# **Mục lục**

[**Mục lục** 1](#_Toc192969543)

[**Chương I: Giới thiệu** 1](#_Toc192969544)

[**Chương II: Cơ sở lý thuyết và công nghệ liên quan** 2](#_Toc192969545)

[2.1. **Tổng quan về xử lý ngôn ngữ tự nhiên** 2](#_Toc192969546)

[2.2. **Công nghệ và mô hình áp dụng** 3](#_Toc192969547)

[2.2.1 Tổng quan về lý thuyết và mô hình liên quan 3](#_Toc192969548)

[2.2.1.1 Mạng CNN 3](#_Toc192969549)

[2.2.1.1.1 **Cấu trúc tổng quan của CNN** 3](#_Toc192969550)

[2.2.2.1.2 Các thành phần chính trong CNN 4](#_Toc192969551)

[2.2.1.2 Kiến trúc Transformer 11](#_Toc192969552)

[2.2.2.2.1 Encoder 13](#_Toc192969553)

[2.2.2.2.1.1 Đầu vào của transformer 13](#_Toc192969554)

[2.2.2.2.1.2 Multi-Head Attention 15](#_Toc192969555)

[2.2.2.2.1.3 Feed-Forward Networks 18](#_Toc192969556)

[2.2.2.2.2 Decoder 19](#_Toc192969557)

[2.2.2.2.2.1 Masked Multi-Head Attention 19](#_Toc192969558)

[2.2.2.2.2.2 Cross-Attention 21](#_Toc192969559)

[2.2.2.2.3 Tổng kết kiến trúc 22](#_Toc192969560)

[**Chương III : Phương pháp tiếp cận và giải pháp giải quyết bài toán** 23](#_Toc192969561)

[3.1 Phương pháp tiếp cận 23](#_Toc192969562)

[3.2 Phương pháp sử dụng 26](#_Toc192969563)

[3.2.1 Kiến trúc tổng quan 26](#_Toc192969564)

[3.2.1.1 Xử lý hai nhánh riêng hình ảnh và âm thanh 26](#_Toc192969565)

Lời cảm ơn

Tóm tắt nội dung

# **Chương I: Giới thiệu**

1.1. **Lý do chọn đề tài**  
1.2. **Mục tiêu nghiên cứu**  
1.3. **Phạm vi nghiên cứu**  
1.4. **Ý nghĩa thực tiễn của đề tài**

# **Chương II: Cơ sở lý thuyết và công nghệ liên quan**

## 2.1. **Tổng quan về xử lý ngôn ngữ tự nhiên**

**Xử lý ngôn ngữ tự nhiên (NLP)** là một lĩnh vực trong trí tuệ nhân tạo (AI) và khoa học máy tính, tập trung vào việc tương tác giữa máy tính và ngôn ngữ con người. Mục tiêu chính của NLP là giúp máy tính hiểu, phân tích, và tạo ra ngôn ngữ tự nhiên theo cách tương tự như con người. Các ứng dụng của NLP xuất hiện ở khắp mọi nơi trong đời sống hiện đại, từ dịch máy, nhận diện giọng nói, chatbot, phân tích cảm xúc, cho đến tóm tắt văn bản tự động.

Trước năm 2024, NLP đã đạt được nhiều thành tựu đáng chú ý nhờ sự phát triển của các mô hình học sâu, đặc biệt là các mô hình dựa trên Transformer như BERT, GPT, và T5. Các nghiên cứu này không chỉ cải thiện khả năng xử lý văn bản mà còn mở rộng sang các lĩnh vực như nhận diện giọng nói và xử lý đa phương thức. BERT (Bidirectional Encoder Representations from Transformers) đã cách mạng hóa NLP bằng cách áp dụng cơ chế Attention để học ngữ cảnh từ cả hai chiều trái và phải của văn bản. GPT (Generative Pre-trained Transformer) tập trung vào việc tạo văn bản tự nhiên với chất lượng cao, trong khi T5 (Text-to-Text Transfer Transformer) thống nhất các tác vụ NLP dưới dạng bài toán "văn bản-đến-văn bản". Ngoài ra, các nghiên cứu về xử lý giọng nói như DeepSpeech, Kaldi, và các hệ thống nhận diện giọng nói của Google đã giúp nâng cao độ chính xác trong nhận diện âm thanh.

Một tác vụ nổi bật trong xử lý ngôn ngữ tự nhiên là **nhận diện giọng nói tự động (ASR - Automatic Speech Recognition)**, nơi máy tính chuyển đổi âm thanh thành văn bản. Đây là một trong những ứng dụng cốt lõi của NLP, giúp máy móc có thể hiểu và xử lý ngôn ngữ nói của con người. Whisper, một mô hình được phát triển bởi OpenAI, đại diện cho một bước tiến lớn trong tác vụ này. Whisper không chỉ có khả năng chuyển đổi giọng nói thành văn bản với độ chính xác cao mà còn hoạt động tốt trong nhiều ngôn ngữ và trong các điều kiện âm thanh không lý tưởng, chẳng hạn như môi trường có tạp âm hoặc giọng nói không rõ ràng. Ngoài ra, Whisper còn hỗ trợ các tác vụ bổ sung như đồng bộ hóa phụ đề, dịch ngôn ngữ, và phân tích âm thanh, làm nổi bật tính linh hoạt và ứng dụng rộng rãi của nó trong lĩnh vực NLP.

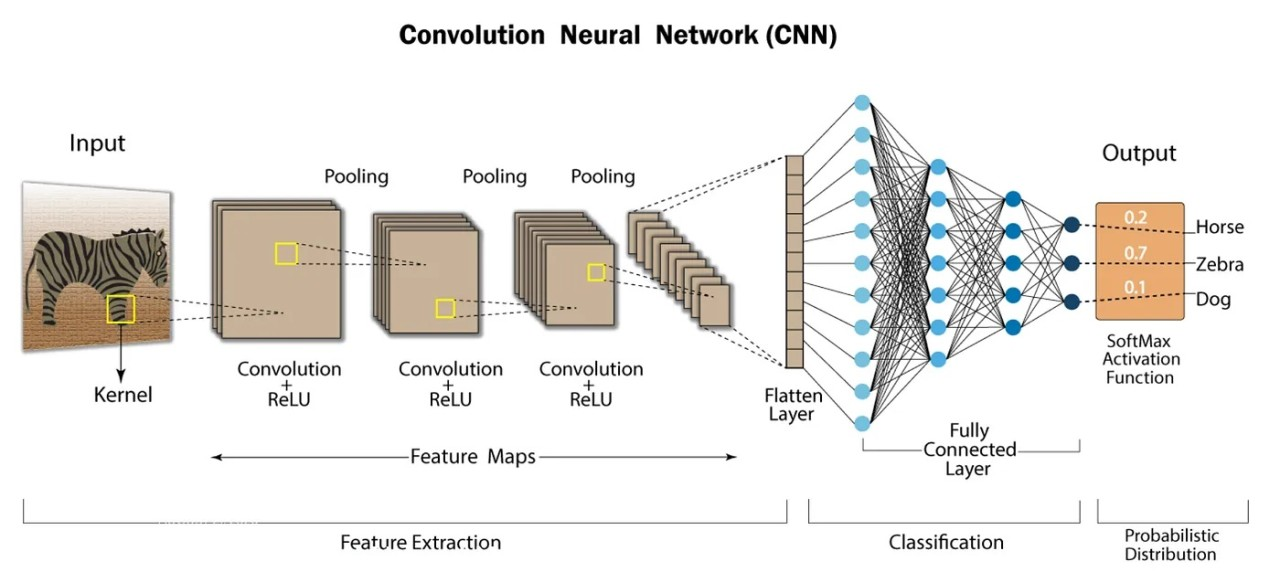
Những tiến bộ vượt bậc trong xử lý ngôn ngữ tự nhiên, bao gồm cả các ứng dụng như Whisper, không chỉ cải thiện khả năng tương tác giữa con người và máy tính mà còn mở ra nhiều tiềm năng mới trong các lĩnh vực như giáo dục, chăm sóc sức khỏe, và truyền thông. NLP đã và đang trở thành nền tảng quan trọng thúc đẩy sự phát triển của AI hiện đại.

