khi quá trình huấn luyện hoàn tất, mô hình sẽ được đánh giá bằng các thước đo hiệu suất như accuracy, precision, recall, F1-score, ROC-AUC. Việc kiểm tra trên tập dữ liệu kiểm thử giúp xác định xem mô hình có thể dự đoán đặc trưng peptide một cách chính xác hay không [12].

Cuối cùng, Explainable AI (XAI) sẽ được áp dụng để giải thích cách mô hình đưa ra dự đoán. Các kỹ thuật như SHAP (SHapley Additive Explanations), LIME (Local Interpretable Model-agnostic Explanations) sẽ giúp xác định những đặc trưng peptide nào quan trọng nhất đối với dự đoán [25]. Ngoài ra, Attention Visualization sẽ giúp phân tích cách Transformer tập trung vào các phần quan trọng trong chuỗi peptide, cung cấp cái nhìn sâu sắc hơn về cơ chế hoạt động của mô hình.

* + 1. ***Các công trình nghiên cứu trước đây về phân tích và thiết kế peptide***

Lĩnh vực phân tích và thiết kế peptide đã chứng kiến nhiều bước tiến quan trọng trong những năm gần đây. Một số công trình tiêu biểu bao gồm:

1. *ProteinBERT (Nielsen et al., 2022):* Ứng dụng kiến trúc BERT để phân tích trình tự protein và dự đoán cấu trúc. Nghiên cứu này tập trung vào việc khai thác dữ liệu lớn từ các cơ sở dữ liệu protein công khai nhưng chưa áp dụng cụ thể cho peptide và thiếu các thành phần giải thích [21].
2. *DeepBinding (Wang et al., 2021):* Sử dụng mạng nơ-ron sâu CNN để dự đoán khả năng liên kết của peptide với các thụ thể. Mô hình này đạt được độ chính xác cao nhưng hoạt động như một “hộp đen” thiếu khả năng giải thích [23].
3. *PepVAE (Zhang et al., 2023):* Áp dụng Variational Autoencoder để tạo ra các trình tự peptide mới có hoạt tính sinh học mong muốn. Nghiên cứu đã chứng minh khả năng tạo ra các ứng viên peptide tiềm năng nhưng chưa giải thích được các đặc tính quan trọng quyết định hoạt tính [26].
4. *ToxinPred (Sharma et al., 2022):* Sử dụng Random Forest kết hợp với biểu diễn đặc trưng thủ công để dự đoán độc tính của peptide. Mô hình này có khả năng giải thích nhưng bị hạn chế bởi khả năng học các mẫu phức tạp [19].
   * 1. ***Sự khác biệt của đề tài hiện tại***
        1. Phương pháp học sâu kết hợp với XAI

Trong khi các nghiên cứu trước đây thường tập trung vào một trong hai hướng - hoặc tối ưu hóa hiệu suất dự đoán (như DeepBinding, ProteinBERT) hoặc tối ưu hóa khả năng giải thích (như ToxicPred), thì nghiên cứu trong đồ án này tích hợp cả hai yếu tố này. Bằng cách kết hợp mô hình Transformer hiệu suất cao với phương pháp SHAP, không chỉ đạt được độ chính xác cao mà còn cung cấp được cái nhìn sâu sắc về cơ chế hoạt động của mô hình.

* + - 1. Chiến lược học chuyển tiếp đa dữ liệu

Điểm đột phá trong nghiên cứu là chiến lược huấn luyện đặc biệt:

*Huấn luyện theo giai đoạn:* Khác với các nghiên cứu trước thường huấn luyện trên một bộ dữ liệu duy nhất, tôi triển khai quy trình huấn luyện nhiều giai đoạn:

* Giai đoạn 1: Huấn luyện mô hình ban đầu trên một bộ dữ liệu lớn để nắm bắt các đặc trưng chung của peptide
* Giai đoạn 2: Huấn luyện chuyển tiếp trên nhiều bộ dữ liệu chuyên biệt khác nhau
* Giai đoạn 3: So sánh và phân tích hiệu suất trên các bộ dữ liệu khác nhau

*Tối ưu hóa siêu tham số tích hợp:* Tôi không chỉ chọn hyperparameter tốt nhất cho từng bộ dữ liệu mà còn phát triển một chiến lược tích hợp các cấu hình tối ưu từ nhiều bộ dữ liệu khác nhau, tạo ra một mô hình “meta-optimal” thực sự hiệu quả trên phổ rộng các loại peptide.

* + - 1. Tận dụng cơ chế Self-Attention cho phân tích đặc trưng

Các nghiên cứu trước đây về peptide thường tập trung vào các đặc tính vật lý hóa học hoặc các mẫu cục bộ trong chuỗi. Nghiên cứu của tôi khai thác cơ chế Self-Attention của Transformer để:

* Phát hiện các tương tác xa trong chuỗi peptide mà các phương pháp truyền thống khó nắm bắt
* Xác định chính xác các vùng quan trọng quyết định đến chức năng của peptide
* Trực quan hóa ma trận attention để hiểu rõ hơn về cấu trúc không gian của peptide
  + - 1. Tích hợp dữ liệu tối ưu

Điểm nổi bật của nghiên cứu là cách tiếp cận dữ liệu:

* Các nghiên cứu trước thường sử dụng một bộ dữ liệu duy nhất hoặc gộp tất cả dữ liệu lại
* Nghiên cứu của tôi áp dụng phương pháp chọn lọc có phân tích: sau khi đánh giá hiệu suất trên các bộ dữ liệu riêng biệt, tôi kết hợp lại những bộ dữ liệu có hiệu quả cao nhất thành một tập huấn luyện tối ưu
* Quá trình này không chỉ đơn thuần là gộp dữ liệu mà còn bao gồm phân tích trọng số và tính quan trọng của từng tập dữ liệu

Cách tiếp cận này giúp mô hình vừa nắm bắt được các đặc trưng chung của peptide, vừa học được các đặc tính chuyên biệt từ các nhóm peptide khác nhau, dẫn đến khả năng tổng quát hóa tốt hơn so với các phương pháp truyền thống.

Thông qua việc kết hợp các tiến bộ trong kiến trúc mô hình, phương pháp học tập và chiến lược dữ liệu, nghiên cứu của tôi hướng tới việc không chỉ cải thiện độ chính xác trong phân tích và thiết kế peptide mà còn nâng cao hiểu biết cơ bản về mối quan hệ giữa cấu trúc và chức năng của các phân tử quan trọng này.

## **Kỹ thuật**

* + 1. ***Tổng quan về Anaconda Navigator***

Anaconda Navigator là giao diện người dùng đồ họa (GUI) được tích hợp trong Bản phân phối Anaconda, được thiết kế nhằm cung cấp phương thức trực quan để quản lý các gói phần mềm, môi trường và kênh conda mà không cần sử dụng giao diện dòng lệnh (CLI). Công cụ này cho phép người dùng thực hiện tìm kiếm các gói trên Anaconda.org hoặc trong Kho lưu trữ Anaconda cục bộ một cách hiệu quả. Anaconda Navigator tương thích với nhiều hệ điều hành phổ biến bao gồm Windows, macOS và Linux [11].

Anaconda Navigator cung cấp phương pháp tiếp cận đơn giản theo cơ chế “trỏ và nhấp” (point-and-click) để tương tác với hệ sinh thái conda mà không đòi hỏi kiến thức về các lệnh dòng trong cửa sổ terminal. Người dùng có thể dễ dàng thực hiện các tác vụ thiết yếu như tìm kiếm, cài đặt, vận hành và cập nhật các gói phần mềm trong môi trường được chỉ định - tất cả đều được tích hợp trong một giao diện thống nhất [11].

Anaconda là nền tảng mã nguồn mở và hoàn toàn miễn phí, cung cấp truy cập đến hơn 1.500 gói thư viện Python/R chuyên dụng cho khoa học dữ liệu. Nền tảng này hỗ trợ xây dựng và phát triển các mô hình học máy và học sâu thông qua các thư viện tiên tiến như scikit-learn, TensorFlow và Theano. Bên cạnh đó, người dùng có thể tận dụng Dask, NumPy, Pandas và Numba để thực hiện phân tích dữ liệu với hiệu suất cao. Nền tảng này đóng vai trò quan trọng trong việc đơn giản hóa quy trình quản lý và triển khai gói phần mềm. Anaconda được trang bị các công cụ chuyên biệt cho phép thu thập dữ liệu từ nhiều nguồn khác nhau thông qua các kỹ thuật học máy và trí tuệ nhân tạo [11].

* + 1. ***Ngôn ngữ Python***

Python là một chương trình thông dịch, hướng đối tượng, cấp cao ngôn ngữ với ngữ nghĩa động. Dữ liệu tích hợp sẵn ở mức cao của nó. Cấu trúc, kết hợp với gõ động và ràng buộc động, làm cho nó rất hấp dẫn để phát triển ứng dụng nhanh chóng, cũng như để sử dụng như một ngôn ngữ kịch bản. Cú pháp đơn giản, dễ học của Python nhấn mạnh khả năng đọc và do đó giảm chi phí bảo trì chương trình. Python hỗ trợ các mô-đun và gói, khuyến khích chương trình mô-đun và tái sử dụng mã. Trình thông dịch Python và mở rộng Thư viện tiêu chuẩn có sẵn ở dạng nguồn hoặc nhị phân miễn phí cho tất cả các nền tảng chính và có thể được phân phối tự do [22].

Thông thường, các lập trình viên yêu Python vì tăng năng suất nó cung cấp. Vì không có biên dịch Bước, chu kỳ chỉnh sửa-kiểm tra-gỡ lỗi cực kỳ nhanh. Gỡ lỗi Python Chương trình rất dễ dàng: một lỗi hoặc đầu vào xấu sẽ không bao giờ gây ra phân đoạn lỗi. Thay vào đó, khi phiên dịch viên phát hiện ra một lỗi, nó sẽ đưa ra một ngoại lệ. Khi chương trình không bắt được ngoại lệ, Trình thông dịch in dấu vết ngăn xếp. Trình gỡ lỗi cấp nguồn cho phép kiểm tra các biến địa phương và toàn cầu, đánh giá tùy ý biểu thức, thiết lập điểm ngắt, bước qua mã một dòng tại một lần, và như vậy. Trình gỡ lỗi được viết bằng chính Python, làm chứng cho sức mạnh nội tâm của Python. Mặt khác, thường Cách nhanh nhất để gỡ lỗi một chương trình là thêm một vài câu lệnh in Đến nguồn: Chu trình chỉnh sửa-kiểm tra-gỡ lỗi nhanh làm cho điều này trở nên đơn giản Cách tiếp cận rất hiệu quả [22].

Python là một ngôn ngữ lập trình phổ biến. Nó được tạo ra bởi Guido van Rossum, và phát hành vào năm 1991. Nó được sử dụng cho:

* Phát triển web (phía máy chủ),
* Phát triển phần mềm,
* Toán học
* Kịch bản hệ thống.

Python có thể được sử dụng trên một máy chủ để tạo các ứng dụng web. Python có thể được sử dụng cùng với phần mềm để tạo quy trình công việc. Python có thể kết nối với các hệ thống cơ sở dữ liệu. Nó cũng có thể đọc và sửa đổi các tập tin. Python có thể được sử dụng để xử lý dữ liệu lớn và thực hiện toán học phức tạp [22]. Python có thể được sử dụng để tạo mẫu nhanh hoặc để phát triển phần mềm sẵn sàng sản xuất.

Python hoạt động trên các nền tảng khác nhau (Windows, Mac, Linux, Raspberry Pi, v.v.). Python có cú pháp đơn giản tương tự như ngôn ngữ tiếng Anh. Python có cú pháp cho phép các nhà phát triển viết các chương trình với ít dòng hơn một số ngôn ngữ lập trình khác. Python chạy trên một hệ thống thông dịch, có nghĩa là mã có thể được thực thi ngay khi nó được viết. Điều này có nghĩa là việc tạo mẫu có thể rất nhanh. Python có thể được xử lý theo cách thủ tục, cách hướng đối tượng hoặc cách chức năng [22].

* + 1. ***Tensorflow***

TensorFlow là một thư viện phổ biến để triển khai các giải pháp dựa trên máy học. Nó bao gồm một API cấp thấp được gọi là lõi TensorFlow và nhiều API cấp cao, bao gồm hai API phổ biến nhất, được gọi là TensorFlow Estimators và Keras. Trong chương này, chúng ta sẽ tìm hiểu về những điều cơ bản của TensorFlow và xây dựng một mô hình học máy sử dụng hồi quy logistic để phân loại các chữ số viết tay làm ví dụ [10].

TensorFlow là một thư viện mã nguồn mở phổ biến được sử dụng để triển khai học máy và học sâu. Ban đầu, nó được xây dựng tại Google để sử dụng nội bộ và được phát hành công khai vào ngày 9 tháng 11 năm 2015. Kể từ đó, TensorFlow đã được sử dụng rộng rãi để phát triển các mô hình học máy và học sâu trong một số lĩnh vực kinh doanh [10].

Để sử dụng TensorFlow trong các dự án của mình, chúng ta cần học cách lập trình bằng API TensorFlow. TensorFlow có nhiều API có thể được sử dụng để tương tác với thư viện. API TensorFlow được chia thành hai cấp độ [10]:

* API cấp thấp: API được gọi là lõi TensorFlow cung cấp chức năng cấp thấp chi tiết. Vì lý do này, API cấp thấp này cung cấp khả năng kiểm soát hoàn toàn khi được sử dụng trên các mô hình. Chúng tôi sẽ đề cập đến lõi TensorFlow trong chương này.
* API cấp cao: Các API này cung cấp các chức năng cấp cao được xây dựng trên lõi TensorFlow và tương đối dễ học và triển khai. Một số API cấp cao bao gồm Estimators, Keras, TFLearn, TFSlim và Sonnet. Chúng tôi cũng sẽ đề cập đến Keras trong chương này.
  + 1. ***Trung tâm Thông tin Công nghệ Sinh học Quốc gia (NCBI)***

Trung tâm Thông tin Công nghệ Sinh học Quốc gia (NCBI) là một nhóm nghiên cứu đa ngành đóng vai trò là nguồn thông tin sinh học phân tử. Nó được thành lập vào năm 1988 như một sự bổ sung cho các hoạt động của Viện Y tế Quốc gia (NIH) và Thư viện Y khoa Quốc gia (NLM). Cơ sở của nó được đặt tại Bethesda, Maryland, Hoa Kỳ [3]. Ban đầu, việc tạo ra NCBI nhằm hỗ trợ tìm hiểu các cơ chế phân tử ảnh hưởng đến sức khỏe và bệnh tật của con người với các mục tiêu sau: tạo và duy trì cơ sở dữ liệu công cộng, phát triển phần mềm để phân tích dữ liệu bộ gen và tiến hành nghiên cứu về sinh học tính toán. NCBI cũng bắt đầu cung cấp các dịch vụ:

Phát triển các phương pháp mới để đối phó với khối lượng và độ phức tạp của dữ liệu, nghiên cứu các phương pháp có thể phân tích cấu trúc và chức năng của các đại phân tử [3].

Tạo ra các hệ thống máy tính để lưu trữ và phân tích dữ liệu về sinh học phân tử. Cung cấp quyền truy cập vào các công cụ phân tích và tính toán (tạo điều kiện thuận lợi cho việc sử dụng cơ sở dữ liệu và phần mềm) cho các nhà nghiên cứu và công chúng [3].

Trong quá trình phát triển cơ sở dữ liệu, NCBI đã hình thành các tiêu chuẩn cơ sở dữ liệu như danh pháp cơ sở dữ liệu cũng được sử dụng bởi các cơ sở dữ liệu không phải NCBI khác. Một cơ sở dữ liệu NCBI là GenBank, cơ sở dữ liệu trình tự axit nucleic chứa thông tin trình tự từ hơn 200.000 sinh vật khác nhau. GenBank có lẽ là cơ sở dữ liệu phổ biến nhất được sử dụng và thực sự có trước NCBI. Đối với nhiều người, tên của nó đồng nghĩa với NCBI [18].

*Genbank làm cơ sở dữ liệu mô hình:* một trong những vai trò của NCBI là duy trì cơ sở dữ liệu có sẵn công khai [18]. Nhưng chính xác thì cơ sở dữ liệu là gì, và tại sao chúng lại quan trọng đối với sinh học phân tử? Về cơ bản, cơ sở dữ liệu là một khối dữ liệu lớn và có tổ chức [5]. Nhưng một trong những tiêu chí chính cho cơ sở dữ liệu sinh học là dữ liệu liên tục. Nói cách khác, thông tin được mã hóa và thể hiện bởi dữ liệu có thể thay đổi nhưng loại dữ liệu có khả năng thay đổi tốt hơn. Sự thiếu linh hoạt này của dữ liệu là sự phản ánh những gì bao gồm các đại phân tử và cách các nhà khoa học đã chọn để tượng trưng cho thiên nhiên.

Tìm kiếm một trình tự trong GenBank được gọi là “thực hiện truy vấn”. Thông tin xuất hiện được gọi là “bản ghi” (mục nhập) cho truy vấn. Hồ sơ cho mỗi trình tự trong GenBank chứa một mô tả ngắn gọn về trình tự, tên khoa học và phân loại của sinh vật nguồn mà từ đó trình tự được bắt nguồn, tài liệu tham khảo thư mục và danh sách các “đặc điểm”. Các tính năng bao gồm các vùng trình tự mã hóa của axit nucleic và các vị trí khác có tầm quan trọng sinh học. Ngoài ra, các trình tự protein của các vùng mã hóa axit nucleic được dịch được bao gồm. Mỗi bản ghi GenBank được gán một “số truy cập”, là một mã định danh ổn định và duy nhất của bản ghi không thay đổi theo thời gian [18].

Hệ thống truy xuất cơ sở dữ liệu chính tại NCBI là Entrez, liên kết một số cơ sở dữ liệu với nhau bao gồm GenBank. Cơ sở dữ liệu trung tâm trong Entrez là cơ sở dữ liệu nucleotide Genbank, liên kết đến các cơ sở dữ liệu sau: PubMed, Trình tự protein, Bộ gen, Phân loại, Cấu trúc, Dân số, Di truyền Mendel trực tuyến ở người (OMIM) [18].

* + 1. ***Cấu trúc chuỗi peptide***

Chuỗi peptide kháng khuẩn (AMPs) được cấu tạo từ các amino acid, là đơn vị cơ bản quyết định tính chất hóa học và hoạt tính kháng khuẩn. Chúng thuộc nhóm peptide, gồm các chuỗi ngắn từ 10–50 amino acid liên kết với nhau qua liên kết peptide. AMPs mang điện tích dương và có tính phân cực, cho phép chúng tương tác mạnh với màng tế bào vi sinh vật, vốn mang điện tích âm. Các peptide này thường có cấu trúc phân cực, với một đầu kỵ nước và một đầu ưa nước, giúp chúng dễ dàng xuyên qua lớp màng lipid kép của vi khuẩn, như defensin và cathelicidin, chúng tiêu diệt vi khuẩn bằng cách phá vỡ màng tế bào, làm giảm khả năng vi khuẩn phát triển kháng thuốc. AMPs có thể hình thành từ sự cắt nhỏ của các protein lớn hơn [14].

AMPs được tìm thấy trong nhiều sinh vật khác nhau, bao gồm động vật, thực vật và vi sinh vật, chúng đóng vai trò quan trọng trong hệ miễn dịch tự nhiên. AMPs là các chuỗi peptide ngắn, được mã hóa bởi các gen trong DNA và trải qua quá trình phiên mã (mRNA) và dịch mã (ribosome) để hình thành. Trong quá trình dịch mã, các amino acid được sắp xếp theo trình tự quy định bởi mRNA, tạo nên chuỗi polypeptide [14].

A diagram of a dna sequence

AI-generated content may be incorrect.

Hình 1. 1. Khung đọc mở ORF

Quá trình tổng hợp peptide kháng khuẩn (AMPs) bắt đầu từ DNA trong nhân tế bào, nơi thông tin di truyền được sao chép thành mRNA. mRNA này chứa vùng khung đọc mở (ORF), nơi mã hóa chuỗi peptide [17]. Khung đọc mở bắt đầu từ codon khởi đầu (AUG) và kết thúc tại một trong các codon kết thúc (UAA, UAG hoặc UGA). Sau khi quá trình dịch mã hoàn tất, mRNA được chuyển thành chuỗi peptide, trong đó các amino acid được sắp xếp theo một trình tự cụ thể, tạo thành AMPs hoàn chỉnh [14].

Quá trình tổng hợp peptide bắt đầu từ DNA, nơi lưu trữ thông tin di truyền của sinh vật. Đầu tiên, DNA được sử dụng làm khuôn mẫu để phiên mã thành mRNA. mRNA này chứa bản sao mã hóa thông tin di truyền và đóng vai trò như một bản hướng dẫn, truyền tải thông tin cần thiết cho việc tổng hợp peptide. Sau đó, mRNA di chuyển ra ngoài nhân tế bào, nơi nó được dịch mã thành chuỗi peptide qua sự tham gia của ribosome và các tRNA mang amino acid tương ứng [17].

Phần lớn các peptide kháng khuẩn (AMPs) tự nhiên được phát hiện thông qua phương pháp phân lập từ nguồn sinh học và đặc trưng hóa bằng các kỹ thuật truyền thống, tuy nhiên, các phương pháp này thường tốn nhiều thời gian và công sức. Để khắc phục những hạn chế này, phương pháp tính toán dựa trên dữ liệu peptide và bộ gen của các sinh vật đã được áp dụng [17]. Phương pháp này có thể chia thành năm nhóm chính: Nhóm một, dựa trên peptide trưởng thành (Mature peptide), tập trung vào các đặc điểm cấu trúc như tính axit, phân cực và khả năng tương tác với màng tế bào để xác định các peptide kháng khuẩn. Nhóm hai, dựa trên propeptide, phân tích các peptide nguyên mẫu chưa qua xử lý, dự đoán các peptide có khả năng trở thành AMPs sau khi trải qua quá trình cắt và biến đổi dịch mã. Nhóm ba, kết hợp peptide trưởng thành và propeptide, giúp tăng độ chính xác trong dự đoán AMPs bằng cách tận dụng đặc điểm của cả hai giai đoạn trong quá trình sinh tổng hợp peptide. Nhóm bốn, dựa trên enzyme xử lý (Processing Enzyme), phân tích các enzyme chịu trách nhiệm cắt và xử lý propeptide thành AMPs, hỗ trợ dự đoán các peptide kháng khuẩn tiềm năng. Nhóm 5, dựa trên ngữ cảnh gen (Genomic Context), khai thác thông tin bộ gen của sinh vật để xác định các gen mã hóa AMPs và làm rõ các yếu tố môi trường và sinh lý có thể kích hoạt quá trình sinh tổng hợp AMPs [17].

* + 1. ***Cloud Amazon Web Service (AWS)***

Năm 2006, Amazon Web Services (AWS) bắt đầu cung cấp dịch vụ cơ sở hạ tầng công nghệ thông tin cho các doanh nghiệp dưới dạng dịch vụ web, hiện nay thường được gọi là điện toán đám mây. Một trong những lợi ích chính của điện toán đám mây là cơ hội thay thế chi phí cơ sở hạ tầng vốn ban đầu bằng chi phí biến đổi thấp có thể mở rộng theo quy mô doanh nghiệp của bạn. Với đám mây, các doanh nghiệp không còn cần phải lập kế hoạch và mua máy chủ và cơ sở hạ tầng công nghệ thông tin khác trước nhiều tuần hoặc nhiều tháng. Thay vào đó, họ có thể ngay lập tức khởi động hàng trăm hoặc hàng nghìn máy chủ trong vài phút và cung cấp kết quả nhanh hơn. Ngày nay, AWS cung cấp nền tảng cơ sở hạ tầng có độ tin cậy cao, có thể mở rộng và chi phí thấp trên đám mây, hỗ trợ hàng trăm nghìn doanh nghiệp tại 190 quốc gia trên toàn thế giới [15].

Điện toán đám mây cung cấp một cách đơn giản để truy cập máy chủ, lưu trữ, cơ sở dữ liệu và một bộ dịch vụ ứng dụng rộng lớn qua Internet. Các nhà cung cấp điện toán đám mây như AWS sở hữu và duy trì phần cứng được kết nối mạng cần thiết cho các dịch vụ ứng dụng này [15].

* + - 1. Tổng quan về Cloud EC2

Năm 2006, Amazon đã ra mắt AWS EC2, một máy ảo dựa trên đám mây hiện đã thay đổi đáng kể ngành công nghiệp. AWS EC2 là viết tắt của Amazon Elastic Compute Cloud, một dịch vụ web của Amazon cung cấp các dịch vụ dựa trên đám mây tính toán có thể tùy chỉnh cao. Nó cũng có thể thay đổi kích thước vì bạn có thể tăng hoặc giảm đáng kể số lượng phiên bản máy chủ triển khai nếu nhu cầu tính toán thay đổi. Trên EC2 của Amazon, một phiên bản là một máy chủ ảo đang được sử dụng để chạy các ứng dụng. Nó cũng có thể được coi là một loại khía cạnh của một máy tính lớn, với ổ cứng, kết nối mạng, hệ điều hành riêng,.. Tuy nhiên, tất cả đều là một môi trường ảo. Ngay cả trên một máy vật lý duy nhất, bạn có thể có nhiều máy tính “nhỏ”, sau đây được gọi là Phiên bản [15].

* + - 1. Tính năng của EC2

EC2 là một nền tảng cho phép bạn triển khai máy chủ của mình trên đám mây. Trong AWS, máy chủ thường được gọi là phiên bản. Vì chúng được ảo hóa, bạn có thể truy cập phiên bản của mình từ hầu như mọi nơi và mọi lúc [15].

Một phiên bản được tạo từ AMI hoặc Amazon Machine Image, là một mẫu được cấu hình sẵn của hệ điều hành và ứng dụng. Điều này loại bỏ nhu cầu phải trải qua tất cả các bước cấu hình triển khai ứng dụng sau hệ điều hành, cũng như thời gian phân phối chung cho các phiên bản [15].

EC2 cung cấp nhiều phiên bản với các cấu hình CPU, bộ nhớ và mạng khác nhau. Điều này cho phép người dùng chọn phiên bản tốt nhất dựa trên kiến ​​trúc CPU, bộ nhớ và mạng của bạn [15].

Về bảo mật của phiên bản, EC2 cung cấp một Cặp khóa. Nếu bạn không có, bạn phải tạo một cặp khóa khi tạo phiên bản. AWS giữ lại khóa công khai và cung cấp cho bạn khóa riêng. Không ai có thể truy cập phiên bản của bạn trừ khi bạn cung cấp khóa riêng [15].

AWS phân bổ một địa chỉ IP công cộng tùy chọn và một địa chỉ IP riêng cho phiên bản EC2 của bạn sau khi bạn bắt đầu tạo. Mặt khác, địa chỉ IP công cộng là động và có thể dao động nếu phiên bản của bạn bị chấm dứt hoặc khởi động lại. Địa chỉ IP đàn hồi có thể được sử dụng để có được địa chỉ IP công cộng tĩnh [15].

Đám mây AWS trải dài trên nhiều châu lục và quốc gia. Bạn có thể chọn vị trí của phiên bản của mình bằng cách sử dụng vùng và vùng khả dụng [15].

### ***Nghiên cứu kiến trúc mô hình Transformer***

* + - 1. Giới thiệu về Transformer

Mô hình Transformer được giới thiệu lần đầu trong bài báo nghiên cứu *”Attention Is All You Need”* (2017) của nhóm tác giả Vaswani et al. Mục tiêu chính của Transformer là loại bỏ sự phụ thuộc vào các kiến trúc tuần tự như RNN (Recurrent Neural Network) hoặc CNN (Convolutional Neural Network) bằng cách sử dụng cơ chế *Self-Attention* (tự chú ý). Cơ chế này cho phép mô hình xử lý các phụ thuộc dài hạn trong dữ liệu hiệu quả hơn, đồng thời tận dụng khả năng tính toán song song để tăng tốc độ huấn luyện. Transformer đã trở thành nền tảng cho nhiều ứng dụng trong xử lý ngôn ngữ tự nhiên (NLP) như dịch máy, sinh văn bản, và gần đây được mở rộng sang lĩnh vực thị giác máy tính (Vision Transformers) và xử lý đa phương tiện.

* + - 1. Kiến trúc Transformer
         1. *Encoder và Decoder*

Encoder: Gồm một chồng các lớp giống nhau (thường từ 6–12 lớp). Mỗi lớp encoder có hai thành phần chính: Multi-Head, Self-Attention và Feed-Forward Network (FFN). Encoder nhận đầu vào là chuỗi các token (ví dụ: từ trong câu) và tạo ra biểu diễn ngữ cảnh đa tầng, mã hóa thông tin về mối quan hệ giữa các token.

Decoder: Cũng gồm nhiều lớp tương tự encoder, nhưng thêm một lớp Masked Multi-Head Attention để đảm bảo quá trình dự đoán token tiếp theo chỉ phụ thuộc vào các token đã sinh trước đó. Decoder sử dụng đầu ra từ encoder để sinh chuỗi kết quả (ví dụ: bản dịch).

Cơ chế Self-Attention cho phép mỗi token trong chuỗi đầu vào “chú ý” đến tất cả các token khác để xác định mức độ quan trọng của chúng.

Multi-Head Attention: mở rộng cơ chế này bằng cách thực hiện Self-Attention song song trên h ”đầu” (heads), sau đó ghép nối kết quả và chiếu xuống không gian ban đầu

Positional Encoding: do Transformer không có khả năng mặc định để nhận biết thứ tự các token, *Positional Encoding* được thêm vào embedding đầu vào để mã hóa thông tin vị trí.

Feed-Forward Network (FFN): mỗi lớp FFN gồm hai phép biến đổi tuyến tính với hàm kích hoạt ReLU hoặc GELU ở giữa

Layer Normalization và Residual Connections: để ổn định quá trình huấn luyện, đầu ra của mỗi lớp Self-Attention và FFN được chuẩn hóa bằng *Layer Normalization* và kết hợp với đầu vào gốc qua kết nối tắt (residual connection).

* + - 1. Ưu điểm của Transformer
* Xử lý song song: Khác với RNN, Transformer không cần xử lý tuần tự từng token, cho phép tận dụng tối đa phần cứng như GPU/TPU.
* Hiệu quả với phụ thuộc dài hạn: Self-Attention tính toán trực tiếp mối quan hệ giữa các token cách xa nhau, tránh được vấn đề tiêu biến gradient trong RNN.
* Linh hoạt: Kiến trúc có thể áp dụng cho nhiều loại dữ liệu, từ văn bản (BERT, GPT) đến ảnh (Vision Transformers) và âm thanh.
* LayerNorm: Chuẩn hóa đầu ra của từng lớp con (Self-Attention/FFN) theo từng mẫu, giúp ổn định phân phối giá trị và tăng tốc độ hội tụ.
* Residual Connection: Kết nối tắt (bỏ qua một hoặc nhiều lớp) giữa đầu vào và đầu ra, đảm bảo thông tin gốc không bị mất qua các lớp sâu.

### ***Nghiên cứu Explainable Artificial Intelligence***

* + - 1. Giới thiệu về Explainable Artificial Intelligence
         1. *Vấn đề hộp đen trong học máy*

Sự đổi mới trong các thuật toán học máy đã dẫn đến những tiến bộ lớn về sức mạnh và độ chính xác của dự đoán. Tuy nhiên, chúng ngày càng trở nên phức tạp hơn. Đây là sự đánh đổi đáng tiếc giữa chất lượng được cải thiện và tính minh bạch. Chúng ta có thể quan sát tập hợp các đầu ra cho một tập hợp các đầu vào nhất định cho một mô hình, mà không cần biết hoặc hiểu về hoạt động bên trong của nó. Không giống như các mô hình toán học có cấu trúc vốn có, các mô hình học máy có thể học cách ánh xạ các đầu vào thành đầu ra trực tiếp từ dữ liệu. Đối với một số mô hình như cây quyết định, việc ánh xạ này dễ dàng nhận thấy. Đối với các mô hình khác như rừng ngẫu nhiên hoặc mô hình học sâu, việc hiểu cách đưa ra dự đoán trở nên gần như không thể. Nhiều mô hình học máy và học sâu về cơ bản là “hộp đen” không tiết lộ các cơ chế và sắc thái bên trong đối với các dự đoán của chúng [13].

Sự thiếu minh bạch và hiểu biết này có thể gây ra hậu quả nghiêm trọng đối với lòng tin và việc áp dụng các mô hình này của chúng ta. Ví dụ, làm sao chúng ta biết được liệu các dự đoán của mô hình có thể sai không? Điều này đặc biệt quan trọng trong các lĩnh vực có rủi ro cao như chăm sóc sức khỏe. Liệu bác sĩ hoặc bệnh nhân có tin tưởng vào dự đoán ung thư nếu một mô hình được đào tạo có độ chính xác là 99 phần trăm không? Nếu chúng ta không biết, mô hình bỏ lỡ các trường hợp ác tính nhất thì sao? Nếu độ chính xác cao là do rò rỉ dữ liệu trong dữ liệu thử nghiệm, khiến hiệu suất ngoài mẫu kém hơn nhiều thì sao? Đây là lý do tại sao XAI được là yếu tố quan trọng đối với việc chúng ta áp dụng máy học. Đối với các quyết định có rủi ro cao như các khoản vay tín dụng, các đơn xin tại ngoại và ân xá có phân biệt đối xử, chẩn đoán y tế, v.v., thì việc các mô hình máy học có thể giải thích được trở nên bắt buộc [13].

* + - * 1. *Tầm quan trọng của Explainable Artificial Intelligence*

XAI (XAI) tìm cách cung cấp cho chúng ta cái nhìn sâu sắc về khả năng ra quyết định của một hệ thống AI. Nó giúp chúng ta hiểu cách thức, thời điểm và lý do tại sao các dự đoán được đưa ra. Do đó, nó có thể xây dựng lòng tin lớn hơn và cải thiện tính an toàn khi chúng ta sử dụng các mô hình AI, khuyến khích việc áp dụng chúng nhiều hơn trong xã hội của chúng ta. Chúng tôi bắt đầu khám phá XAI bằng cách xác định một số mục tiêu có liên quan với nhau: khả năng hiểu, khả năng hiểu được, khả năng diễn giải và tính minh bạch. Mỗi khái niệm này đều gắn chặt với độ phức tạp của mô hình. Mặc dù nhiều khái niệm trong số này có thể khác nhau hoặc chồng chéo giữa các miền khác nhau, nhưng chúng khác biệt về kết quả mong muốn, đặc điểm và/hoặc cách tiếp cận [13]:

* Khả năng hiểu được là việc chức năng của mô hình AI có thể được con người nắm bắt mà không cần hiểu chi tiết thuật toán bên trong. Ví dụ, bộ mã hóa tự động có thể được hiểu ở mức khái niệm ngay cả khi không rõ cách nó nén và giải nén dữ liệu.
* Khả năng hiểu trong mô hình học máy đề cập đến mức độ con người có thể hiểu kiến thức mà mô hình đã học. Dù khó đo lường tuyệt đối, ta có thể so sánh tương đối — ví dụ, kết quả từ phân tích nhân tố thường dễ hiểu hơn PCA.
* Khả năng diễn giải là khả năng mô tả cấu trúc và dự đoán của mô hình theo cách dễ hiểu. Đây là khái niệm chủ quan, phụ thuộc vào bối cảnh và đối tượng người dùng.
* Tính minh bạch phản ánh mức độ dễ hiểu của cấu trúc và thuật toán mô hình. Nó giúp giải thích tại sao mô hình hoạt động như vậy, với mỗi mô hình có thể có mức độ minh bạch khác nhau.