## 2.2. **Công nghệ và mô hình áp dụng**

### 2.2.1 Mạng CNN

Convolutional Neural Network (CNN) là một loại mạng thần kinh nhân tạo đặc biệt được thiết kế để xử lý dữ liệu dạng lưới, chẳng hạn như hình ảnh, video. CNN được sử dụng rộng rãi trong nhận dạng hình ảnh, phát hiện vật thể, phân loại hình ảnh, và nhiều ứng dụng khác trong thị giác máy tính. Dưới đây là tổng quan đầy đủ về các thành phần và các lớp chính của CNN.

#### 2.2.1.1 **Cấu trúc tổng quan của CNN**



Cấu trúc của một mạng CNN

CNN thường bao gồm ba loại lớp chính:

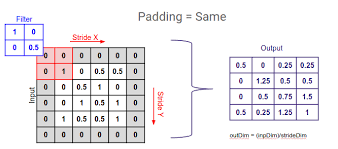
1. **Lớp tích chập (Convolutional Layer)** - Trích xuất đặc trưng từ dữ liệu đầu vào.
2. **Lớp gộp (Pooling Layer)** - Giảm kích thước không gian của đặc trưng, giúp giảm tính toán và tránh overfitting.
3. **Lớp kết nối đầy đủ (Fully Connected Layer)** - Kết hợp đặc trưng để đưa ra dự đoán cuối cùng.

Ngoài ra, CNN còn sử dụng các lớp hỗ trợ như:

* Lớp kích hoạt (Activation Layer) để tạo phi tuyến tính.
* Lớp chuẩn hóa (Normalization Layer) để ổn định và tăng tốc độ huấn luyện.

#### **2.2.1.2 Các thành phần chính trong CNN**

**Lớp Tích Chập (Convolutional Layer)**



**Phép tích chập**

Lớp tích chập là nền tảng chính của CNN. Nó thực hiện phép tích chập (convolution) giữa đầu vào và một tập các bộ lọc (filter/kernel).

Thành phần chính:

- Kernel (bộ lọc): Ma trận nhỏ có kích thước k × k (ví dụ: 3 × 3, 5 × 5), trích xuất đặc trưng cụ thể như cạnh, góc, hoặc hoa văn.

- Stride: Bước nhảy khi kernel di chuyển trên đầu vào. Stride lớn giúp giảm kích thước đầu ra.

- Padding: Thêm các giá trị (thường là 0) xung quanh đầu vào để duy trì kích thước không gian sau tích chập.

- Output feature map: Đầu ra của phép tích chập, chứa thông tin đặc trưng đã được trích xuất.

Công thức tính kích thước đầu ra:

O = (I - K + 2P) / S + 1

- I: Kích thước đầu vào.  
- K: Kích thước kernel.  
- P: Padding.  
- S: Stride.

Output của phép tích chập là một **feature map** (bản đồ đặc trưng), mang ý nghĩa biểu diễn các đặc trưng đã được trích xuất từ dữ liệu đầu vào. Mỗi giá trị trong feature map đại diện cho mức độ mà một đặc trưng cụ thể, chẳng hạn như cạnh, góc, hoặc họa tiết, xuất hiện tại một vị trí tương ứng trên dữ liệu đầu vào. Feature map nắm bắt được thông tin cục bộ quan trọng, đồng thời làm giảm nhiễu và thông tin không cần thiết.

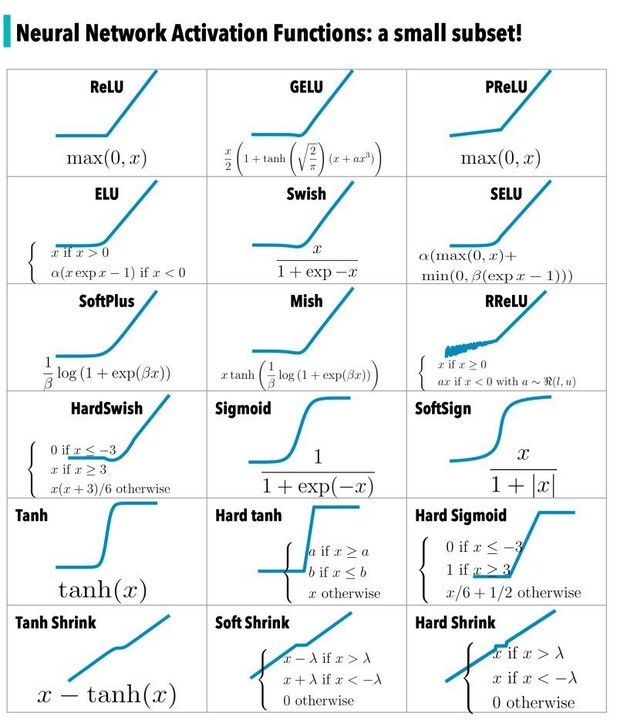
Ví dụ, trong bài toán nhận diện khuôn mặt, các lớp tích chập ban đầu có thể phát hiện cạnh và góc, trong khi các lớp sau cùng xác định mắt, mũi, hoặc các đặc điểm nhận dạng khác. Nhờ đó, feature map giúp CNN nắm bắt thông tin cần thiết để đưa ra dự đoán chính xác, đồng thời giảm bớt sự dư thừa trong dữ liệu đầu vào.