Lĩnh vực mô hình hóa học máy đã phát triển nhanh chóng trong thế kỷ qua. Nhiều mô hình tính toán đã được tạo ra để mô hình hóa các quá trình nhận thức và sinh học trong đời thực, và sự ra đời của máy tính đã khởi đầu cho một sự bùng nổ các thuật toán mới trước đây bị hạn chế bởi sức mạnh tính toán [13].

* + - 1. Mục đích của AI có thể giải thích

XAI trình bày một số vấn đề quan trọng bao gồm các cân nhắc thực tế, đạo đức, triết học và công bằng. Các phương pháp XAI có thể giải quyết và giảm thiểu các vấn đề này theo nhiều cách và sự thành công của các ứng dụng AI có thể sẽ được thúc đẩy bởi các phương pháp XAI trong tương lai [13]:

* Tính thông tin: XAI giúp khám phá mối quan hệ trong mô hình AI, từ đó xác định liệu mục tiêu của mô hình có phù hợp với mục đích ra quyết định hay không.
* Độ tin cậy: Mức độ tin tưởng vào AI phụ thuộc vào khả năng giải thích của nó. Sự minh bạch và dễ hiểu tăng cường lòng tin, dù lòng tin và độ tin cậy là hai khái niệm khác nhau.
* Công bằng: XAI hỗ trợ phát hiện và giải quyết các vấn đề thiên vị trong dữ liệu hoặc thuật toán, đảm bảo hành vi của AI công bằng và không phân biệt đối xử.
* Tính minh bạch: Giúp hiểu cách AI đưa ra quyết định, nhưng không đảm bảo sự công bằng hoặc khả năng giải thích. Ví dụ, một mô hình minh bạch có thể vẫn thiên vị nếu dữ liệu không đa dạng.
* Tính nhân quả: XAI giúp khám phá mối quan hệ nhân – quả trong dữ liệu, điều mà các mô hình học máy truyền thống thường bỏ qua do chỉ tập trung vào tương quan.
* Khả năng chuyển giao: XAI giúp hiểu cách mô hình học và hoạt động, từ đó hỗ trợ áp dụng mô hình đã học vào các nhiệm vụ khác (học chuyển giao).
* Độ ổn định & mạnh mẽ: XAI cho phép đánh giá mức độ mô hình duy trì quyết định nhất quán và đáng tin cậy trong các tình huống tương tự.
* Khả năng tiếp cận: XAI giúp người không chuyên dễ hiểu và tiếp cận AI hơn, từ đó thúc đẩy việc ứng dụng rộng rãi hơn.
* Quyền riêng tư: XAI hỗ trợ đánh giá liệu các thuật toán có vi phạm quyền riêng tư không mà không cần truy cập vào dữ liệu gốc.
  + - 1. **SHAP (SHapley Additive exPlanations)**
         1. *Tầm quan trọng của SHAP*

Nền tảng lý thuyết của SHAP kết hợp các khái niệm từ lý thuyết trò chơi và học máy để phát triển một cách tiếp cận có nguyên tắc để giải thích mô hình. Cốt lõi của nó là giá trị Shapley từ lý thuyết trò chơi hợp tác, cung cấp một phương pháp nghiêm ngặt về mặt toán học để quy kết giá trị cho đóng góp của mỗi người chơi trong một liên minh. Trong bối cảnh học máy, các tính năng được coi là người chơi trong một trò chơi, với dự đoán của mô hình đại diện cho tổng giá trị của liên minh. Nhằm lượng hóa đóng góp của từng đặc trưng đối với kết quả dự đoán. Phương pháp SHAP thể hiện tầm quan trọng đặc biệt qua các đặc điểm sau [7]:

* Tính nhất quán về mặt toán học: Đảm bảo phân phối công bằng đóng góp của các đặc trưng, tuân thủ nghiêm ngặt các tiên đề toán học cơ bản.
* Khả năng áp dụng linh hoạt: Có thể triển khai cho cả mô hình “hộp đen” (như mạng nơ-ron sâu) lẫn mô hình “hộp trắng” (như cây quyết định).
* Khả năng giải thích đa cấp độ: Cung cấp cả giải thích chi tiết cho từng dự đoán cụ thể và phân tích tổng thể về ảnh hưởng của các đặc trưng trên toàn bộ tập dữ liệu.
* Định lượng chính xác hướng ảnh hưởng: Xác định rõ đặc trưng nào làm tăng hoặc giảm giá trị dự đoán, hỗ trợ đắc lực cho công tác thiết kế phân tử hoặc chẩn đoán trong lĩnh vực y sinh.
* Ứng dụng hiệu quả trong nghiên cứu y sinh: Giúp xác định chính xác các amino acid hoặc cấu trúc không gian quyết định đến hoạt tính sinh học, độc tính, hoặc độ bền của peptide

Nền tảng toán học:Các giá trị SHAP được tính toán bằng cách sử dụng một khuôn khổ đáp ứng một số thuộc tính chính [4]:

* Tính cộng: Đóng góp của tất cả các tính năng phải tổng cộng với sự khác biệt giữa dự đoán của mô hình cho một đầu vào cụ thể và dự đoán dự kiến trên tất cả các đầu vào. Thuộc tính này đảm bảo rằng lời giải thích vừa đầy đủ vừa phù hợp với hành vi tổng thể của mô hình.
* Tuyến tính: Đóng góp của một tính năng phải chia tỷ lệ tuyến tính với giá trị của nó. Thuộc tính này đảm bảo rằng lời giải thích tỷ lệ thuận với tác động của đối tượng đối với dự đoán.
* Tính trung lập đối với trật tự: Lời giải thích phải nhất quán bất kể thứ tự mà các đặc điểm được xem xét. Thuộc tính này rất quan trọng để đảm bảo rằng cách giải thích là đáng tin cậy và không thiên vị.
* Thiếu tích: Các đối tượng không có (tức là có giá trị bằng không) sẽ không ảnh hưởng đến dự đoán. Thuộc tính này đảm bảo rằng giải thích chỉ tính đến các tính năng có trong dữ liệu đầu vào.

Thuộc tính của giá trị SHAP: Các giá trị SHAP đáp ứng một số thuộc tính chính đảm bảo hiệu quả của chúng như các đóng góp tính năng có thể diễn giải [4]:

* Hiệu quả (Công bằng): Tổng các giá trị SHAP cho tất cả các tính năng trong liên minh bằng sự khác biệt giữa dự đoán của mô hình cho trường hợp đó và dự đoán trung bình trên tất cả các trường hợp.
* Đối xứng: Các tính năng có đóng góp giống hệt nhau nhận được cùng giá trị SHAP.
* Tuyến tính: Giá trị SHAP của một tính năng chia tỷ lệ tuyến tính với giá trị của nó.
* Độ chính xác cục bộ: Giá trị SHAP thể hiện chính xác sự đóng góp của đối tượng vào dự đoán cụ thể.

Các thuộc tính này chung đảm bảo rằng các giá trị SHAP cung cấp phân bổ công bằng và chính xác về các đóng góp của tính năng vào dự đoán mô hình, làm cho chúng trở thành một công cụ mạnh mẽ để hiểu các mô hình máy học phức tạp.

* + - * 1. *So sánh SHAP với các phương pháp giải thích khác*

|  |  |  |
| --- | --- | --- |
| Phương pháp | Ưu điểm | Nhược điểm |
| SHAP | Nhất quán về mặt toán học. | Tính toán chậm với dữ liệu lớn. |
| Giải thích cục bộ + toàn cục. | Phức tạp khi có nhiều tương tác đặc trưng. |
| Hỗ trợ đa dạng mô hình. |  |
| LIME | Đơn giản, dễ triển khai. | Không đảm bảo tính nhất quán. |
| Tập trung vào giải thích cục bộ. | Nhạy cảm với nhiễu dữ liệu. |
| Feature Importance | Cung cấp cái nhìn toàn cục. | Không giải thích được từng dự đoán riêng lẻ. |
| Dễ hiểu với mô hình dựa trên cây. | Bỏ qua tương tác giữa các đặc trưng. |
| Saliency Maps | Hiệu quả với mô hình học sâu (CNN, Transformer). | Chỉ tập trung vào đầu vào, không xét cấu trúc mô hình. |
| Trực quan hóa vùng ảnh hưởng trên ảnh/văn bản. | Dễ bị nhiễu khi gradient không ổn định. |

Bảng 1. 1. So sánh SHAP với các phương pháp giải thích khác

# **CHƯƠNG II: PHƯƠNG PHÁP THỰC HIỆN**

## **Yêu cầu kỹ thuật**

Để phát triển một hệ thống phân tích đặc trưng và thiết kế peptide hiệu quả dựa trên học sâu và Explainable AI, cần đáp ứng các yêu cầu kỹ thuật sau:

* + 1. ***Yêu cầu về dữ liệu***

Đầu vào: mô hình phải xử lý được các trình tự peptide ở dạng chuỗi amino acid với độ dài tối ưu từ 10-50 amino acid, đảm bảo tính nhất quán cho quá trình học.

* + 1. ***Yêu cầu về mô hình học sâu***

Độ chính xác: mô hình cần đạt độ chính xác cao trong việc dự đoán các đặc tính peptide quan trọng, với các chỉ số đánh giá như accuracy, precision, recall và F1-score đạt mức tối ưu.

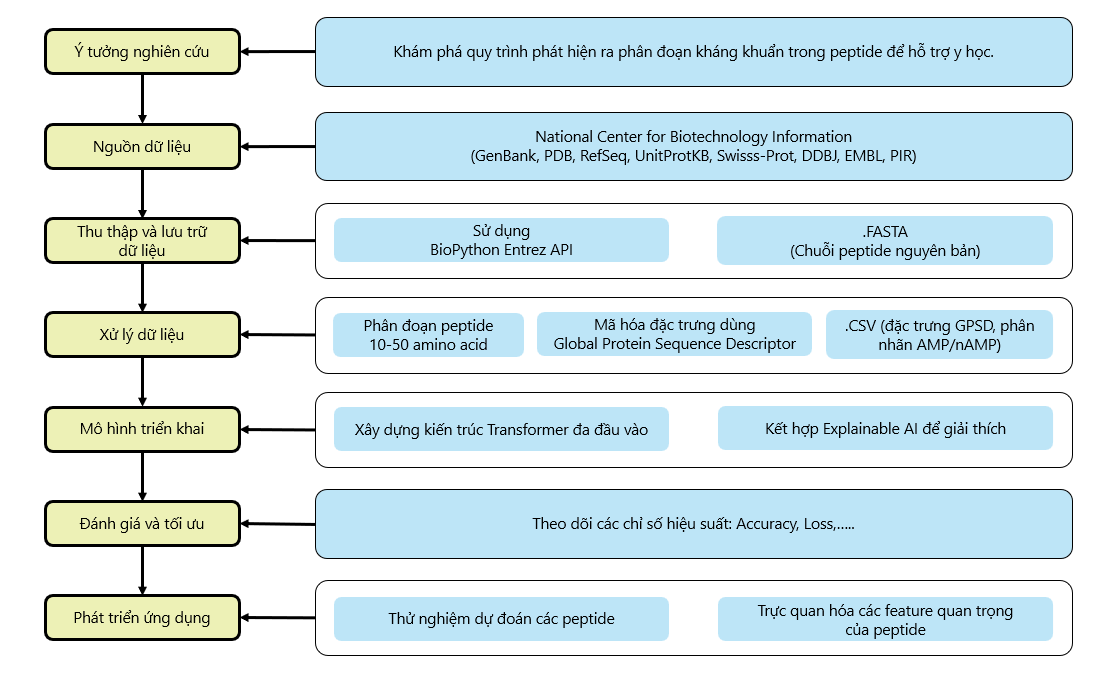
* + 1. ***Yêu cầu về Explainable AI (XAI)***

Có khả năng xác định và lượng hóa tầm quan trọng của từng amino acid trong trình tự đối với kết quả dự đoán thông qua phương pháp SHAP (SHapley Additive exPlanations).

Trực quan hóa được các attention weights từ mô hình Transformer để hiểu rõ cơ chế tập trung của mô hình vào các vị trí cụ thể trong chuỗi peptide.

Biểu diễn kết quả giải thích dưới dạng các biểu đồ trực quan như waterfall plot, summary plot, và heat map nhằm tối ưu hóa việc truyền đạt thông tin cho người dùng.

## **Sơ đồ phương pháp thực hiện**



Sơ đồ 2. 1. Sơ đồ phương pháp thực hiện thiết kế tổng quan hệ thống

A diagram of a computer

AI-generated content may be incorrect.

Sơ đồ 2. 2. Luồng thực hiện của mô hình

* + 1. ***Ý tưởng nghiên cứu***

Kháng kháng sinh hiện đang là một trong những thách thức lớn nhất trong y học hiện đại. Peptide kháng khuẩn (Antimicrobial Peptides – AMP) được xem như một giải pháp tiềm năng do khả năng tiêu diệt hoặc ức chế sự phát triển của vi khuẩn, virus và nấm. Đề tài này nhằm xây dựng một hệ thống hỗ trợ phát hiện và phân tích các peptide có khả năng kháng khuẩn từ kho dữ liệu sinh học công khai, ứng dụng học sâu để hỗ trợ phát triển thuốc mới.

* + 1. ***Nguồn dữ liệu***

Hệ thống được xây dựng dựa trên các nguồn dữ liệu uy tín từ National Center for Biotechnology Information (NCBI) cùng các cơ sở dữ liệu liên kết như GenBank, PDB, RefSeq, UniProtKB, Swiss-Prot, DDBJ, EMBL và PIR. Những nguồn này được chọn lựa kỹ lưỡng nhờ độ tin cậy cao và khả năng cung cấp thông tin chi tiết về trình tự protein cần thiết cho việc khai thác peptide có tiềm năng kháng khuẩn.

Bảng mô tả dữ liệu:

|  |  |  |
| --- | --- | --- |
| Tên cột | Loại dữ liệu | Mô tả |
| id\_sequence | Định danh (định tính) | Mã định danh duy nhất cho mỗi trình tự peptide từ các cơ sở dữ liệu sinh học. |
| label | Phân loại (định tính) | Nhãn phân loại: AMP (peptide có hoạt tính kháng khuẩn) hoặc nAMP (không có hoạt tính). |
| sequence | Trình tự sinh học (định tính) | Trình tự amino acid (ký hiệu bằng mã chữ cái một ký tự). |

Bảng 2. 1. Mô tả dữ liệu

* + 1. ***Thu thập và lưu trữ dữ liệu***

Quá trình thu thập dữ liệu được thực hiện thông qua BioPython Entrez API, công cụ hiệu quả cho phép truy xuất thông tin từ NCBI một cách có hệ thống. Sau khi thu thập, dữ liệu peptide được lưu trữ dưới định dạng FASTA, đây là định dạng chuẩn trong sinh học tính toán dùng để biểu diễn chuỗi peptide nguyên bản một cách chính xác và thuận tiện cho các bước xử lý tiếp theo.

* + 1. ***Xử lý dữ liệu***

Dữ liệu thô sau khi thu thập được xử lý bằng cách phân đoạn thành các chuỗi peptide có độ dài từ 10-50 axit amin, phù hợp với khoảng độ dài điển hình của các peptide kháng khuẩn. Tiếp đó, mỗi chuỗi peptide được mã hóa thành các đặc trưng nhằm trích xuất các thuộc tính lý hóa quan trọng. Kết quả xử lý được lưu trữ dưới dạng file CSV, bao gồm các đặc trưng đã được trích xuất cùng với nhãn phân loại tương ứng.

* + 1. ***Mô hình triển khai***

Để nhận diện peptide kháng khuẩn một cách hiệu quả, đề tài sử dụng mô hình học sâu dựa trên kiến trúc Transformer, một kiến trúc hiện đại đã chứng minh được hiệu quả vượt trội trong xử lý dữ liệu dạng chuỗi. Bên cạnh đó, mô hình còn tích hợp công nghệ Explainable AI (XAI) giúp làm rõ quá trình ra quyết định của mô hình, từ đó xác định được những đặc trưng cốt lõi quyết định đến tính kháng khuẩn của peptide.

* + 1. ***Đánh giá và tối ưu***

Hiệu suất của mô hình được đánh giá toàn diện thông qua các chỉ số như Accuracy, Loss, cùng các thông số khác như Precision, Recall và F1-Score. Quá trình huấn luyện và tối ưu được diễn ra liên tục nhằm đảm bảo mô hình không chỉ đạt độ chính xác cao trên tập dữ liệu huấn luyện mà còn có khả năng tổng quát tốt khi áp dụng với các peptide mới chưa từng xuất hiện trong dữ liệu đào tạo.