**Lớp Kích Hoạt (Activation Layer)**

Lớp kích hoạt trong mạng nơ-ron đóng vai trò quan trọng trong việc thêm tính phi tuyến tính vào mạng, giúp mô hình học được các đặc trưng phức tạp hơn từ dữ liệu. Nếu không có hàm kích hoạt, mạng chỉ thực hiện các phép tính tuyến tính, khiến nó không thể giải quyết các bài toán phức tạp như phân loại hoặc nhận diện mẫu phức tạp trong dữ liệu. Một số hàm kích hoạt phổ biến được sử dụng bao gồm:

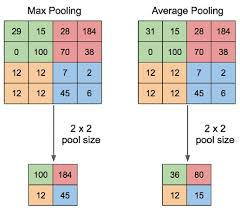
ReLU (Rectified Linear Unit), được định nghĩa là ReLU(x) = max(0, x), là một hàm kích hoạt đơn giản nhưng hiệu quả, giúp mô hình hội tụ nhanh hơn trong quá trình huấn luyện. Sigmoid là một hàm chuyển đầu ra về khoảng [0, 1], thích hợp cho các bài toán yêu cầu xác suất, chẳng hạn như phân loại nhị phân. Tanh, chuyển đầu ra về khoảng [-1, 1], thường được sử dụng khi dữ liệu có cả giá trị âm và dương, vì nó tạo ra sự đối xứng và cải thiện việc hội tụ. Leaky ReLU là một biến thể của ReLU, giải quyết vấn đề chết neuron (neuron output luôn bằng 0) bằng cách thêm một gradient nhỏ khi giá trị đầu vào nhỏ hơn 0, giúp các neuron vẫn học được trong các trường hợp đặc biệt.

Hàm kích hoạt mang lại nhiều lợi ích, chẳng hạn trong bài toán phân loại ảnh, ReLU thường được sử dụng vì tính đơn giản và hiệu quả của nó khi làm việc với các đặc trưng cục bộ trong hình ảnh. Trong các bài toán như dự đoán xác suất mắc bệnh dựa trên các đặc điểm sinh học, hàm Sigmoid là sự lựa chọn lý tưởng vì nó trả về kết quả trực quan trong khoảng xác suất từ 0 đến 1. Đối với bài toán phân tích chuỗi tài chính, hàm Tanh giúp phản ánh rõ ràng cả giá trị tăng và giảm trong dữ liệu, tạo điều kiện cho mô hình hiểu rõ hơn về các biến động. Leaky ReLU lại rất hữu ích trong các mạng sâu, nơi gradient vanishing có thể xảy ra, giúp đảm bảo rằng mọi neuron đều đóng góp vào quá trình học. Nhờ các hàm kích hoạt, mạng nơ-ron có thể học được các quan hệ phức tạp trong dữ liệu, từ đó giải quyết hiệu quả nhiều bài toán thực tế.



**Một số hàm kích hoạt**

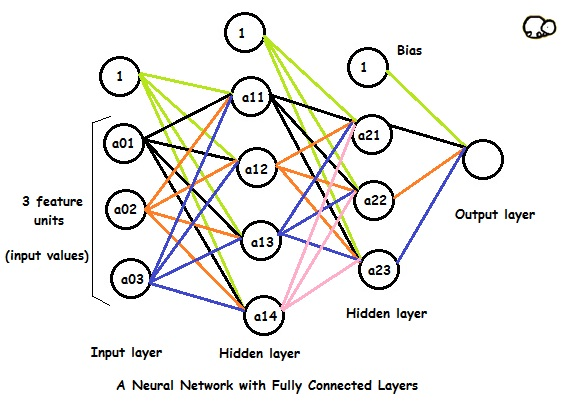
**Lớp Gộp (Pooling Layer)**



**Minh họa phương pháp gộp nhằm giảm kích thước ma trận**

Lớp gộp giảm kích thước không gian của feature map, giúp giảm tính toán và tăng khả năng tổng quát hóa của mô hình.  
Các loại gộp:  
- Max Pooling: Lấy giá trị lớn nhất trong vùng kernel.  
- Average Pooling: Lấy giá trị trung bình của các phần tử trong vùng kernel.  
- Global Pooling: Lấy giá trị max hoặc trung bình trên toàn bộ feature map, thường được sử dụng trước lớp fully connected.

**Lớp Kết Nối Đầy Đủ (Fully Connected Layer - FC)**



**Mạng neural phân loại điển hình**

Lớp Fully Connected (FC) trong mạng CNN là một thành phần quan trọng, đảm nhiệm vai trò tổng hợp các đặc trưng đã được trích xuất từ các lớp tích chập (convolutional layers) và gộp (pooling layers), sau đó sử dụng chúng để đưa ra dự đoán cuối cùng. Input của lớp FC thường là một vector một chiều (1D) được tạo ra bằng cách "trải phẳng" (flatten) feature map ba chiều (3D) từ lớp trước đó. Vector này chứa các giá trị đại diện cho các đặc trưng cốt lõi đã được trích xuất từ đầu vào, chẳng hạn như các đường viền, hình dạng, hoặc các yếu tố phức tạp hơn tùy thuộc vào mức độ sâu của mạng.

Khi đi qua lớp FC, mỗi giá trị trong vector đầu vào được kết nối với mọi neuron trong lớp tiếp theo thông qua các trọng số. Lớp FC thực hiện phép nhân ma trận giữa vector đầu vào và ma trận trọng số, sau đó cộng thêm bias (hệ số bù). Kết quả sau đó được đưa qua hàm kích hoạt, chẳng hạn như ReLU hoặc Sigmoid, để thêm tính phi tuyến tính, giúp mô hình học được các mối quan hệ phức tạp hơn giữa các đặc trưng.

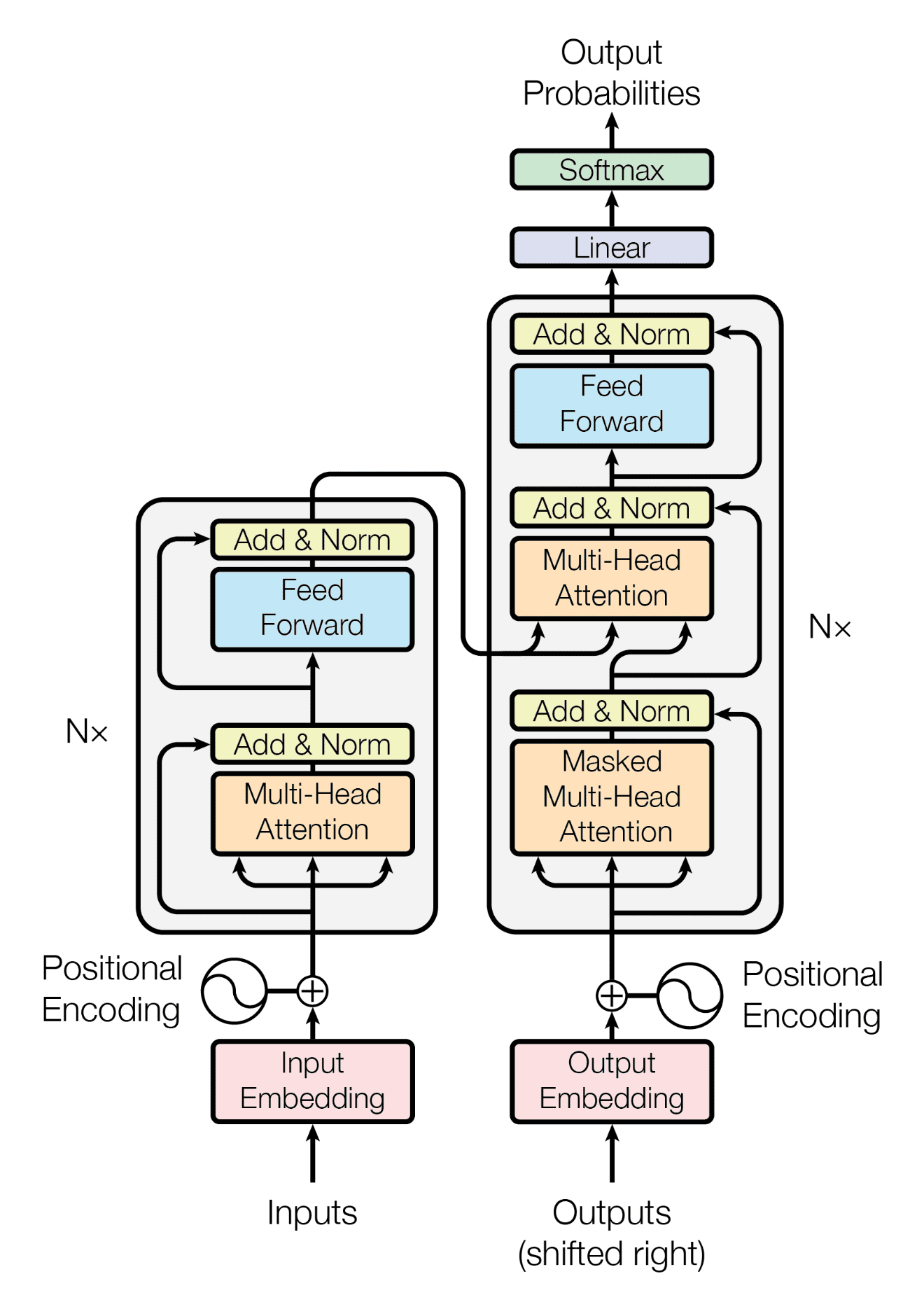
Sau khi qua lớp FC, đầu ra thường là một vector với kích thước bằng số lớp (categories) mà mô hình cần phân loại. Ví dụ, trong bài toán phân loại ảnh có 10 lớp (như chữ số từ 0 đến 9), lớp FC cuối cùng sẽ tạo ra một vector 10 giá trị, trong đó mỗi giá trị biểu thị xác suất thuộc về từng lớp. Các giá trị này thường được đưa qua hàm softmax để chuẩn hóa thành xác suất.

**Cách lớp FC cải thiện mô hình**:

1. **Tích hợp đặc trưng**: Lớp FC tổng hợp các đặc trưng từ toàn bộ ảnh, giúp mô hình hiểu được bức tranh tổng thể thay vì chỉ các đặc trưng cục bộ.
2. **Học quan hệ phi tuyến**: Nhờ vào việc sử dụng trọng số và hàm kích hoạt, lớp FC có thể học được các quan hệ phức tạp giữa các đặc trưng, từ đó cải thiện độ chính xác của mô hình.
3. **Phân loại**: Đây là nơi diễn ra việc ánh xạ các đặc trưng đã học thành các nhãn cụ thể, đóng vai trò như một lớp phân loại trong các bài toán classification.
4. **Feedback và Backpropagation**: Trong quá trình huấn luyện, gradient của hàm lỗi được lan truyền ngược qua lớp FC để cập nhật trọng số, từ đó tối ưu hóa khả năng phân loại của mô hình.

Lớp FC không chỉ kết nối toàn bộ các neuron mà còn đóng vai trò quyết định trong việc chuyển đổi từ các đặc trưng đã học sang kết quả cuối cùng có ý nghĩa, giúp CNN đạt hiệu suất cao trong các bài toán nhận diện và phân loại.

### 2.2.2 Kiến trúc Transformer



Kiên trúc transfomer

Transformer được giới thiệu vào năm 2017 qua bài báo "Attention is All You Need" của Vaswani và cộng sự như một bước đột phá trong lĩnh vực xử lý ngôn ngữ tự nhiên (NLP) và các chuỗi dữ liệu. Transformer ra đời để giải quyết những hạn chế quan trọng của các kiến trúc truyền thống như Recurrent Neural Networks (RNN) và Long Short-Term Memory (LSTM), vốn đã từng là nền tảng của các bài toán chuỗi. Những hạn chế chính của RNN và LSTM bao gồm việc mất thông tin khi xử lý các chuỗi dài và sự chậm chạp trong quá trình huấn luyện do tính tuần tự.

Vấn đề mất thông tin trong RNN xuất phát từ hiện tượng **vanishing gradient**, một thách thức lớn khi sử dụng hàm kích hoạt phi tuyến trong mạng nhiều lớp. Trong các mô hình RNN, mỗi bước thời gian mới phụ thuộc vào thông tin được truyền từ bước trước đó. Khi chuỗi dữ liệu trở nên dài, thông tin ở các bước đầu dần bị "phai nhạt" qua nhiều lần tính toán. Kết quả là, các thông tin quan trọng ở xa đầu chuỗi có thể bị lãng quên hoặc không được mô hình khai thác đầy đủ, dẫn đến việc giảm hiệu suất khi phải xử lý các chuỗi dài.

Mặc dù LSTM được thiết kế để khắc phục hiện tượng vanishing gradient nhờ vào cơ chế các cổng (gates) như cổng đầu vào, cổng đầu ra, và cổng quên, nhưng nó vẫn gặp giới hạn khi chuỗi quá dài hoặc khi dữ liệu yêu cầu mô hình phải nhớ các quan hệ phức tạp giữa các thành phần cách xa nhau. Việc lưu trữ và quản lý thông tin qua quá nhiều bước thời gian không chỉ phức tạp mà còn đòi hỏi tài nguyên lớn.

Ngoài ra, cả RNN và LSTM đều huấn luyện một cách **tuần tự**, nghĩa là mỗi trạng thái hiện tại phải được tính toán dựa trên trạng thái trước đó. Điều này khiến các mô hình này không thể tận dụng được lợi thế của việc song song hóa tính toán trên các GPU hoặc TPU hiện đại. Kết quả là thời gian huấn luyện kéo dài đáng kể, làm chậm quá trình phát triển mô hình và tăng chi phí tính toán.

Transformer ra đời với một thiết kế khác biệt, không dựa trên tính tuần tự mà sử dụng cơ chế **Attention**, đặc biệt là Self-Attention, để mô hình hóa mối quan hệ giữa mọi phần tử trong chuỗi đầu vào một cách trực tiếp. Cơ chế này cho phép Transformer xử lý chuỗi dữ liệu dài mà không mất thông tin, đồng thời tận dụng khả năng song song hóa để tăng tốc độ huấn luyện đáng kể. Sự đổi mới này đã mở ra một kỷ nguyên mới trong NLP và các lĩnh vực liên quan, định hình lại cách tiếp cận với các bài toán chuỗi dữ liệu.