* + 1. ***Phát triển mô hình***

Sau khi hoàn thiện, mô hình được ứng dụng vào việc thử nghiệm dự đoán các peptide tiềm năng có khả năng kháng khuẩn chưa được xác định trước đó. Đồng thời, việc trực quan hóa các đặc trưng quan trọng của peptide giúp các nhà nghiên cứu hiểu rõ hơn về cơ chế hoạt động và những yếu tố quyết định tính kháng khuẩn, mở ra hướng tiếp cận mới trong quá trình nghiên cứu và phát triển các chất kháng khuẩn thế hệ mới.

Mô hình được xây dựng trong đề tài này cung cấp một quy trình toàn diện từ thu thập, xử lý đến phân tích peptide kháng khuẩn. Với việc kết hợp công nghệ học sâu và AI có khả năng giải thích, mô hình không chỉ đạt hiệu quả cao trong việc dự đoán mà còn tăng tính minh bạch trong quá trình phân tích. Kết quả nghiên cứu có tiềm năng đóng góp đáng kể vào công cuộc tìm kiếm và phát triển các thuốc kháng khuẩn mới, góp phần giải quyết vấn đề kháng thuốc ngày càng nghiêm trọng trong y học hiện đại.

## **Lựa chọn công nghệ và phương án thiết kế**

* + 1. ***Công nghệ và phương án đề xuất***

TensorFlow: Framework này được chọn cho việc xây dựng mô hình học sâu vì khả năng mở rộng và hỗ trợ tối ưu cho các kiến trúc mạng phức tạp như Transformer. TensorFlow cung cấp công cụ toàn diện cho việc phát triển, huấn luyện và triển khai các mô hình học máy, cùng với hệ sinh thái rộng lớn các công cụ bổ sung như TensorBoard cho việc theo dõi và trực quan hóa quá trình huấn luyện.

Kiến trúc Transformer: Kiến trúc này được áp dụng cho việc phân tích trình tự peptide nhờ khả năng nắm bắt xuất sắc các mối quan hệ phụ thuộc xa trong chuỗi amino acid. Cơ chế Attention cho phép mô hình tập trung vào các phần quan trọng của trình tự, đồng thời cung cấp cơ sở cho việc giải thích kết quả thông qua việc phân tích các trọng số attention.

SHAP: Phương pháp này được chọn làm công cụ XAI chính vì khả năng giải thích các dự đoán của mô hình theo cách nhất quán về mặt toán học, dựa trên lý thuyết trò chơi và giá trị Shapley. SHAP cung cấp cách tiếp cận thống nhất để định lượng mức độ đóng góp của từng đặc trưng đầu vào đối với kết quả dự đoán, giúp xác định chính xác vai trò của các amino acid trong trình tự peptide.

AWS EC2: Nền tảng điện toán đám mây này được sử dụng để cung cấp năng lực tính toán cao cần thiết cho việc huấn luyện các mô hình học sâu phức tạp, đặc biệt khi làm việc với bộ dữ liệu lớn. Các instance GPU chuyên dụng trên AWS EC2 giúp giảm đáng kể thời gian huấn luyện và cho phép thực hiện các thử nghiệm song song.

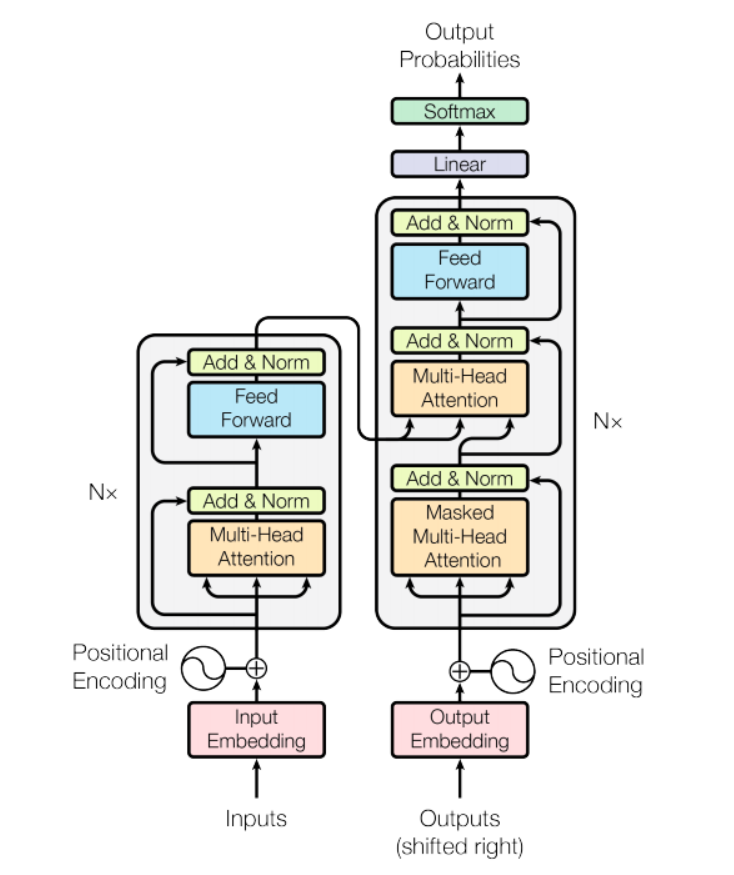
* + 1. ***Lý do lựa chọn phương án và công nghệ.***
       1. Lý do chọn Transformer
* Khả năng xử lý hiệu quả các dữ liệu chuỗi như trình tự peptide, vượt trội so với các mô hình truyền thống như RNN hay LSTM trong việc nắm bắt các đặc tính cấu trúc phức tạp.
* Cơ chế self-attention giúp mô hình nắm bắt được các mối tương tác xa trong chuỗi peptide, điều này đặc biệt quan trọng khi phân tích các đặc tính chức năng phụ thuộc vào sự tương tác giữa các amino acid không liền kề.
* Khả năng mở rộng và chuyển giao học tập từ các mô hình được huấn luyện trước trong lĩnh vực sinh học, tận dụng kiến thức từ các bộ dữ liệu lớn như UniRef và SwissProt.
* Kiến trúc song song hóa cao cho phép tăng tốc quá trình huấn luyện và suy luận khi được triển khai trên các phần cứng GPU hiện đại.
  + - 1. Lý do chọn SHAP cho XAI
* Có thể áp dụng cho bất kỳ mô hình ML nào, không chỉ giới hạn ở các kiến trúc cụ thể, tạo tính linh hoạt trong việc so sánh và đánh giá các phương pháp khác nhau.
* Cung cấp giải thích ở cả cấp độ toàn cục và cục bộ, cho phép phân tích tầm quan trọng của đặc trưng trên toàn bộ tập dữ liệu cũng như đối với từng dự đoán cụ thể.
* Cho phép trực quan hóa đóng góp của từng đặc trưng một cách rõ ràng thông qua các biểu đồ như waterfall plots và summary plots, tạo điều kiện thuận lợi cho việc truyền đạt kết quả đến các chuyên gia không có nền tảng kỹ thuật.
* Hỗ trợ phân tích tương tác giữa các đặc trưng, giúp phát hiện các mối quan hệ phức tạp giữa các vị trí amino acid trong trình tự peptide.
  + - 1. Lý do chọn AWS EC2
* Khả năng mở rộng linh hoạt theo yêu cầu tính toán, cho phép điều chỉnh tài nguyên dựa trên độ phức tạp của mô hình và kích thước dữ liệu.
* Hỗ trợ GPU chuyên dụng cho deep learning như NVIDIA Tesla và NVIDIA A100, tối ưu hóa hiệu suất cho các mô hình Transformer phức tạp.
* Chi phí hiệu quả với mô hình trả tiền theo giờ sử dụng, giúp tối ưu ngân sách nghiên cứu thông qua việc chỉ sử dụng tài nguyên khi cần thiết.
* Dễ dàng thiết lập và quản lý thông qua giao diện web, cùng với khả năng tự động hóa quy trình triển khai thông qua các công cụ như AWS CloudFormation và AWS CLI.
* Tích hợp liền mạch với các dịch vụ AWS khác như S3 cho lưu trữ dữ liệu và SageMaker cho việc quản lý quy trình học máy

## **Thách thức thuật toán**

* + 1. ***Mô tả thuật toán áp dụng***

Mã hóa trình tự peptide: Nghiên cứu áp dụng kỹ thuật mã hóa tokenizer và embedding để chuyển đổi trình tự amino acid thành các biểu diễn số học mà mô hình có thể xử lý hiệu quả. Quá trình này chuyển đổi chuỗi ký tự amino acid thành các vector đặc trưng có chiều cao, giữ lại thông tin về đặc tính lý hóa và cấu trúc của từng amino acid.

* + - 1. Mô hình Transformer



Sơ đồ 2. 3. Kiến trúc Transformer

Transformer tỏ ra vượt trội trong việc xử lý dữ liệu văn bản vốn có tính chất tuần tự. Nó lấy một chuỗi văn bản làm đầu vào và tạo ra một chuỗi văn bản khác. Ví dụ như bài toán Machine Translation hay Text Summarization.

Về thành phần cấu tạo, Transformers bao gồm một nhóm các bộ Encoders (*Encoder stack*) và một nhóm các bộ Decoder (*Decoder stack*). Ngoài ra còn có các thành phần Embedding, Encoding, Mask, … khác để xử lý dữ liệu đầu vào và đầu ra.

Transformer bao gồm 2 phần chính:

* Encoder (trái): Mã hóa thông tin từ đầu vào.
* Decoder (phải): Sinh đầu ra dựa trên thông tin mã hóa từ encoder và thông tin đã sinh ra trước đó.
  + - * 1. *Encoder*

Mỗi encoder gồm N lớp lặp lại (N thường là 6 trong Transformer gốc), mỗi lớp có 2 khối:

Multi-Head Attention: áp dụng cơ chế attention nhiều lần song song (multi-head). Đầu vào sẽ tự tương tác với chính nó để xác định mức độ chú ý (self-attention). Giúp mô hình học được các mối quan hệ giữa các từ bất kể khoảng cách.

Feed Forward Neural Network (FFNN): một mạng nơ-ron đơn giản gồm 2 lớp tuyến tính với hàm kích hoạt ReLU hoặc GELU ở giữa.

Add & Norm: sau mỗi khối (Attention hoặc FFNN), kết quả được cộng với đầu vào ban đầu (residual connection) và chuẩn hóa lớp (Layer Normalization).

Positional Encoding: vì mô hình không có kiến trúc tuần tự như RNN, nên cần thêm thông tin vị trí của từ thông qua vector Positional Encoding được cộng vào embedding.

* + - * 1. *Decoder*

Mỗi decoder cũng gồm N lớp lặp lại, nhưng có 3 khối thay vì 2:

*Masked Multi-Head Attention:* tương tự attention ở encoder, nhưng có mask để không cho mô hình nhìn trước các từ tương lai (dùng khi huấn luyện mô hình sinh văn bản).

*Multi-Head Attention (tương tác với encoder):* attention giữa đầu ra hiện tại với đầu ra từ encoder. Mục đích: giúp decoder hiểu ngữ cảnh từ đầu vào.

*Feed Forward + Add & Norm:* tương tự encoder.

*Output Embedding + Positional Encoding:* các đầu ra đã sinh ra trước đó được nhúng (embedding) và cộng thêm thông tin vị trí giống encoder.

* + - * 1. *Đầu ra*

Sau khi qua tất cả các lớp decoder, đầu ra được đưa qua một lớp tuyến tính và hàm Softmax để sinh xác suất của từng từ trong từ vựng.

* + - * 1. *Áp dụng thuật toán của mô hình vào đồ án, chỉ sử dụng encoder:*

Sử dụng Positional Encoding để mã hóa vị trí của từng amino acid trong chuỗi, giúp mô hình phân biệt được thứ tự xuất hiện của các amino acid mặc dù kiến trúc Transformer không có tính tuần tự vốn có.

Áp dụng cơ chế Self-Attention để nắm bắt các tương tác giữa các vị trí khác nhau trong trình tự, cho phép mô hình học được mối liên hệ giữa các amino acid ở vị trí xa nhau trong chuỗi peptide.

Sử dụng Multi-Head Attention để học nhiều biểu diễn khác nhau của cùng một chuỗi đầu vào, tăng khả năng mô hình nắm bắt các khía cạnh khác nhau của mối tương tác trong trình tự.

Kết hợp Layer Normalization và Residual Connections để cải thiện quá trình huấn luyện, giảm thiểu vấn đề gradient biến mất và tăng tốc độ hội tụ của mô hình.

* + - 1. Phân tích SHAP

Để giải thích mô hình chi tiết, SHAP cung cấp một số chức năng chính [4]:

* Tạo một đối tượng SHAP bằng cách sử dụng mô hình đã tải.
* Gọi phương thức explain với dữ liệu đầu vào của bộ kiểm tra để tạo ra các giá trị SHAPE.
* Trả về ma trận giá trị SHAP kết quả, đại diện cho các đóng góp của tính năng cho dự đoán mô hình.

Trực quan hóa đóng một vai trò quan trọng trong việc giải thích kết quả SHAPE:

* Sử dụng các hàm gốc của SHAP để tạo các loại biểu đồ khác nhau, bao gồm:
* Biểu đồ tóm tắt: Cung cấp tổng quan nhanh về tầm quan trọng của tính năng trên tất cả các phiên bản
* Biểu đồ phụ thuộc: Hiển thị mối quan hệ giữa một tính năng cụ thể và đầu ra của mô hình
* Biểu đồ thác nước: Chia nhỏ các dự đoán riêng lẻ để hiển thị tác động của đối tượng
* Biểu đồ bầy ong: Hiển thị giá trị SHAP cho nhiều phiên bản, hiển thị phân bố tính năng

Khả năng của SHAP vượt ra ngoài cách giải thích cơ bản, cung cấp thông tin chi tiết về các mô hình phức tạp trên các ứng dụng khác nhau [4]:

* Hoạt động với nhiều loại mô hình, bao gồm hồi quy tuyến tính, cây quyết định, rừng ngẫu nhiên, mô hình tăng cường gradient và mạng nơ-ron
* Duy trì đo lường tầm quan trọng tính năng nhất quán trên các kiến trúc mô hình khác nhau
* Cung cấp cả giải thích cục bộ và toàn cầu, cho phép kiểm tra chi tiết các dự đoán riêng lẻ và hành vi tổng thể của mô hình
* Hỗ trợ dữ liệu chiều cao thông qua các phương pháp tính toán hiệu quả như KernelSHAP và TreeSHAP
* Cho phép đánh giá tính công bằng của mô hình bằng cách kiểm tra các đóng góp của tính năng trên các kết quả dự đoán khác nhau

Tính toán giá trị SHAP cho từng amino acid trong trình tự, định lượng mức độ ảnh hưởng của mỗi vị trí đến kết quả dự đoán cuối cùng:

* Xác định mức đóng góp của từng vị trí đến dự đoán cuối cùng, phân biệt giữa các amino acid có tác động tích cực và tiêu cực đến chức năng được dự đoán.
* Trực quan hóa kết quả thông qua các biểu đồ waterfall và summary, cung cấp cái nhìn tổng quan về tầm quan trọng tương đối của từng amino acid và các mẫu chung trong tập dữ liệu.

Thuật toán tối ưu hóa cho thiết kế peptide:

* Sử dụng tìm kiếm di truyền hoặc tối ưu hóa Bayesian để khám phá không gian trình tự rộng lớn một cách hiệu quả, tìm kiếm các trình tự tối ưu theo tiêu chí xác định.
* Điều hướng tìm kiếm dựa trên dự đoán từ mô hình học sâu, sử dụng phản hồi từ mô hình đã được huấn luyện để hướng quá trình tối ưu hóa tới các vùng có khả năng cao.
* Đánh giá trình tự được tạo ra dựa trên các tiêu chí được định nghĩa trước, bao gồm các đặc tính cụ thể và độ tin cậy của dự đoán.
  + 1. ***Khó khăn và giải pháp thay thế***
       1. Vấn đề dữ liệu hạn chế

Khó khăn: Dữ liệu peptide có chức năng được chú thích đầy đủ còn hạn chế, gây khó khăn cho việc huấn luyện các mô hình học sâu đòi hỏi lượng dữ liệu lớn để hoạt động hiệu quả.

Giải pháp: Áp dụng kỹ thuật data augmentation như biến đổi bảo toàn cấu trúc và thay thế amino acid tương đồng, kết hợp với transfer learning từ các mô hình protein lớn hơn đã được huấn luyện trước trên các tập dữ liệu quy mô lớn như UniRef và SwissProt.

* + - 1. Độ phức tạp tính toán của SHAP:

Khó khăn: Tính toán giá trị SHAP đòi hỏi nhiều tài nguyên tính toán, đặc biệt với mô hình lớn, có thể dẫn đến thời gian xử lý dài và yêu cầu phần cứng cao cấp.

Giải pháp: Sử dụng KernelSHAP với phương pháp lấy mẫu thông minh thay vì tính toán chính xác toàn bộ, triển khai tính toán song song trên các instance AWS EC2 có GPU để tăng tốc quá trình. Áp dụng các kỹ thuật tối ưu hóa như phân tích theo lô và lưu trữ kết quả trung gian.