Transformer bao gồm hai thành phần chính:

**Bộ mã hóa (Encoder)** và **Bộ giải mã (Decoder)**, được thiết kế để xử lý và biến đổi chuỗi dữ liệu một cách hiệu quả. Mỗi thành phần được xây dựng từ các khối cơ bản, trong đó **Multi-Head Attention** là cốt lõi. Thành phần này cho phép mô hình học được các mối quan hệ ngữ cảnh giữa các phần tử trong chuỗi bằng cách sử dụng nhiều "đầu chú ý" (attention heads) để nắm bắt các kiểu thông tin khác nhau. **Feed-Forward Networks (FFN)**, một mạng thần kinh đơn giản, được áp dụng sau Multi-Head Attention để tăng cường khả năng trích xuất đặc trưng.

Để duy trì sự ổn định trong quá trình huấn luyện, Transformer sử dụng **Residual Connections** (kết nối dư) để thêm đầu vào ban đầu vào đầu ra của mỗi khối, kết hợp với **Layer Normalization** để chuẩn hóa các giá trị, giúp tăng tốc độ hội tụ. Một thành phần quan trọng khác là **Positional Encoding**, được thêm vào các vector đầu vào để cung cấp thông tin về vị trí của các phần tử trong chuỗi, khắc phục việc thiếu thông tin thứ tự trong kiến trúc. Bộ mã hóa chịu trách nhiệm trích xuất và nén thông tin từ đầu vào, trong khi bộ giải mã sử dụng thông tin này để tạo ra chuỗi đầu ra, thường kết hợp với cơ chế Attention chéo (Cross-Attention) để học cách ánh xạ từ đầu vào sang đầu ra một cách hiệu quả.

#### 2.2.2.1 Encoder

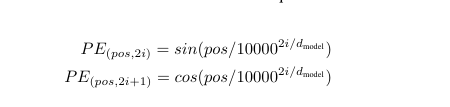
##### 2.2.2.1.1 Đầu vào của transformer

Đầu vào của Transformer là một chuỗi các câu, trong đó mỗi từ hoặc token được ánh xạ thành một vector biểu diễn bằng cách sử dụng **Embedding Layer**. Lớp embedding chuyển đổi các token—thường được mã hóa thành các số nguyên thông qua các bộ từ điển hoặc tokenizer—thành các vector số học trong một không gian nhiều chiều. Các vector này được thiết kế để biểu diễn ý nghĩa ngữ nghĩa của các token, trong đó các từ có ý nghĩa tương tự sẽ nằm gần nhau trong không gian vector. Quá trình embedding thường được thực hiện bằng cách tra cứu trong một ma trận embedding được học trong quá trình huấn luyện, trong đó mỗi hàng của ma trận tương ứng với vector biểu diễn của một từ trong từ vựng. Vector embedding không chỉ giúp giảm kích thước biểu diễn mà còn giữ lại các đặc trưng quan trọng, giúp mô hình hiểu được các mối quan hệ ngữ nghĩa giữa các từ.



Minh họa cho embedding một câu

Tuy nhiên, bản thân các vector embedding không chứa thông tin về vị trí của các token trong chuỗi, điều này khiến Transformer cần một cơ chế để bổ sung thông tin thứ tự, đó là **Positional Encoding**. Positional Encoding là một phương pháp đặc biệt, trong đó thông tin về vị trí của mỗi token được mã hóa và cộng trực tiếp vào các vector embedding. Cơ chế này sử dụng các hàm toán học sin và cos với các tần số khác nhau để tạo ra một biểu diễn vị trí độc nhất cho từng token. Cụ thể, đối với mỗi vị trí pospos và mỗi chiều ii trong vector embedding, giá trị Positional Encoding được tính như sau:



**Công thức tham chiếu cách tính position encoding trong paper**

Hàm sin được áp dụng cho các chiều chẵn, và hàm cos được áp dụng cho các chiều lẻ, giúp tạo ra một biểu diễn tuần hoàn có khả năng mã hóa vị trí tương đối giữa các token.

Việc sử dụng hàm sin và cos có hai lợi ích quan trọng:

thứ nhất, nó đảm bảo rằng các vị trí gần nhau trong chuỗi sẽ có biểu diễn tương tự, giúp mô hình dễ dàng học được các mối quan hệ cục bộ;

thứ hai, các giá trị này có tính tuần hoàn, cho phép mô hình học được các mối quan hệ dài hạn trong chuỗi. Sau khi tính toán, vector Positional Encoding được cộng trực tiếp vào vector embedding của mỗi token.

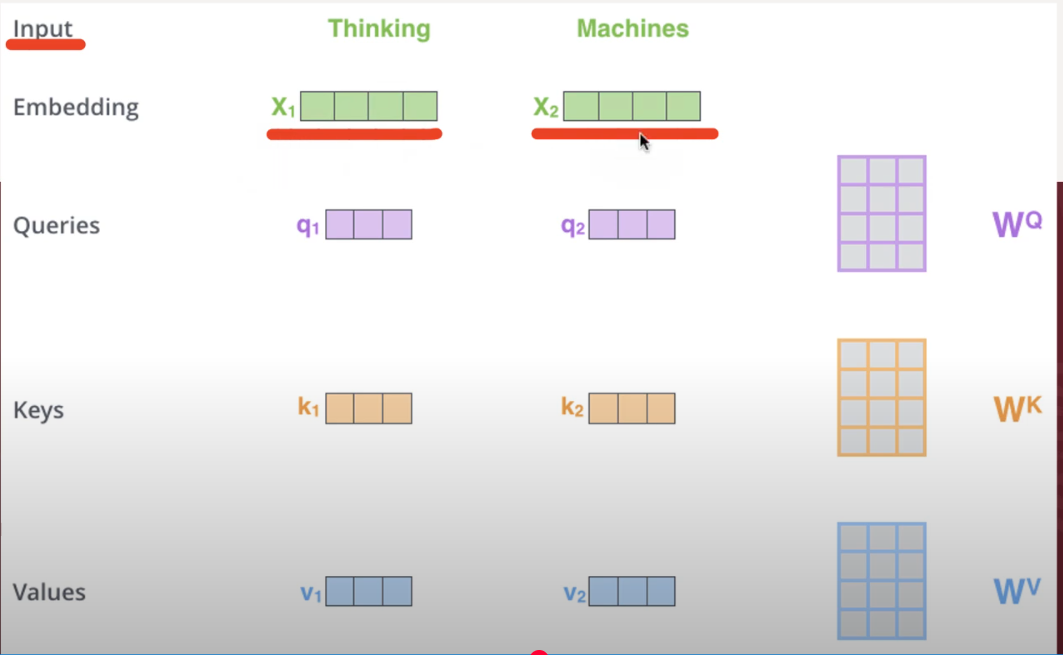
Kết quả là một chuỗi vector kết hợp cả thông tin ngữ nghĩa (từ embedding) và thông tin vị trí (từ Positional Encoding), tạo điều kiện cho cơ chế Attention trong Transformer xử lý hiệu quả mối quan hệ giữa các token trong toàn chuỗi, bất kể khoảng cách giữa chúng. Điều này giúp Transformer có thể nắm bắt được cả ngữ nghĩa của từ và ngữ cảnh toàn cục trong chuỗi một cách linh hoạt và mạnh mẽ.