* + - 1. Cân bằng giữa độ chính xác và khả năng giải thích:

Khó khăn: Các mô hình phức tạp thường có độ chính xác cao nhưng khó giải thích, tạo ra thách thức khi cần truyền đạt kết quả cho các chuyên gia trong lĩnh vực sinh học.

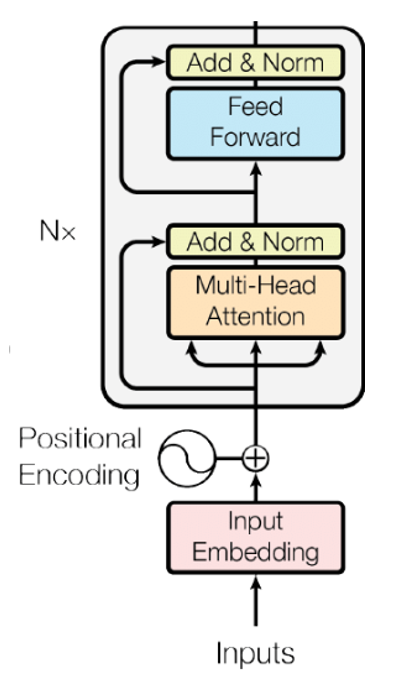
Giải pháp: Xây dựng kiến trúc hybrid kết hợp các thành phần có thể giải thích được với các mô hình học sâu. Phát triển các phương pháp trực quan hóa tiên tiến để làm rõ quá trình ra quyết định của mô hình và tạo cầu nối giữa độ phức tạp của mô hình với nhu cầu hiểu biết của người dùng.

# **CHƯƠNG III: CÀI ĐẶT THỰC NGHIỆM, KẾT QUẢ VÀ ĐÁNH GIÁ**

* 1. **Quá trình thực hiện và thiết kế mô hình**
     1. ***Phân tích yêu cầu và lựa chọn giải pháp***

Qua quá trình phân tích chi tiết và nghiên cứu tài liệu liên quan, đề án đã xác định được các yêu cầu chức năng cốt lõi cần thiết cho việc phát triển mô hình dự đoán hoạt tính peptide dựa trên trí tuệ nhân tạo. Các yêu cầu này được đúc kết như sau:

* Xử lý và chuẩn hóa dữ liệu peptide: Xây dựng quy trình xử lý dữ liệu peptide đa dạng từ nhiều nguồn dữ liệu khác nhau, bao gồm các bước tiền xử lý, chuẩn hóa và biểu diễn chuỗi amino acid.
* Phát triển mô hình học sâu đặc thù: Triển khai mô hình Transformer - một kiến trúc mạng neural tiên tiến được tối ưu hóa cho bài toán xử lý chuỗi, nhằm nắm bắt hiệu quả các đặc trưng phức tạp trong cấu trúc chuỗi peptide. Mô hình này cần được điều chỉnh các tham số để phù hợp với đặc thù dữ liệu sinh học, đồng thời tích hợp các cơ chế học hiệu quả từ lượng dữ liệu hạn chế thường gặp trong nghiên cứu peptide.
* Tích hợp kỹ thuật Explainable AI (XAI): Áp dụng các phương pháp XAI tiên tiến để giải thích vai trò của từng amino acid và đặc tính lý hóa trong việc quyết định hoạt tính của peptide, từ đó nâng cao độ tin cậy của mô hình dự đoán và cung cấp thông tin hữu ích cho quá trình thiết kế peptide mới. Điều này giúp mô hình không chỉ là một “hộp đen” đưa ra dự đoán mà còn cung cấp cái nhìn sâu sắc về cơ chế hoạt động của peptide.
  + 1. ***Thiết kế chi tiết*** 
       1. Mô hình Transformer Encoder



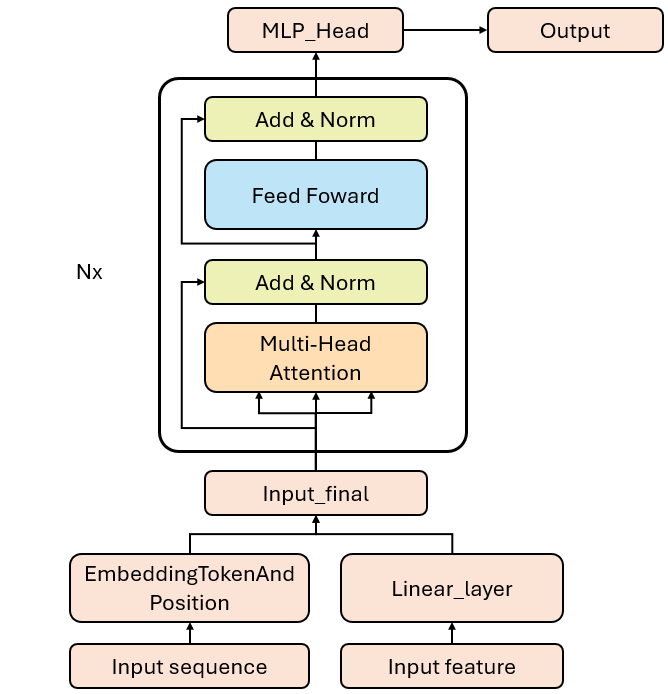
Sơ đồ 3. 1. Kiến trúc Transformer Encoder

Kiến trúc Encoder gốc của Transformer theo bài báo “Attention is All You Need” (Vaswani et al., 2017) hoạt động với các thành phần chính:

* Inputs: Dữ liệu đầu vào trình tự peptide
* Input Embedding: Các token amino acid được chuyển thành vector nhúng có chiều cao
* Positional Encoding: Thông tin vị trí được mã hóa vào embedding do Transformer không có cơ chế tuần tự như RNN
* Multi-Head Attention: Học các mối quan hệ khác nhau giữa các vị trí amino acid trong chuỗi peptide
* Add & Norm: Kết nối phần dư (residual connection) kết hợp với chuẩn hóa lớp (Layer Normalization)
* Feed Forward: Mạng nơ-ron hai lớp áp dụng cho từng vị trí amino acid
* Nx: Các khối được lặp lại N lần.

Trong quá trình xử lý, Encoder biến đổi chuỗi peptide đầu vào thành biểu diễn vector mang ngữ nghĩa về cấu trúc và tính chất hoá học.

* + - 1. Mô hình được xây dựng lại từ Transformer Encoder



Sơ đồ 3. 2. Kiến trúc được xây dựng lại từ Transformer Encoder gốc cho phù hợp

Phiên bản Transformer Encoder đã được điều chỉnh một cách kỹ lưỡng để phù hợp với đặc thù của bài toán phân loại peptide kháng khuẩn, thể hiện qua kiến trúc được thiết kế như trong sơ đồ. Cấu trúc này không chỉ thể hiện sự linh hoạt của mô hình Transformer mà còn phản ánh sự tối ưu hóa cho dữ liệu sinh học đặc biệt.

* + - * 1. *Đầu vào kép*

Kiến trúc được thiết kế với cơ chế đầu vào kép, cho phép tích hợp đồng thời hai nguồn thông tin bổ sung cho nhau. Input sequence đại diện cho chuỗi amino acid của peptide - thông tin cốt lõi về trình tự và cấu trúc sơ cấp, trong khi Input feature chứa đặc trưng GPSD (General Protein Sequence Descriptors) bao gồm 126 đặc trưng số mô tả các tính chất lý hóa quan trọng của peptide như tính kỵ nước, điện tích, cấu trúc thứ cấp tiềm năng và các tham số khác có ảnh hưởng đến hoạt tính sinh học.

* + - * 1. *Xử lý song song*

Quá trình xử lý được thiết kế theo hướng song song để tối ưu hóa hiệu suất tính toán, với hai nhánh xử lý riêng biệt trước khi tích hợp thông tin:

* EmbeddingTokenAndPosition đảm nhiệm việc mã hóa các amino acid thành các vector nhúng đa chiều, nắm bắt không chỉ bản chất của mỗi amino acid mà còn tích hợp thông tin vị trí trong chuỗi - một yếu tố quan trọng vì hoạt tính của peptide thường phụ thuộc vào vị trí tương đối của các amino acid. Quá trình embedding này chuyển đổi dữ liệu rời rạc (chuỗi ký tự) thành không gian vector liên tục, tạo điều kiện cho các phép toán học sâu tiếp theo. Linear\_layer thực hiện biến đổi tuyến tính đối với vector đặc trưng GPSD thông qua các phép nhân ma trận và phép cộng bias. Lớp này không chỉ đơn thuần thực hiện phép chiếu tuyến tính mà còn có vai trò quan trọng trong việc chuyển đổi kích thước của vector đặc trưng để đảm bảo tính tương thích với kích thước của vector embedding, tạo điều kiện cho việc kết hợp thông tin từ hai nguồn.

Kết hợp thông tin: Input\_final đóng vai trò then chốt trong kiến trúc bằng cách kết hợp hai nguồn thông tin đã được xử lý: vector embedding chứa thông tin trình tự và vị trí, cùng với vector đặc trưng GPSD đã qua biến đổi. Quá trình kết hợp này có thể được thực hiện thông qua phép cộng vector (element-wise addition) hoặc phép nối (concatenation) tùy thuộc vào đặc điểm dữ liệu và yêu cầu mô hình hóa. Việc lựa chọn giữa hai phương pháp này ảnh hưởng đến số lượng tham số và khả năng học của mô hình, với phép nối thường cung cấp biểu diễn phong phú hơn nhưng đòi hỏi nhiều tham số hơn.

* + - * 1. *Encoder blocks*

Phần cốt lõi của kiến trúc là chuỗi các Encoder blocks được lặp lại Nx lần (với N là siêu tham số có thể điều chỉnh). Mỗi block bao gồm ba thành phần chính:

* Multi-Head Attention là cơ chế độc đáo của Transformer, cho phép mô hình tập trung vào nhiều vị trí khác nhau trong chuỗi đầu vào đồng thời, từ đó nắm bắt được các mối tương quan phức tạp giữa các amino acid và các đặc trưng lý hóa kèm theo. Cơ chế này đặc biệt hữu ích trong việc học các tương tác khoảng xa trong chuỗi peptide, vượt trội so với các mô hình CNN hoặc RNN truyền thống.
* Feed Forward Network là mạng neural feed-forward đơn giản gồm hai lớp tuyến tính với hàm kích hoạt phi tuyến ở giữa, thường là ReLU. Thành phần này gia tăng khả năng biểu diễn phi tuyến tính của mô hình, cho phép học các mối quan hệ phức tạp hơn.
* Add & Norm là lớp kết nối tắt (residual connection) kết hợp với layer normalization, giúp tránh vấn đề gradient vanishing/exploding và ổn định quá trình huấn luyện, đặc biệt quan trọng khi số lượng Encoder blocks tăng lên.
* Việc lặp lại các block này Nx lần cho phép mô hình xây dựng biểu diễn phân cấp với mức độ trừu tượng ngày càng cao, từ đó trích xuất được các đặc trưng phức tạp từ dữ liệu peptide mà các mô hình đơn giản không thể nắm bắt được.
  + - * 1. *Đầu ra*

Phần cuối của kiến trúc gồm MLP\_Head - một mạng perceptron đa lớp chuyên biệt hóa cho bài toán phân loại. Thành phần này biến đổi biểu diễn phong phú cuối cùng thu được từ chuỗi Encoder blocks thành dự đoán nhị phân, cung cấp xác suất peptide được phân loại là kháng khuẩn (AMP - Antimicrobial Peptide) hoặc không kháng khuẩn (nAMP - non-Antimicrobial Peptide). Cấu trúc MLP\_Head thường bao gồm một hoặc nhiều lớp ẩn với hàm kích hoạt phi tuyến, và lớp đầu ra sử dụng hàm sigmoid để đưa ra xác suất trong khoảng [0, 1].

Sau khi phân tích chi tiết mô hình, việc triển khai code sẽ được thực hiện bằng các framework học sâu hiện đại như TensorFlow hoặc PyTorch. Quá trình này đòi hỏi sự chú ý đặc biệt đến một số khía cạnh quan trọng ảnh hưởng đến hiệu suất của mô hình:

* Số lượng head trong Multi-Head Attention cần được cân nhắc kỹ lưỡng - quá ít có thể giới hạn khả năng nắm bắt các mối tương quan phức tạp, trong khi quá nhiều có thể dẫn đến overfitting và tăng chi phí tính toán không cần thiết. Thông thường giá trị từ 4 đến 8 head được coi là phù hợp cho dữ liệu peptide.
* Kích thước của embedding vector là một siêu tham số quan trọng ảnh hưởng trực tiếp đến khả năng biểu diễn thông tin của mô hình. Vector quá nhỏ có thể không đủ sức chứa thông tin phức tạp của amino acid và vị trí, trong khi vector quá lớn dễ dẫn đến overfitting, đặc biệt với bộ dữ liệu peptide thường có kích thước khiêm tốn.
* Cơ chế dropout được tích hợp vào nhiều vị trí trong mô hình nhằm ngăn chặn overfitting - một thách thức phổ biến trong bài toán sinh học với dữ liệu hạn chế. Tỷ lệ dropout cần được tinh chỉnh cẩn thận, thường từ 0.1 đến 0.3 tùy thuộc vào kích thước dữ liệu và độ phức tạp của mô hình.
* Kỹ thuật normalization, đặc biệt là layer normalization, đóng vai trò then chốt trong việc ổn định quá trình huấn luyện Transformer. Khác với batch normalization thường dùng trong CNN, layer normalization hoạt động hiệu quả hơn với dữ liệu chuỗi và không phụ thuộc vào kích thước batch, phù hợp với đặc thù của dữ liệu sinh học.

Việc sử dụng kiến trúc hybrid kết hợp cả thông tin trình tự và đặc trưng lý hóa là một điểm sáng trong thiết kế này. Phương pháp này vượt trội so với các mô hình truyền thống chỉ sử dụng một loại dữ liệu, cho phép nắm bắt đồng thời cả thông tin cấu trúc từ trình tự amino acid và các đặc tính biến thiên từ đặc trưng GPSD. Nhiều nghiên cứu gần đây trong lĩnh vực sinh tin học đã chứng minh rằng các mô hình hybrid tương tự có khả năng dự đoán chính xác hơn đáng kể, đặc biệt là trong các bài toán phân loại peptide nơi cả trình tự và tính chất lý hóa đều đóng vai trò quan trọng trong việc quyết định hoạt tính sinh học.

* + 1. ***Triển khai dựa trên mô hình kiến trúc***
       1. Thu thập dữ liệu

Trong giai đoạn này, việc thu thập dữ liệu chuỗi peptide (thuộc họ vi sinh vật) kháng khuẩn (AMP) và không kháng khuẩn (nAMP) từ cơ sở dữ liệu NCBI Protein được thực hiện. Quá trình này sử dụng công cụ Entrez của thư viện Biopython, cho phép truy vấn và tải về dữ liệu từ các cơ sở dữ liệu sinh học trực tuyến.

Thiết lập kết nối với NCBI: mô hình được thiết kế để sử dụng module Entrez từ thư viện Biopython để giao tiếp với cơ sở dữ liệu NCBI. Thông tin xác thực bao gồm email và API key (nếu có) được cung cấp để đáp ứng yêu cầu của NCBI, nhằm tránh bị chặn do vi phạm giới hạn truy vấn trong quá trình thu thập dữ liệu lớn.

* + - * 1. *Quản lý dữ liệu hiện có*

Trước khi tải dữ liệu mới, mô hình thực hiện kiểm tra các tệp FASTA đã lưu trữ (nếu tồn tại) để tránh trùng lặp:

* Đọc từng chuỗi sequence từ tệp FASTA và lưu vào một tập hợp (set) để tối ưu việc kiểm tra trùng lặp với độ phức tạp O(1).
* Nếu số lượng sequence hiện có đã đạt ngưỡng mong muốn (ví dụ: 2.6 triệu), quá trình tải sẽ dừng ngay lập tức để tiết kiệm tài nguyên.
  + - * 1. *Truy vấn dữ liệu AMP/nAMP từ GenBank*

Bước 1: Tìm kiếm tổng quan, chương trình sử dụng hàm esearch để gửi truy vấn đến database “protein” của NCBI với từ khóa đặc thù (ví dụ: “antimicrobial peptide” cho AMP hoặc loại trừ từ khóa AMP cho nAMP). Kết quả trả về bao gồm tổng số bản ghi phù hợp (Count), cùng với WebEnv và QueryKey để truy xuất dữ liệu theo từng batch mà không cần gửi lại truy vấn ban đầu.

Bước 2: Tải dữ liệu theo batch, dữ liệu được tải về theo nhóm (mỗi nhóm tối đa 5.000 bản ghi). Sequence mới được thêm vào tập hợp existing\_sequences để đảm bảo không trùng lặp ở các batch sau.

* + - * 1. *Kiểm soát quá trình thu thập*

Lưu trữ dữ liệu: dữ liệu được ghi vào tệp FASTA ở chế độ append ('a'), đảm bảo không ghi đè dữ liệu cũ khi chương trình chạy nhiều lần. Định dạng header chuẩn: >*ID [AMP/nAMP]* để dễ dàng phân biệt loại sequence trong các bước xử lý sau.

* + - * 1. *Giới hạn và rủi ro*

Hạn chế API NCBI: Nếu không có API key, giới hạn request là 10 request/giây, nhưng code đã đặt độ trễ 0.4 giây để đảm bảo an toàn.

Khả năng thiếu dữ liệu: Nếu tổng số bản ghi tìm được ít hơn amp\_count/namp\_count, mô hình sẽ dừng sau khi tải hết dữ liệu khả dụng từ NCBI.

* + - 1. Tiền xử lý dữ liệu
         1. *Quy ước về tách chuỗi peptide theo độ dài ổn định*

Amino acid được phân loại thành hai nhóm chính dựa trên đặc tính của nhóm R: phân cực (polar) và không phân cực (non-polar) [20].

Amino acid phân cực bao gồm hai loại [20]:

* + **Tích điện** (acidic/basic): Asp (D), Glu (E) mang điện tích âm; Lys (K), Arg (R), His (H) mang điện tích dương.
  + **Không tích điện** (hydrophilic): Ser (S), Thr (T), Asn (N), Gln (Q), Tyr (Y), Cys (C).

Amino acid không phân cực (hydrophobic): Gly (G), Ala (A), Val (V), Leu (L), Ile (I), Met (M), Pro (P), Phe (F), Trp (W).

* + - * 1. *Tiền xử lý dữ liệu chuỗi – trước khi tách chuỗi*

Quá trình tiền xử lý dữ liệu chuỗi peptide trước khi tách nhỏ được thực hiện nhằm đảm bảo tính nhất quán, loại bỏ thông tin dư thừa và chuẩn hóa định dạng đầu vào. Dưới đây là các bước chi tiết:

Mô hình kiểm tra sự tồn tại của tệp FASTA đầu ra (nếu có) để tránh trùng lặp dữ liệu.

Mỗi chuỗi peptide được kiểm tra độ dài ngay sau khi hoàn thiện:

* + Nếu chuỗi có độ dài ≤ 50 amino acid, nó được coi là đạt yêu cầu và chuyển thẳng đến bước ghi vào tệp đầu ra (sau khi kiểm tra trùng lặp).
  + Nếu chuỗi dài > 50 amino acid, nó sẽ được chuyển sang quy trình tách chuỗi con để trích xuất các đoạn peptide ngắn hơn đáp ứng tiêu chí sinh học.
    - * 1. *Quy ước về sinh 126 đặc trưng dựa vào đặc tính lý hóa của amino acid*

Để trích xuất thông tin từ chuỗi protein, các đặc điểm được tính dựa trên ba thành phần chính:

Composition (C): Mô tả thành phần tổng thể của chuỗi protein, tức là tỷ lệ phần trăm các nhóm axit amin trong chuỗi.

Transition (T): Tần suất thay đổi tính chất (như sự chuyển đổi từ nhóm polar sang hydrophobic) dọc theo toàn bộ chiều dài chuỗi protein.

Distribution (D): Mô tả mẫu phân bố các tính chất (polar, neutral, hydrophobic) trên toàn bộ trình tự protein.

|  |  |
| --- | --- |
| Thành phần | Số lượng đặc trưng mỗi thuộc tính |
| Composition (C) | 3 đặc trưng (Polar, Neutral, Hydrophobic) |
| Transition (T) | 3 đặc trưng (chuyển đổi giữa các nhóm) |
| Distribution (D) | 6 × 3 = 18 đặc trưng (vị trí 0%, 20%, 40%, 60%, 80%, 100% cho mỗi nhóm) |

Bảng 3. 1. Mô tả sinh ra số lượng đặc trưng từ chuỗi peptide

Trong bài dùng 7 thuộc tính lý hóa (properties) cho một chuỗi peptide: tổng số đặc trưng là 7 (properties) × 18 = 126 đặc trưng

* + - * 1. *Tiền xử lý dữ liệu – sau khi tách chuỗi và sinh đặc trưng*

Dữ liệu được xử lý bao gồm hai tập dữ liệu chính là data (dùng để huấn luyện) và data\_test (dùng để kiểm tra). Các bước thực hiện trong quá trình xử lý dữ liệu được tiến hành như sau:

Kiểm tra và xử lý dữ liệu khuyết thiếu, sau khi kiểm tra các cột có giá trị NaN, không phát hiện thấy bất kỳ giá trị khuyết thiếu nào trong dữ liệu.

Cột Label chứa nhãn phân loại, với giá trị nAMP được chuyển thành 0 (không kháng khuẩn) và các giá trị khác chuyển thành 1 (kháng khuẩn). Việc chuyển đổi này đã được thực hiện thành công và kiểu dữ liệu của Label đã được chuyển thành float.

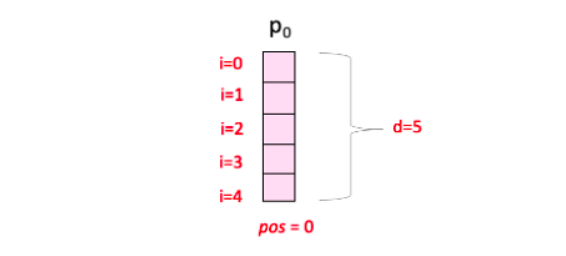
Dữ liệu được tách thành các tập con:

* X\_train và y\_train: Dùng để huấn luyện mô hình
* X\_test và y\_test: Dùng để kiểm tra mô hình
* X\_test tiếp tục được chia thành X\_val và X\_test với tỷ lệ 50% cho mỗi tập dữ liệu kiểm tra.

Chuỗi peptide ban đầu ở dạng văn bản được chuyển đổi thành các chuỗi số để có thể sử dụng trong mô hình học máy. Công cụ Tokenizer từ TensorFlow được sử dụng để mã hóa chuỗi văn bản thành các số.

* + - 1. Xây dựng mô hình Transformer
         1. *Phân tích chi tiết các thành phần*
  1. Lớp Token và Position Embedding

Do Transformer không có khả năng mặc định để nhận biết thứ tự các token, *Positional Encoding* được thêm vào embedding đầu vào để mã hóa thông tin vị trí. Công thức sử dụng hàm sine và cosine [2]:



Hình 3. 1. Ví dụ Position Embebđing

*,*

Trong đó [2]:

* pos: Vị trí của token trong chuỗi.
* i: Chỉ số chiều trong embedding

Việc xử lý chuỗi amino acid cần được chuyển đổi thành dạng số để mô hình có thể hiểu được. Quá trình này được thực hiện qua hai bước chính:

* *Token Embedding*: Chuyển đổi mỗi amino acid thành vector thông qua lớp *Embedding*
* *Positional Embedding*: Thêm thông tin vị trí vào vector embedding, giúp mô hình hiểu được thứ tự xuất hiện của amino acid

Đối với vấn đề xử lý chuỗi peptide dài, đã áp dụng *GlobalAveragePooling1D* để rút ngắn chiều dài từ tối đa của chuỗi peptide xuống 50.

* 1. Cơ chế Self-Attention

Self-Attention: cho phép mỗi token trong chuỗi đầu vào “chú ý” đến tất cả các token khác để xác định mức độ quan trọng của chúng. Công thức tính toán Self-Attention như sau [1]:

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

Trong đó:

* Q(Query), K(Key), V(Value) là các ma trận được tạo bằng cách nhân đầu vào với các trọng số .
* là hệ số chuẩn hóa để tránh giá trị điểm attention quá lớn, gây ra gradient không ổn định.

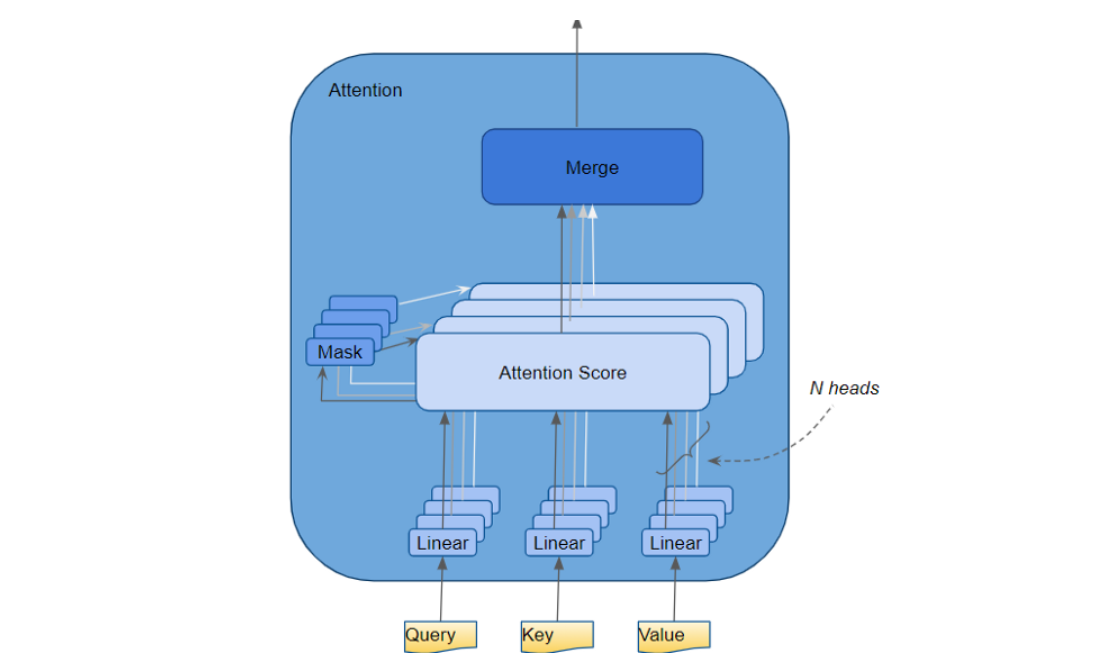
Ví dụ: Với trình tự peptide “Ala-Gly-Ser-Val”, mỗi amino acid được biểu diễn bằng một vector. QQ, KK, VV giúp xác định mối quan hệ giữa các amino acid này.

* Tính điểm Attention [1]:
  + : Đo độ tương đồng giữa các amino acid. Amino acid ở vị trí i(Query) sẽ so sánh với tất cả các vị trí j (Key) để xác định mức độ liên quan.
  + : Chuẩn hóa để tránh gradient quá lớn, giúp quá trình huấn luyện ổn định.
  + Softmax: Chuẩn hóa điểm thành phân phối xác suất (tổng bằng 1), thể hiện trọng số đóng góp của từng amino acid.

Nhân với V: Tổng hợp thông tin từ các amino acid quan trọng nhất.

* 1. Multi-Head Attention

*Multi-Head Attention:* Mở rộng cơ chế này bằng cách thực hiện Self-Attention song song trên h ”đầu” (heads), sau đó ghép nối kết quả và chiếu xuống không gian ban đầu [1]:



Sơ đồ 3. 3. Cơ chế của Multihead Attention

MultiHead(*Q*,*K*,*V*) = Concat(head1,…..headh)

Mỗi “đầu” tính toán độc lập [1]:

*Headi = Attention()*

Giải thích: Thay vì dùng một cơ chế Self-Attention duy nhất, Multi-Head chia đầu vào thành h ”đầu” (thường h = 8), mỗi đầu học cách tập trung vào khía cạnh khác nhau của dữ liệu [1].

Ví dụ:

* Đầu 1: Tập trung vào tương tác hóa học giữa các amino acid.
* Đầu 2: Phân tích cấu trúc bậc hai (alpha-helix, beta-sheet).
* Đầu 3: Xem xét vị trí tương đối trong chuỗi.

Ghép nối và chiếu: Kết quả từ các đầu được ghép lại và chiếu xuống không gian ban đầu qua , tổng hợp đa dạng thông tin.

*Trong bài ở Multi-Head Attention*: Sử dụng 6 đầu *attention, c*ơ chế này giúp mô hình phát hiện các mối liên kết giữa các amino acid, ví dụ như nhóm kỵ nước thường xuất hiện gần nhau.

* 1. Feed Forward Network

*Feed Forward Network*: mỗi lớp FFN gồm hai phép biến đổi tuyến tính với hàm kích hoạt ReLU hoặc GELU ở giữa:

*FFN(x) = ReLU()*

Giải thích:

* Chức năng: biến đổi phi tuyến các đặc trưng trích xuất từ Self-Attention. Mỗi FFN gồm hai lớp tuyến tính với hàm kích hoạt ReLU/GELU ở giữa.
* Vai trò: Mở rộng không gian đặc trưng (thường ). Học mẫu hình phức tạp hơn, ví dụ: tương tác không gian 3D của peptide.

Trong bài ở FFN gồm hai lớp *Dense* lần lượt có 500 và 250 *nơ-ron* với hàm kích hoạt *ReLU*. Cấu trúc này giúp mô hình học được các đặc trưng phi tuyến phức tạp.

* 1. Layer Normalization và Dropout

Để ổn định quá trình huấn luyện, đầu ra của mỗi lớp Self-Attention và FFN được chuẩn hóa bằng *Layer Normalization* và kết hợp với đầu vào gốc qua kết nối tắt (residual connection):

*Trong bài ở Layer Normalization* và *Dropout*: Mô hình kết hợp chuẩn hóa lớp để ổn định quá trình huấn luyện và Dropout với tỷ lệ 0.5 để ngăn overfitting.

* 1. Thiết kế lớp đầu ra

Sau khi đi qua các lớp Transformer, dữ liệu được xử lý cuối cùng như sau:

* Áp dụng GlobalAveragePooling1D để làm phẳng dữ liệu
* Đưa qua hai lớp Dense (400 và 200 nơ-ron) với Dropout 0.2 sau mỗi lớp
* Lớp cuối cùng sử dụng sigmoid để dự đoán nhãn nhị phân
  + - 1. Huấn luyện và đánh giá mô hình

Sau khi xây dựng mô hình Transformer, bước tiếp theo là huấn luyện mô hình và đánh giá hiệu quả của nó trên tập dữ liệu kiểm tra. Quá trình huấn luyện bao gồm các bước sau:

* Chuẩn bị dữ liệu huấn luyện và kiểm tra: Dữ liệu huấn luyện được chia thành các đặc trưng số và chuỗi. Cả hai loại dữ liệu này được đưa vào mô hình trong suốt quá trình huấn luyện. Việc đảm bảo dữ liệu được chuẩn bị đúng cách và có thể được sử dụng trong mô hình là rất quan trọng.

Mô hình được huấn luyện với tốc độ học rất nhỏ (learning\_rate=0.000000005) để đảm bảo hội tụ ổn định. Các kỹ thuật tối ưu hóa huấn luyện bao gồm:

* ModelCheckpoint: Lưu mô hình tốt nhất dựa trên val\_loss
* EarlyStopping: Dừng huấn luyện sau 5 epoch không cải thiện val\_loss
* ReduceLROnPlateau: Giảm tốc độ học xuống 10% sau 2 epoch không cải thiện val\_loss

Quá trình huấn luyện được thực hiện với batch\_size=32 và tối đa 200 epochs, sử dụng hàm mất mát binary\_crossentropy và đo lường accuracy.

Đánh giá mô hình: Sau khi huấn luyện xong, tập kiểm tra được sử dụng để đánh giá hiệu suất của mô hình. Các chỉ số như độ chính xác (accuracy) và mất mát (loss) được theo dõi để đánh giá chất lượng mô hình.

Huấn luyện mô hình: Mô hình được huấn luyện trong số epoch quy ước, cùng với kích thước của batch được quy ước cho phù hợp, và sử dụng các *callback* như *ModelCheckpoint*, *EarlyStopping*, và *ReduceLROnPlateau* để tối ưu hóa quá trình huấn luyện.

* + - 1. Ứng dụng Explainable Artificial Intelligence (XAI) để giải thích thành phần quan trọng của chuỗi peptide

SHAP kết hợp giá trị Shapley từ lý thuyết trò chơi với các phương pháp xấp xỉ hiệu quả, tạo nên quy trình tính toán tối ưu [7].

* + - * 1. *Xác định đặc trưng*

Bài toán: Giải thích sự khác biệt giữa dự đoán của một mẫu cụ thể x và giá trị dự đoán trung bình .

Người chơi: Các đặc trưng (features) của mô hình (ví dụ: amino acid trong peptide, cấu trúc không gian).

Giá trị Shapley: đóng góp của đặc trưng i vào sự khác biệt .

* + - * 1. *Tính toán giá trị Shapley*

Cơ sở lý thuyết với giá trị Shapley: Đo lường đóng góp trung bình của một đặc trưng thông qua phân tích toàn diện các tổ hợp đặc trưng có thể xảy ra. Công thức toán học [7]:

Trong đó:

F: tập hợp tất cả các đặc trưng.

S: Tập con của F

f(S): Giá trị dự đoán khi chỉ sử dụng tập đặc trưng S.

Giải thích từng bước:

Bước 1: Liệt kê tất cả tổ hợp đặc trưng: Xét mọi tập con S không chứa đặc trưng i.

Bước 2: Tính đóng góp biên tế: Với mỗi S, tính sự thay đổi dự đoán khi thêm i vào S:

Bước 3: Gán trọng số: Mỗi được nhân với trọng số , phản ánh xác suất xuất hiện của tổ hợp S.

Bước 4: Tổng hợp: Cộng tất cả  đã được trọng số để ra

* + - * 1. *Xử lý thách thức tính toán*

Với M đặc trưng, số tổ hợp S là (rất lớn khi M > 20). SHAP sử dụng phương pháp xấp xỉ [7]:

Kernel SHAP (cho mô hình “hộp đen”): Áp dụng cho mọi mô hình (kể cả học sâu). Tuy nhiên, chậm với dữ liệu lớn do số lượng tổ hợp tăng theo cấp số nhân [7]:

Bước 1: Tạo các tổ hợp đặc trưng giả bằng cách “bật/tắt” đặc trưng:

* Đặc trưng “bật”: Giữ giá trị thực của mẫu x*x*.
* Đặc trưng “tắt”: Thay bằng giá trị baseline (ví dụ: trung bình, median).