##### 2.2.2.1.2 Multi-Head Attention

Multi-Head Attention là thành phần cốt lõi trong kiến trúc **Encoder** của Transformer, chịu trách nhiệm học và nắm bắt mối quan hệ giữa các token trong chuỗi đầu vào bất kể khoảng cách giữa chúng. Thành phần này được xây dựng dựa trên cơ chế **Self-Attention**, trong đó mỗi token trong chuỗi "tập trung" vào các token khác để học cách chúng liên kết với nhau, giúp mô hình hiểu được ngữ cảnh toàn cục.

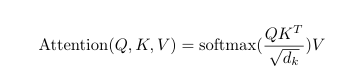
Input đầu tiên của Multi-Head Attention là một chuỗi các vector đã được kết hợp giữa **Embedding** và **Positional Encoding**, với mỗi vector biểu diễn một token trong chuỗi. Thành phần này không chỉ xử lý dữ liệu ở một mức độ ngữ cảnh duy nhất, mà còn chia dữ liệu thành nhiều không gian khác nhau (các "đầu chú ý"), mỗi không gian học một kiểu quan hệ riêng biệt, từ đó tăng cường khả năng tổng quát hóa của mô hình (theo paper họ chế ra 8 cái đầu này khác nhau để học từng mối quan hệ riêng từng từ).

Kiến trúc của Multi-Head Attention bao gồm ba ma trận chính được khởi tạo và học trong quá trình huấn luyện: **Query (Q)**, **Key (K)** và **Value (V)**. Các ma trận này được tạo ra bằng cách áp dụng các trọng số đã học (các ma trận trọng số) lên chuỗi đầu vào, với kích thước của mỗi ma trận phụ thuộc vào số chiều của vector đầu vào.



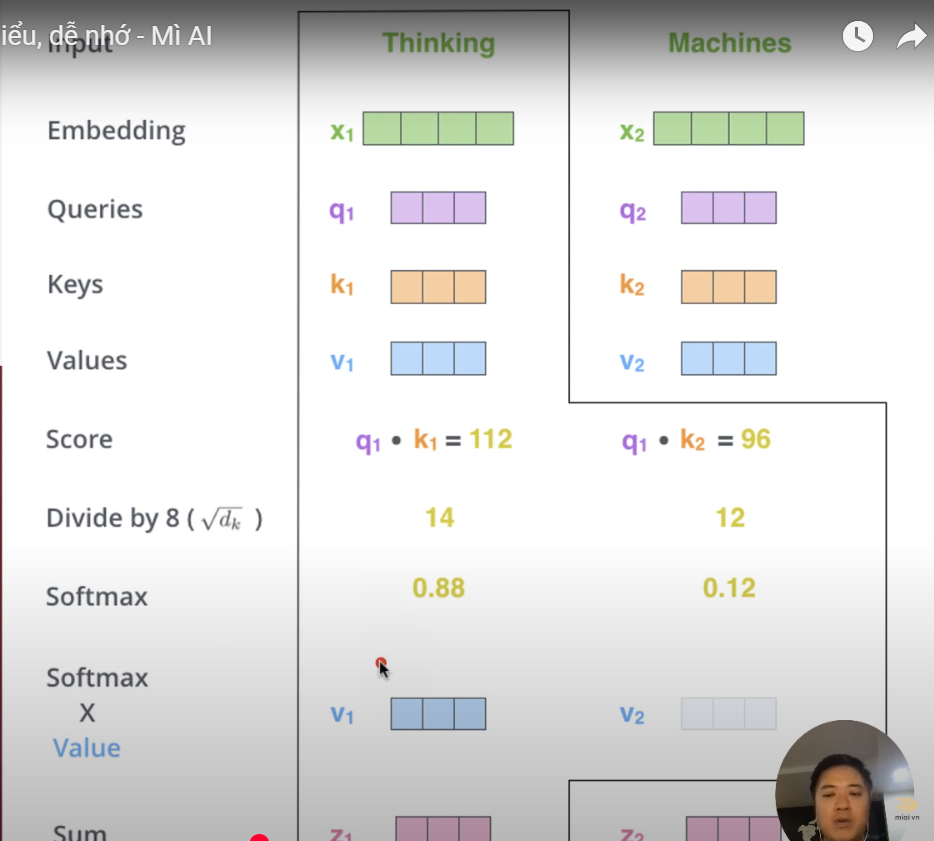
Các ma trận và thành phần tương ứng với mỗi embedding đầu vào + với các ma trận q,k,v đc khởi tạo ngẫu nhiên và học trong quá trình huấn luyện

Các ma trận Q, K, và V này lần lượt biểu diễn: token hiện tại cần tập trung vào điều gì (Query), đặc trưng của các token khác trong chuỗi (Key), và giá trị thông tin thực tế mà các token đó chứa (Value). Để tính toán mức độ quan trọng giữa các token, các phép nhân giữa ma trận Query và Key được thực hiện, sau đó được chia cho căn bậc hai của số chiều \_ (để chuẩn hóa) và áp dụng hàm softmax, tạo ra một ma trận trọng số chú ý. Công thức tính ma trận chú ý như sau:



Ma trận trọng số này xác định mức độ quan trọng của mỗi token đối với token hiện tại. Tiếp theo, ma trận trọng số được nhân với ma trận Value để tạo ra đầu ra Self-Attention, là sự kết hợp có trọng số của các thông tin liên quan trong chuỗi.

Điểm đặc biệt của Multi-Head Attention là thay vì chỉ sử dụng một "đầu chú ý", nó thực hiện quá trình này song song trên nhiều "đầu" (head), mỗi head sử dụng một tập trọng số Q, K, và V khác nhau để học các kiểu quan hệ ngữ cảnh khác nhau. Số lượng head (thường là 8 hoặc 16) được xác định trong cấu hình của mô hình và mỗi head học một khía cạnh khác nhau của mối quan hệ giữa các token, chẳng hạn như các mối quan hệ ngữ nghĩa hoặc cú pháp.

Kết quả của các head này được ghép lại (concatenated) và chuyển qua một lớp Dense (Fully Connected) để tổng hợp thành đầu ra cuối cùng. 

Ví dụ minh họa về cách tính toán của mutil head attention đầu vào là vector thực hiện phép tính rồi cho ra một vector mang nhiều ý nghĩa

Multi-Head Attention trong Encoder được tổ chức theo các lớp (layer), với số lớp thường dao động từ 6 đến 12 trong các mô hình tiêu chuẩn. Mỗi lớp bao gồm một khối Multi-Head Attention và một khối Feed-Forward Networks, được kết hợp với cơ chế Residual Connections (kết nối dư) và Layer Normalization để đảm bảo tính ổn định và khả năng hội tụ trong quá trình huấn luyện.

Kết quả đầu ra của Multi-Head Attention là một chuỗi vector, trong đó mỗi vector đại diện cho một token trong chuỗi đầu vào, nhưng đã được làm giàu thông tin ngữ cảnh thông qua việc học các mối quan hệ chú ý giữa các token. Những vector này tiếp tục được đưa qua các khối tiếp theo trong Encoder để trích xuất đặc trưng ngày càng phức tạp, chuẩn bị cho bước xử lý cuối cùng hoặc chuyển sang Decoder để thực hiện các tác vụ như dịch máy hoặc sinh văn bản.

##### 2.2.2.1.3 Feed-Forward Networks

Feed-Forward Networks (FFN) trong Transformer được thiết kế để mở rộng và tinh chỉnh thông tin ngữ cảnh đã được học bởi Multi-Head Attention. Dù cơ chế Attention rất mạnh mẽ trong việc nắm bắt các mối quan hệ giữa các token, nhưng nó hoạt động chủ yếu ở mức tương tác giữa các token mà chưa thực sự trích xuất và khuếch đại các đặc trưng riêng lẻ. FFN được sử dụng để xử lý độc lập từng token, thực hiện chuyển đổi phi tuyến tính trên các vector đầu vào, giúp mô hình học được những biểu diễn phong phú và trừu tượng hơn.

Hiểu đơn giản, Attention giống như việc bạn xác định mối quan hệ giữa các từ trong một câu (ai liên quan đến ai), còn FFN giống như việc xử lý từng từ riêng lẻ để hiểu ý nghĩa sâu hơn của nó.

Ví dụ: Sau khi Attention xác định từ "mèo" có liên quan đến "chạy" và "nhanh", FFN sẽ bổ sung thông tin như "mèo là động vật", "chạy nhanh" có ý nghĩa gì, giúp mô hình hiểu rõ hơn về từng từ trong ngữ cảnh. Điều này rất cần thiết để tạo ra biểu diễn mạnh mẽ hơn cho các bước xử lý sau.

Về cấu trúc, FFN bao gồm hai lớp Dense (Fully Connected) với một hàm kích hoạt phi tuyến giữa chúng, thường là ReLU.Đầu tiên, vector đầu vào được ánh xạ qua lớp Dense đầu tiên để tăng số chiều (thường gấp 4 lần chiều đầu vào ban đầu), giúp mạng có khả năng biểu diễn phức tạp hơn. Sau đó, kết quả được truyền qua hàm ReLU để thêm tính phi tuyến, và cuối cùng được ánh xạ ngược trở lại không gian ban đầu thông qua lớp Dense thứ hai.

Kết quả cuối cùng của FFN là một chuỗi các vector mới, trong đó mỗi vector đã được khuếch đại thông tin và làm nổi bật các đặc trưng quan trọng. Điều này giúp tăng khả năng biểu diễn ngữ nghĩa và tương tác sâu hơn của các token trong không gian trừu tượng, chuẩn bị cho các bước xử lý tiếp theo. FFN được áp dụng độc lập trên từng token, giúp giữ được tính song song hóa trong quá trình xử lý, đồng thời tối ưu hóa hiệu quả tính toán.

#### 2.2.2.2 Decoder

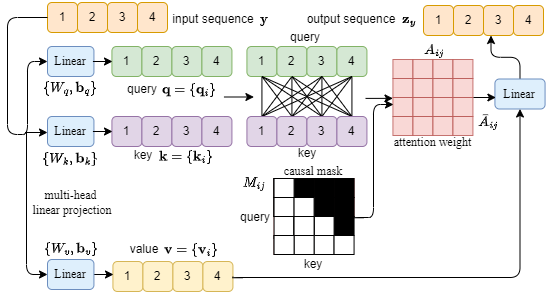
Decoder trong Transformer về cơ bản có một số thành phần tương tự với Encoder, như **Multi-Head Attention**, **Feed-Forward Networks**, **Residual Connections**, và **Layer Normalization**. Những thành phần này giúp xử lý thông tin đầu ra một cách hiệu quả, duy trì tính ổn định và học được các đặc trưng cần thiết từ chuỗi dữ liệu.

Tuy nhiên, Decoder cũng có một số điểm khác biệt quan trọng để thực hiện nhiệm vụ riêng là tạo ra đầu ra có ngữ cảnh và ý nghĩa liên kết với đầu vào. Một trong những khác biệt lớn là **Masked Multi-Head Attention**, được sử dụng để ngăn mô hình truy cập thông tin ở các token tương lai trong chuỗi đầu ra trong quá trình huấn luyện, đảm bảo rằng mô hình chỉ dự đoán dựa trên thông tin hiện có và ngữ cảnh trước đó. Thêm vào đó, **Cross-Attention** được tích hợp trong Decoder để kết nối và học cách ánh xạ giữa chuỗi đầu vào (được mã hóa bởi Encoder) và chuỗi đầu ra, đảm bảo rằng thông tin từ đầu vào có thể định hướng rõ ràng cho việc sinh đầu ra. Những thành phần bổ sung này rất quan trọng, vì nhiệm vụ của Decoder không chỉ là hiểu chuỗi dữ liệu mà còn phải tạo ra một đầu ra hợp lý, ngữ nghĩa và tương thích với ngữ cảnh đầu vào. Sự khác biệt này giúp Decoder đảm bảo tính linh hoạt trong các ứng dụng như dịch ngôn ngữ hoặc sinh văn bản.

##### 2.2.2.2.1 Masked Multi-Head Attention

**Masked Multi-Head Attention** là một thành phần quan trọng trong Decoder của Transformer, được thiết kế đặc biệt để đảm bảo rằng mô hình chỉ dự đoán dựa trên thông tin từ các token trước đó trong chuỗi đầu ra, mà không "nhìn trước" các token ở tương lai. Thành phần này tồn tại vì trong các tác vụ như dịch ngôn ngữ hoặc sinh văn bản, việc dự đoán một token tại thời điểm hiện tại không được phép sử dụng các token sẽ xuất hiện sau đó (tức là các token tương lai), nhằm giữ tính tuần tự và đảm bảo quá trình sinh chuỗi đầu ra diễn ra chính xác.