Bước 2: Dự đoán giá trị cho các tổ hợp này bằng mô hình f

Bước 3: Áp dụng hồi quy tuyến tính để tìm hệ số  sao cho:

Tree SHAP (cho mô hình cây): Độ phức tạp giảm từ xuống . Tối ưu hóa dựa trên cấu trúc cây quyết định [7]:

Bước 1: Duyệt cây từ gốc đến lá, theo giá trị đặc trưng của mẫu x

Bước 2: Tính xác suất đi qua các nhánh để xác định tổ hợp S

Bước 3: Dùng công thức đệ quy để tính trực tiếp từ các nút, tránh liệt kê tổ hợp.

Tối ưu hóa dựa trên cấu trúc cây quyết định:

Bước 1: Duyệt cây từ gốc đến lá, theo giá trị đặc trưng của mẫu x.

Bước 2: Tính xác suất đi qua các nhánh để xác định tổ hợp

Phương pháp xấp xỉ tính toán tiên tiến:

* Kernel SHAP: Áp dụng kỹ thuật hồi quy tuyến tính để ước lượng giá trị Shapley, đặc biệt phù hợp với các mô hình phức tạp.
* Tree SHAP: Phương pháp tối ưu hóa chuyên biệt cho các mô hình dựa trên cây (Random Forest, XGBoost), giảm đáng kể độ phức tạp tính toán từ xuống . Với T là số cây, L là số lá, D là độ sâu của cây.

Biểu đồ SHAP đa dạng:

* Summary plot: Trực quan hóa tầm quan trọng toàn cục của các đặc trưng.
* Force plot: Cung cấp giải thích chi tiết cho từng dự đoán cụ thể, minh họa cách các đặc trưng “tác động” đến kết quả.

Dependence plot: Phân tích chuyên sâu mối tương tác giữa các đặc trưng.

* + 1. ***Các khó khăn và thách thức trong quá trình thực hiện***
  1. **Kết quả đạt được**

Thu thập dữ liệu: Mô hình có khả năng thu thập dữ liệu chuỗi peptide kháng khuẩn (AMP) và không kháng khuẩn (nAMP) từ cơ sở dữ liệu NCBI Protein một cách tự động.

Tiền xử lý dữ liệu: Mô hình thực hiện tiền xử lý dữ liệu chuỗi peptide thông qua các bước chuẩn hóa, tách chuỗi dài và tạo ra 126 đặc trưng bằng cách dùng phương pháp GPSD.

Phân tích đặc tính: Phân tích các đặc tính lý hóa của chuỗi peptide bao gồm độ kỵ nước, độ phân cực, khả năng phân cực, điện tích, cấu trúc thứ cấp và khả năng tiếp xúc với dung môi.

Dự đoán tính kháng khuẩn: Mô hình sử dụng kiến trúc Transformer để dự đoán tính kháng khuẩn của chuỗi peptide dựa trên cả dữ liệu chuỗi và đặc trưng số hóa.

Đánh giá mô hình: Mô hình cung cấp các công cụ đánh giá hiệu suất dự đoán thông qua các chỉ số như độ chính xác, độ nhạy, độ đặc hiệu và điểm F1.

* + 1. ***Kết quả triển khai mô hình***

Hiệu suất dự đoán: Mô hình đạt độ chính xác cao trong việc dự đoán tính kháng khuẩn của chuỗi peptide, với các chỉ số đánh giá được xác thực trên tập dữ liệu kiểm tra.

Khả năng mở rộng: Mô hình có thể mở rộng để xử lý các loại chuỗi peptide khác nhau và tích hợp thêm các đặc trưng mới.

Thời gian huấn luyện: Quá trình huấn luyện mô hình được tối ưu hóa với các kỹ thuật như Early Stopping và Reduce Learning Rate on Plateau, giảm đáng kể thời gian huấn luyện.

* + 1. ***Yêu cầu phần cứng và môi trường cài đặt***
       1. Yêu cầu phần cứng
* RAM: Tối thiểu 8GB, khuyến nghị 16-32GB hoặc cao hơn khi xử lý dữ liệu lớn.
* GPU: Khuyến nghị sử dụng GPU có VRAM từ 8GB trở lên (như NVIDIA RTX series) để tăng tốc quá trình huấn luyện.
* Ổ cứng: Tối thiểu 50GB bộ nhớ trống để lưu trữ dữ liệu và mô hình
  + - 1. Môi trường cài đặt
* Thư viện chính: TensorFlow 2.8+, Keras, Biopython, NumPy, Pandas, Scikit-learn
  1. **Đánh giá và phân tích kết quả thử nghiệm**
     1. ***Phương pháp kiểm tra và đánh giá mô hình***

Nghiên cứu này triển khai phương pháp đánh giá mô hình một cách có hệ thống thông qua kỹ thuật hold-out validation trên tập dữ liệu kiểm tra. Để đánh giá hiệu suất mô hình một cách toàn diện, nhiều chỉ số quan trọng đã được theo dõi bao gồm accuracy (phản ánh tỷ lệ phân loại chính xác), loss (đánh giá mức độ sai lệch của dự đoán), và val\_accuracy (đo lường khả năng tổng quát hóa trên dữ liệu chưa gặp).

Trong quá trình huấn luyện, hai callback quan trọng đã được tích hợp: EarlyStopping để tự động dừng quá trình khi hiệu suất không cải thiện sau một số epoch nhất định, nhằm ngăn chặn hiện tượng overfitting; và ModelCheckpoint để lưu trữ phiên bản mô hình có hiệu suất tối ưu nhất dựa trên chỉ số val\_loss. Việc kết hợp hai cơ chế này đảm bảo mô hình đạt được điểm cân bằng tối ưu giữa độ chính xác và khả năng tổng quát hóa.

Ngoài ra, nghiên cứu còn áp dụng kỹ thuật ReduceLROnPlateau để động điều chỉnh learning rate khi val\_loss không cải thiện sau một số epoch nhất định, giúp mô hình vượt qua các local minima và hội tụ hiệu quả hơn. Việc kết hợp đồng thời các kỹ thuật regularization như dropout và batch normalization cũng đóng vai trò quan trọng trong việc ngăn chặn overfitting, đặc biệt khi làm việc với dữ liệu sinh học có tính đa dạng cao như chuỗi peptide.

* + 1. ***Phân tích độ chính xác, độ tin cậy của mô hình học sâu***

Kết quả thực nghiệm cho thấy kiến trúc Transformer đã thể hiện khả năng học tập nhanh chóng và hiệu quả trên dữ liệu peptide. Quá trình cải thiện hiệu suất qua các epoch có thể phân tích chi tiết như sau:

* + - 1. Phân tích huấn luyện mô hình trên bộ dữ liệu Gen Bank

A screenshot of a computer

AI-generated content may be incorrect.

Hình 3. 2. Kết quả huấn luyện mô hình trên bộ dữ liệu Gen Bank

Ngay từ epoch đầu tiên, mô hình đã thể hiện khả năng học tập đáng kể với accuracy đạt 55.79% và val\_accuracy lên tới 69.43%. Sự chênh lệch giữa hai chỉ số này gợi ý rằng tập huấn luyện có thể phức tạp hơn hoặc đa dạng hơn tập kiểm tra. Đến epoch thứ 3, mô hình đã cải thiện rõ rệt với accuracy tăng lên 71.85% và val\_accuracy đạt 79.67%, chứng tỏ khả năng học tập nhanh chóng các đặc trưng phân biệt giữa peptide kháng khuẩn và không kháng khuẩn.

Đặc biệt, quá trình hội tụ nhanh chóng của mô hình (đạt hiệu suất tối ưu chỉ sau khoảng 13 epoch) thể hiện ưu thế của kiến trúc Transformer so với các phương pháp truyền thống, có thể do cơ chế self-attention đã giúp mô hình hiệu quả trong việc nắm bắt các mối tương quan phức tạp giữa các vị trí acid amin trong chuỗi peptide.

Sự ổn định của mô hình trong các epoch sau, kèm theo việc val\_loss thay đổi không đáng kể từ epoch 20-22, cho thấy mô hình đã đạt đến trạng thái tối ưu và khả năng khai thác thông tin từ dữ liệu hiện tại đã gần như đạt ngưỡng giới hạn, phản ánh hiệu quả của kiến trúc được đề xuất trong việc mô hình hóa dữ liệu peptide.

A graph with blue and orange lines

AI-generated content may be incorrect.

Hình 3. 3. Kết quả Accuracy được huấn luyện trên bộ dữ liệu Gen Bank

Đặc biệt, việc val\_accuracy đạt 99.54% ở các epoch cuối cho thấy mô hình có khả năng dự đoán chính xác trên dữ liệu, một yếu tố then chốt khi ứng dụng vào việc phát hiện và thiết kế peptide kháng khuẩn mới. Điều này có ý nghĩa đặc biệt trong lĩnh vực y sinh, nơi độ tin cậy của các dự đoán có tầm quan trọng cao.

A graph with blue and orange lines

AI-generated content may be incorrect.

Hình 3. 4. Kết quả Loss được huấn luyện trên bộ dữ liệu Gen Bank

Ngoài ra, quá trình giảm loss liên tục và ổn định (từ 0.6837 xuống 0.0189) trên tập huấn luyện, đồng bộ với sự giảm val\_loss (từ 0.6261 xuống 0.0181) trên tập kiểm tra, cho thấy mô hình đã thực sự học được các đặc trưng có ý nghĩa sinh học, chứ không đơn thuần ghi nhớ dữ liệu huấn luyện. Điều này khẳng định tính robust của mô hình Transformer trong việc xử lý dữ liệu sinh học phức tạp như chuỗi peptide.

A screenshot of a computer screen

AI-generated content may be incorrect.

Hình 3. 5. Kết quả hiệu năng của mô hình khi được huấn luyện trên bộ dữ liệu Gen Bank

Sau quá trình huấn luyện mô hình Transformer trên tập dữ liệu peptide, hiệu năng mô hình được đánh giá thông qua các chỉ số thống kê phổ biến như precision, recall, f1-score và confusion matrix. Kết quả đạt được được trình bày như sau:

* Precision của lớp nAMP (Class 0) đạt 1.00 và của lớp AMP (Class 1) đạt 0.99, cho thấy mô hình có khả năng phân biệt chính xác giữa các peptide không kháng khuẩn và kháng khuẩn.
* Recall đạt 0.99 với lớp nAMP và 1.00 với lớp AMP, điều này chứng tỏ mô hình hầu như không bỏ sót các peptide kháng khuẩn.
* F1-score, là chỉ số tổng hợp giữa precision và recall, đều đạt giá trị 1.00 cho cả hai lớp, cho thấy sự cân bằng rất tốt giữa độ chính xác và khả năng phát hiện đúng của mô hình.

A blue and white graph

AI-generated content may be incorrect.

Hình 3. 6. Ma trận nhầm lẫn khi được huấn luyện trên bộ dữ liệu Gen Bank

Để làm rõ hơn mức độ chính xác của mô hình, ma trận nhầm lẫn (confusion matrix) được sử dụng để biểu diễn số lượng mẫu được phân loại đúng và sai:

* Trong số 172,500 mẫu thuộc lớp nAMP, có 171,557 mẫu được phân loại đúng và chỉ 943 mẫu bị nhầm lẫn thành AMP.
* Tương tự, trong 166,415 mẫu thuộc lớp AMP, có đến 165,893 mẫu được phân loại đúng, chỉ 522 mẫu bị phân loại nhầm.

Ma trận nhầm lẫn cho thấy tỷ lệ nhầm lẫn rất thấp, đặc biệt trong bối cảnh dữ liệu có kích thước lớn. Điều này chứng tỏ mô hình không chỉ đạt hiệu năng cao mà còn hoạt động ổn định trên cả hai lớp dữ liệu.

* + - 1. Phân tích huấn luyện mô hình trên bộ dữ liệu khác (PDB, RefSeq, UnitProtKB, Swisss-Prot, DDBJ, EMBL, PIR)

A screenshot of a computer program

AI-generated content may be incorrect.

Hình 3. 7. Kết quả huấn luyện trên bộ dữ liệu (PDB, RefSeq, UnitProtKB, Swisss-Prot, DDBJ, EMBL, PIR)

Ngay từ epoch đầu tiên, mô hình đã thể hiện khả năng học tập đáng kể với accuracy đạt 55.52% và val\_accuracy lên tới 64.44%. Sự chênh lệch giữa hai chỉ số này gợi ý rằng tập kiểm tra có thể đơn giản hơn hoặc có cấu trúc dễ nhận diện hơn so với tập huấn luyện. Đến epoch thứ 5, mô hình đã cải thiện rõ rệt với accuracy tăng lên 75.38% và val\_accuracy đạt 74.72%, chứng tỏ khả năng học tập nhanh chóng các đặc trưng phân biệt giữa peptide kháng khuẩn và không kháng khuẩn.

Sự ổn định của mô hình trong các epoch sau (37-45), kèm theo việc val\_loss thay đổi không đáng kể ở mức 0.0172, cho thấy mô hình đã đạt đến trạng thái tối ưu và khả năng khai thác thông tin từ dữ liệu hiện tại đã gần như đạt ngưỡng giới hạn. Điều này được khẳng định thêm bởi việc thuật toán ReduceLROnPlateau liên tục giảm learning rate xuống các giá trị cực nhỏ (đến 4.999999573445172e-15 ở epoch 44), phản ánh hiệu quả của kiến trúc được đề xuất trong việc mô hình hóa dữ liệu peptide.

A graph with a line

AI-generated content may be incorrect.

Hình 3. 8. Kết quả Accuracy khi được huấn luyện trên bộ dữ liệu (PDB, RefSeq, UnitProtKB, Swisss-Prot, DDBJ, EMBL, PIR)

Đặc biệt, việc val\_accuracy đạt 99.56% ở các epoch cuối cho thấy mô hình có khả năng dự đoán chính xác trên dữ liệu chưa từng gặp, một yếu tố then chốt khi ứng dụng vào việc phát hiện và thiết kế peptide kháng khuẩn mới. Điều này có ý nghĩa đặc biệt trong lĩnh vực y sinh, nơi độ tin cậy của các dự đoán có tầm quan trọng cao.

A graph with blue and orange lines

AI-generated content may be incorrect.

Hình 3. 9. Kết quả Loss khi được huấn luyện trên bộ dữ liệu (PDB, RefSeq, UnitProtKB, Swisss-Prot, DDBJ, EMBL, PIR)

Ngoài ra, quá trình giảm loss liên tục và ổn định (từ 0.6854 xuống 0.0185) trên tập huấn luyện, đồng bộ với sự giảm val\_loss (từ 0.6410 xuống 0.0172) trên tập kiểm tra, cho thấy mô hình đã thực sự học được các đặc trưng có ý nghĩa sinh học, chứ không đơn thuần ghi nhớ dữ liệu huấn luyện. Việc accuracy và val\_accuracy đạt mức tương đương nhau (khoảng 99.56%) ở các epoch cuối là bằng chứng rõ ràng nhất về sự không xuất hiện của hiện tượng overfitting, khẳng định tính robust của mô hình Transformer trong việc xử lý dữ liệu sinh học phức tạp như chuỗi peptide.

A screenshot of a computer screen

AI-generated content may be incorrect.

Hình 3. 10. Kết quả hiệu năng của mô hình khi được huấn luyện trên bộ dữ liệu (PDB, RefSeq, UnitProtKB, Swisss-Prot, DDBJ, EMBL, PIR)

Sau quá trình huấn luyện mô hình Transformer trên tập dữ liệu peptide, hiệu năng mô hình được đánh giá thông qua các chỉ số thống kê như precision, recall, f1-score và confusion matrix. Kết quả được thể hiện chi tiết như sau:

* Precision của lớp nAMP (Class 0) đạt 1.00 và của lớp AMP (Class 1) là 0.99, điều này cho thấy mô hình có độ chính xác rất cao trong việc phân biệt giữa peptide không kháng khuẩn và peptide kháng khuẩn.
* Chỉ số recall đạt 0.99 với lớp nAMP và 1.00 với lớp AMP, điều này chứng tỏ mô hình có khả năng phát hiện gần như toàn bộ các peptide kháng khuẩn, với tỷ lệ bỏ sót cực kỳ thấp.
* F1-score – chỉ số tổng hợp giữa precision và recall – đạt 1.00 cho cả hai lớp, phản ánh sự cân bằng tối ưu giữa độ chính xác và khả năng phát hiện đúng, ngay cả trong môi trường dữ liệu có tính mất cân bằng nhẹ.

A blue squares with white numbers

AI-generated content may be incorrect.

Hình 3. 11. Ma trận nhầm lẫn khi được huấn luyện trên bộ dữ liệu (PDB, RefSeq, UnitProtKB, Swisss-Prot, DDBJ, EMBL, PIR)

Bên cạnh các chỉ số thống kê, ma trận nhầm lẫn (confusion matrix) được sử dụng để minh họa trực quan hiệu năng mô hình:

* Trong 142,500 mẫu thuộc lớp nAMP, có 141,610 mẫu được phân loại đúng và chỉ 890 mẫu bị nhầm sang lớp AMP.
* Trong 141,000 mẫu thuộc lớp AMP, có 140,614 mẫu được nhận diện đúng, chỉ 386 mẫu bị phân loại sai thành nAMP.

Từ ma trận nhầm lẫn có thể thấy, tỷ lệ nhầm lẫn giữa hai lớp là rất nhỏ so với tổng thể mẫu, điều này củng cố thêm cho độ tin cậy và tính ổn định của mô hình trên cả hai lớp dữ liệu.

* + - 1. Phân tích huấn luyện mô hình trên bộ dữ liệu kết hợp

A screenshot of a computer screen

AI-generated content may be incorrect.

Hình 3. 12. Kết quả hiệu năng của mô hình khi được huấn luyện trên bộ dữ liệu kết hợp

Sau quá trình huấn luyện mô hình Transformer trên tập dữ liệu peptide, hiệu năng mô hình được đánh giá thông qua các chỉ số thống kê như precision, recall, f1-score và confusion matrix. Kết quả cụ thể được thể hiện như sau:

* Precision của lớp nAMP (Class 0) đạt 1.00, trong khi lớp AMP (Class 1) đạt 0.99, phản ánh độ chính xác rất cao của mô hình trong việc phân biệt giữa peptide không kháng khuẩn và peptide kháng khuẩn.
* Recall đạt 0.99 cho lớp nAMP và 1.00 cho lớp AMP, cho thấy mô hình có khả năng phát hiện gần như toàn bộ các peptide kháng khuẩn, đồng thời giảm thiểu tối đa tỷ lệ bỏ sót.
* F1-score, chỉ số tổng hợp giữa precision và recall, đều đạt 1.00 cho cả hai lớp. Điều này chứng tỏ mô hình duy trì được sự cân bằng tối ưu giữa độ chính xác và khả năng nhận diện đúng trong môi trường dữ liệu lớn.

A blue and white graph

AI-generated content may be incorrect.

Hình 3. 13. Ma trận nhầm lẫn khi được huấn luyện trên bộ dữ liệu kết hợp

Bên cạnh các chỉ số thống kê, ma trận nhầm lẫn (confusion matrix) cung cấp cái nhìn trực quan về hiệu năng mô hình:

* Trong 270,000 mẫu thuộc lớp nAMP, có 268,620 mẫu được phân loại đúng, chỉ 1,380 mẫu bị nhầm sang lớp AMP.
* Trong 270,000 mẫu thuộc lớp AMP, có 268,894 mẫu được phân loại chính xác, chỉ 1,106 mẫu bị nhầm lẫn với lớp nAMP.

Từ ma trận nhầm lẫn có thể thấy, tỷ lệ nhầm lẫn giữa hai lớp là cực kỳ thấp so với quy mô dữ liệu, cho thấy mô hình không chỉ có hiệu suất cao mà còn đáng tin cậy và ổn định khi áp dụng trên tập dữ liệu lớn.

* + 1. **So sánh kết quả huấn luyện khi kết hợp 2 bộ dữ liệu với kết quả huấn luyện trên từng bộ dữ liệu**

Sau khi thực hiện huấn luyện mô hình Transformer trên từng bộ dữ liệu riêng biệt (GenBank và nhóm dữ liệu PDB, RefSeq, UniProtKB, Swiss-Prot, DDBJ, EMBL, PIR), và sau đó huấn luyện lại trên bộ dữ liệu kết hợp từ cả hai nguồn, em nhận thấy có một số điểm khác biệt đáng chú ý, cả về độ chính xác, tính ổn định lẫn khả năng tổng quát hóa của mô hình. Các thông số cụ thể được tổng hợp trong bảng sau:

|  |  |  |  |
| --- | --- | --- | --- |
| Chỉ số đánh giá | GenBank | PDB, RefSeq,… | Kết hợp cả hai |
| Số mẫu kiểm tra | 338,915 | 283,500 | 540,000 |
| Precision (Class 0) | 1.00 | 1.00 | 1.00 |
| Precision (Class 1) | 0.99 | 0.99 | 0.99 |
| Recall (Class 0) | 0.99 | 0.99 | 0.99 |
| Recall (Class 1) | 1.00 | 1.00 | 1.00 |
| F1-score (Class 0) | 1.00 | 1.00 | 1.00 |
| F1-score (Class 1) | 1.00 | 1.00 | 1.00 |
| Accuracy | 1.00 | 1.00 | 1.00 |
| Sai số Class 0 → 1 | 943 | 890 | 1,380 |
| Sai số Class 1 → 0 | 522 | 386 | 1,106 |

Bảng 3. 2. So sánh kết quả huấn luyện khi kết hợp 2 bộ dữ liệu với kết quả huấn luyện trên từng bộ dữ liệu

Phân tích và nhận xét:

* Khi mô hình được huấn luyện trên từng bộ dữ liệu riêng lẻ, hiệu năng đạt mức rất cao với precision, recall, và F1-score gần như tuyệt đối cho cả hai lớp.
* Việc kết hợp hai bộ dữ liệu làm tăng đáng kể kích thước tập huấn luyện và kiểm tra, giúp mô hình học được nhiều đặc trưng hơn, từ đó nâng cao khả năng tổng quát hóa. Mặc dù số lượng mẫu sai sót có tăng lên đôi chút do quy mô dữ liệu lớn hơn, nhưng tỷ lệ nhầm lẫn vẫn duy trì ở mức rất thấp (*<0.5%*), cho thấy mô hình vẫn giữ được độ ổn định và tin cậy cao.
* Về mặt thống kê, tổng số mẫu bị phân loại sai trong mô hình kết hợp (2,486 mẫu) chiếm chỉ 0.46% trên tổng số 540,000 mẫu, chứng minh rằng việc mở rộng quy mô và độ đa dạng của dữ liệu không ảnh hưởng tiêu cực đến chất lượng mô hình.

Kết luận tổng thể: việc kết hợp hai bộ dữ liệu giúp cải thiện khả năng khái quát và giảm thiểu hiện tượng overfitting do tăng độ đa dạng về đặc trưng sinh học của peptide, đồng thời duy trì được hiệu năng cao tương đương như khi huấn luyện riêng lẻ. Điều này khẳng định rằng việc mở rộng dữ liệu đầu vào là hướng đi hiệu quả và cần thiết trong quá trình xây dựng mô hình học sâu cho bài toán phân loại peptide kháng khuẩn.

* + 1. **So sánh kết quả huấn luyện của mô hình Transformer với kết quả huấn luyện của mô hình LSTM**

Kết quả huấn luyện từ mô hình LSTM:

A screenshot of a computer screen

AI-generated content may be incorrect.

Hình 3. 14. Kết quả hiệu năng khi được huấn luyện bằng mô hình LSTM

A blue squares with numbers and labels

AI-generated content may be incorrect.

Hình 3. 15. Ma trận nhầm lẫn khi được huấn luyện bằng mô hình LSTM

* + - 1. Độ chính xác tổng thể (Accuracy)

|  |  |
| --- | --- |
| Mô hình | Accuracy |
| LSTM | 0.98 |
| Transformer | 0.9956 |

Bảng 3. 3. So sánh độ chính xác giữa mô hình Transformer và LSTM

* + - 1. Chỉ số Precision, Recall, F1-score

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Mô hình | Lớp | Precision | Recall | F1-score |
| LSTM | 0 | 0.99 | 0.98 | 0.98 |
|  | 1 | 0.98 | 0.99 | 0.98 |
| Transformer | nAMP (0) | 1.00 | 0.99 | 1.00 |
|  | AMP (1) | 0.99 | 1.00 | 1.00 |

Bảng 3. 4. So sánh chỉ số Precision, Recall, F1-score giữa mô hình Transformer và LSTM

* + - 1. Confusion Matrix (Ma trận nhầm lẫn)

|  |  |  |
| --- | --- | --- |
| Mô hình LSTM | Predicted 0 | Predicted 1 |
| Actual 0 | 19553 | 421 |
| Actual 1 | 237 | 19787 |
| Tổng lỗi: 421 (FP) + 237 (FN) = 658 lỗi | | |

Bảng 3. 5. Confusion Matrix của mô hình LSTM

|  |  |  |
| --- | --- | --- |
| Mô hình Transformer | Predicted nAMP | Predicted AMP |
| Actual nAMP | 171557 | 943 |
| Actual AMP | 522 | 165893 |
| Tổng lỗi: 943 (FP) + 522 (FN) = 1465 lỗi | | |

Bảng 3. 6. Confusion Matrix của mô hình Transformer

Lưu ý: Dù Transformer có số lỗi tuyệt đối cao hơn (do dataset lớn hơn gấp ~8.5 lần), tỉ lệ lỗi vẫn cực kỳ nhỏ, dẫn đến độ chính xác gần tuyệt đối.

* + - 1. Kết luận so sánh

Qua quá trình huấn luyện và đánh giá trên tập dữ liệu phân loại peptide AMP, mô hình Transformer cho thấy hiệu suất vượt trội so với LSTM. Cụ thể, Transformer đạt độ chính xác gần như tuyệt đối (99.56%) với F1-score xấp xỉ 1.00 cho cả hai lớp, trong khi LSTM đạt độ chính xác 98% và F1-score là 0.98. Mặc dù Transformer xử lý một tập dữ liệu lớn hơn gấp nhiều lần so với LSTM, nó vẫn duy trì hiệu quả phân loại rất cao, thể hiện khả năng học và tổng quát hóa mạnh mẽ hơn. Ngược lại, LSTM vẫn cho kết quả tốt trên tập dữ liệu nhỏ hơn, phù hợp với các bài toán có tài nguyên hạn chế. Như vậy, nếu yêu cầu cao về độ chính xác và xử lý dữ liệu lớn, Transformer là lựa chọn ưu việt; còn trong môi trường hạn chế tài nguyên, LSTM vẫn là một mô hình hiệu quả và đáng tin cậy.

* + 1. ***Phân tích khả năng giải thích của mô hình qua Explainable AI SHAP***

A graph with numbers and symbols

AI-generated content may be incorrect.

Hình 3. 16. Kết quả giải thích của SHAP trên một mẫu peptide được dự đoán

Biểu đồ SHAP dạng Waterfall cho mẫu peptide số 1 đã cung cấp những thông tin quý giá về quá trình ra quyết định của mô hình. Kết quả cho thấy giá trị dự đoán ban đầu của mô hình là . Sau khi tính toán ảnh hưởng của từng đặc trưng riêng lẻ, giá trị đầu ra cuối cùng là , nghiêng rõ rệt về hướng phân loại mẫu này thuộc nhóm NAMP (class 0).

Đặc trưng quan trọng ảnh hưởng đến quyết định mô hình, ừ kết quả phân tích SHAP, em đã xác định được những đặc trưng có tác động mạnh nhất đến quyết định phân loại của mô hình:

Các đặc trưng thúc đẩy hướng phân loại AMP (class 1): text\_feat\_6, text\_feat\_7 và num\_feat\_47 (mỗi đặc trưng đóng góp khoảng +0.01 vào xác suất AMP)

Các đặc trưng thúc đẩy hướng phân loại NAMP (class 0): text\_feat\_3, text\_feat\_8, text\_feat\_0, text\_feat\_4, num\_feat\_108 và num\_feat\_41 (các đặc trưng này đẩy mạnh về hướng phân loại NAMP)

A close-up of a computer code

AI-generated content may be incorrect.

Hình 3. 17. Kết quả giải thích được đặc trưng quan trọng của mẫu peptide

Ánh xạ sinh học: Các vị trí text\_namp = [4, 0, 8, 3, 7, 6] tương ứng với các amino acid như glycine (g), threonine (t), leucine (l), serine (s) và valine (v). Đây là những acid amin thường xuất hiện với tần suất cao trong các peptide không kháng khuẩn (NAMP).

Phương pháp SHAP không chỉ tăng tính minh bạch của mô hình mà còn cung cấp những hiểu biết sinh học có giá trị, điều rất quan trọng khi áp dụng các mô hình học máy trong lĩnh vực y sinh và dược phẩm. Từ đó, em có thể tự tin hơn khi triển khai mô hình vào các ứng dụng thực tế trong phát hiện và thiết kế các peptide kháng khuẩn mới.

* + 1. ***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 IV: KẾT LUẬN**

Qua quá trình nghiên cứu và thực hiện đồ án “Ứng dụng học sâu và Explainable Artificial Intelligence trong phân tích đặc trưng và thiết kế 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. 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 axit amin 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 SHAP 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 SHAP để phân tích kết quả đầu ra. 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.

# **TÀI LIỆU THAM KHẢO**

**Tiếng Việt**

[1] SuNT. “Self-Attention và Multi-head Sefl-Attention trong kiến trúc Transformer.” <https://tiensu.github.io/blog/59_self-attention/>.

[2] SuNT. “Transformers “ <https://tiensu.github.io/blog/60_transformer/>.

**Tiếng Anh**

[3] “Database resources of the national center for biotechnology information,” (in English), *Nucleic acids research,* vol. 46, no. D1, pp. D8-D13, 2018.

[4] “Shapley Additive Explanations.” <https://serp.ai/posts/shapley-additive-explanations/>.

[5] Y. Bao *et al.*, “National center for biotechnology information viral genomes project,” (in English), *Journal of virology,* vol. 78, no. 14, pp. 7291-7298, 2004.

[6] A. Chandra, L. Tünnermann, T. Löfstedt, and R. Gratz, “Transformer-based deep learning for predicting protein properties in the life sciences,” (in English), *Elife,* vol. 12, p. e82819, 2023.

[7] M. Christoph, “Interpretable machine learning: A guide for making black box models explainable,” (in English), 2020.

[8] S. Garg, D. Tsipras, P. S. Liang, and G. Valiant, “What can transformers learn in-context? a case study of simple function classes,” (in English), *Advances in Neural Information Processing Systems,* vol. 35, pp. 30583-30598, 2022.

[9] M. M. Islam, G. Aguilar, P. Ponnusamy, C. S. Mathialagan, C. Ma, and C. Guo, “A vocabulary-free multilingual neural tokenizer for end-to-end task learning,” (in English), *arXiv preprint arXiv:2204.10815,* 2022.

[10] A. Jain, A. Fandango, and A. Kapoor, *TensorFlow Machine Learning Projects: Build 13 real-world projects with advanced numerical computations using the Python ecosystem*. Packt Publishing Ltd (in English), 2018.

[11] P. S. Kalchuri. “Anaconda Navigator An Overview.” <https://medium.com/analytics-vidhya/anaconda-navigator-an-overview-4e5d27ca8047>.

[12] P. Kalpana, R. Anandan, A. G. Hussien, H. Migdady, and L. Abualigah, “Plant disease recognition using residual convolutional enlightened Swin transformer networks,” (in English), *Scientific Reports,* vol. 14, no. 1, p. 8660, 2024.

[13] U. Kamath and J. Liu, “Explainable artificial intelligence: an introduction to interpretable machine learning,” (in English), 2021.

[14] masuzi. “What Is The Longest Open Reading Frame Orf In Following Dna Sequence.”.

[15] S. Mathew and J. Varia, “Overview of amazon web services,” (in English), *Amazon Whitepapers,* vol. 105, no. 1, p. 22, 2014.

[16] N. A. O'Leary *et al.*, “Reference sequence (RefSeq) database at NCBI: current status, taxonomic expansion, and functional annotation,” (in English), *Nucleic acids research,* vol. 44, no. D1, pp. D733-D745, 2016.

[17] M. D. Oleg A. Shchelochkov. “​Open Reading Frame.” <https://www.genome.gov/genetics-glossary/Open-Reading-Frame>.

[18] rennylee. “NATIONAL CENTER FOR BIOTECHNOLOGY INFORMATION: AN OVERVIEW.” <https://www.scq.ubc.ca/national-center-for-biotechnology-information-an-overview/#:~:text=The%20National%20Center%20for%20Biotechnology%20Information%20%28NCBI%29%20is,%28NIH%29%20and%20the%20National%20Library%20of%20Medicine%20%28NLM%29>..

[19] N. Sharma, L. D. Naorem, S. Jain, and G. P. Raghava, “ToxinPred2: an improved method for predicting toxicity of proteins,” (in English), *Briefings in bioinformatics,* vol. 23, no. 5, p. bbac174, 2022.

[20] N. Taniguchi, “Amino acids and proteins,” (in English), *Medical Biochemistry, Ed,* vol. 3, pp. 5-21, 2010.

[21] V. Thumuluri, J. J. Almagro Armenteros, A. R. Johansen, H. Nielsen, and O. Winther, “DeepLoc 2.0: multi-label subcellular localization prediction using protein language models,” (in English), *Nucleic acids research,* vol. 50, no. W1, pp. W228-W234, 2022.

[22] G. Van Rossum and F. L. Drake, *An introduction to Python*. Network Theory Ltd. Bristol (in English), 2003.

[23] H. Yang *et al.*, “RNA-binding protein recognition based on multi-view deep feature and multi-label learning,” (in English), *Briefings in Bioinformatics,* vol. 22, no. 3, p. bbaa174, 2021.

[24] C. Yeh, Y. Chen, A. Wu, C. Chen, F. Viégas, and M. Wattenberg, “AttentionViz: A Global View of Transformer Attention (arXiv: 2305.03210). arXiv,” ed, 2023.

[25] M. R. Zafar and N. Khan, “Deterministic local interpretable model-agnostic explanations for stable explainability,” (in English), *Machine Learning and Knowledge Extraction,* vol. 3, no. 3, pp. 525-541, 2021.

[26] H. Zhang *et al.*, “Deep learning-based bioactive therapeutic peptide generation and screening,” (in English), *Journal of Chemical Information and Modeling,* vol. 63, no. 3, pp. 835-845, 2023.

[27] L. Zhao, J. Gao, D. Deng, and X. Li, “SSIR: Spatial shuffle multi-head self-attention for single image super-resolution,” (in English), *Pattern Recognition,* vol. 148, p. 110195, 2024.