**Nhiệm vụ của Masked Multi-Head Attention** là giới hạn tầm nhìn (che đi để nó dự đoán) của mô hình bằng cách sử dụng một **ma trận mặt nạ (mask matrix)**. Ma trận này chặn (mask) các vị trí trong chuỗi đầu ra tương ứng với các token chưa được sinh ra. Cơ chế này áp dụng một giá trị rất nhỏ (ví dụ: âm vô cùng ) cho những vị trí bị chặn, làm cho hàm softmax tại những vị trí đó trả về xác suất gần bằng 0. Điều này đảm bảo rằng mô hình chỉ tập trung vào các token trước đó hoặc hiện tại khi tính toán.**Tức là nó thực hiện phép nhân với một tra trận đặc biệt để điều chỉnh sao cho ouput nó phù hợp với quá trình huấn luyện**



**Hình ảnh minh họa cơ chế hoạt động của Masked Multi-Head Attention**

**Cơ chế hoạt động của Masked Multi-Head Attention**:

1. Tương tự như Multi-Head Attention thông thường, các vector **Query (Q)**, **Key (K)** và **Value (V)** được tính toán từ chuỗi đầu ra hiện tại.
2. Một **ma trận mặt nạ** được áp dụng lên kết quả của Q.K, chặn các giá trị tương ứng với các token tương lai.
3. Kết quả được chuẩn hóa bằng cách chia cho căn bậc hai của chiều và áp dụng hàm softmax, đảm bảo chỉ các vị trí hợp lệ (tương ứng với các token trước đó) được chú ý.
4. Trọng số chú ý thu được từ softmax được nhân với ma trận V, tạo ra đầu ra attention cho mỗi token.

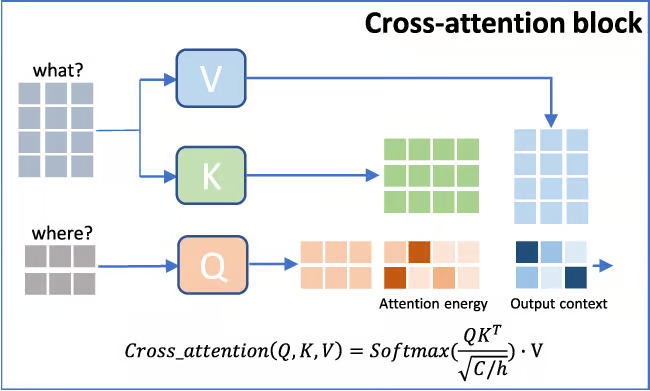
**Output** của Masked Multi-Head Attention là một chuỗi các vector đại diện cho mỗi token trong chuỗi đầu ra, nhưng chỉ dựa trên các token trước đó trong chuỗi. Điều này tạo điều kiện để mô hình xây dựng ngữ cảnh cục bộ và dự đoán token tiếp theo một cách tuần tự, mà không vi phạm nguyên tắc về thông tin tương lai. Kết quả này sau đó được chuyển qua các khối tiếp theo của Decoder, chẳng hạn như Cross-Attention và Feed-Forward Networks, để tiếp tục xử lý và tạo ra đầu ra cuối cùng. Masked Multi-Head Attention đóng vai trò thiết yếu trong việc đảm bảo tính đúng đắn của quá trình sinh chuỗi, đồng thời giữ cho mô hình phù hợp với các yêu cầu của bài toán tuần tự.

##### 2.2.2.2.2 Cross-Attention

**Cross-Attention** trong Decoder là thành phần đặc biệt giúp kết nối thông tin giữa đầu vào (được mã hóa bởi Encoder) và đầu ra đang được sinh ra bởi Decoder. Nó tồn tại vì nhiệm vụ của Decoder không chỉ dựa vào chuỗi đầu ra trước đó mà còn phải liên kết chặt chẽ với ngữ cảnh của chuỗi đầu vào để đảm bảo sự phù hợp ngữ nghĩa.

Cơ chế Cross-Attention hoạt động như sau: đầu ra của Encoder được sử dụng làm **Key (K)** và **Value (V)**, trong khi đầu ra từ **Masked Multi-Head Attention** trong Decoder được dùng làm **Query (Q)**. Mô hình tính toán sự liên quan giữa các token của chuỗi đầu ra với các token trong chuỗi đầu vào thông qua Attention, giúp học được cách ánh xạ các từ hoặc cụm từ tương ứng giữa hai chuỗi.

Cross-Attention là cần thiết vì nó cho phép Decoder truy cập toàn bộ ngữ cảnh đã được mã hóa trong Encoder, từ đó tạo ra đầu ra phù hợp với thông tin đầu vào. Kết quả là một chuỗi vector đã được làm giàu thông tin từ cả ngữ cảnh đầu vào và các dự đoán trước đó, chuẩn bị cho các bước xử lý tiếp theo trong Decoder



Cross attention là cách kết hợp thông tin giữa encoder và decoder

# **Chương III : Phương pháp tiếp cận và giải pháp giải quyết bài toán**

## 3.1 Phương pháp tiếp cận

Phương pháp tiếp cận có thể phân tách thành ba hướng chính.

### **3.1.1 Tiếp cận dựa trên học máy truyền thống (Traditional Machine Learning)**

Trong các giai đoạn đầu của nghiên cứu về nhận diện cảm xúc, các kỹ thuật học máy truyền thống đã được ứng dụng rộng rãi nhờ vào tính đơn giản, dễ triển khai và khả năng mang lại kết quả khả quan khi dữ liệu ở quy mô vừa và nhỏ. Các mô hình như Support Vector Machine (SVM), Random Forest (RF), K-Nearest Neighbors (KNN) và Naïve Bayes thường được sử dụng để xây dựng các bộ phân loại cảm xúc dựa trên các đặc trưng trích xuất thủ công từ nhiều nguồn dữ liệu khác nhau.

Trong bài toán multi-modal emotion recognition, dữ liệu đầu vào thường có nhiều loại : hình ảnh khuôn mặt, giọng nói, văn bản (như hội thoại hoặc nội dung bài phát biểu), thậm chí là tín hiệu sinh học (như EEG hoặc nhịp tim). Với cách tiếp cận truyền thống, mỗi loại dữ liệu cần được xử lý riêng biệt để trích xuất các đặc trưng có ý nghĩa:

* Hình ảnh (khuôn mặt):  
  Các kỹ thuật như Histogram of Oriented Gradients (HOG) và Local Binary Patterns (LBP) được sử dụng để mô tả hình dạng, kết cấu của khuôn mặt và các biểu cảm đặc trưng. Những đặc trưng này sau đó sẽ được đưa vào các mô hình học máy để phân loại các trạng thái cảm xúc như vui, buồn, tức giận, sợ hãi, ngạc nhiên, v.v.
* Âm thanh (giọng nói):  
  Các đặc trưng như Mel-Frequency Cepstral Coefficients (MFCC), Chroma Features, Pitch, và Spectrogram thường được sử dụng để đại diện cho các yếu tố cảm xúc trong giọng nói như cường độ, cao độ, nhịp điệu hoặc âm sắc. Các đặc trưng này phản ánh sự thay đổi cảm xúc một cách tự nhiên trong ngữ điệu của người nói.
* Văn bản:  
  Khi có sẵn dữ liệu lời thoại hoặc nội dung hội thoại, kỹ thuật xử lý ngôn ngữ tự nhiên (NLP) có thể được áp dụng để trích xuất các đặc trưng như TF-IDF, Bag-of-Words, hoặc sử dụng các lexicon về cảm xúc (như NRC hoặc SentiWordNet) để lượng hóa cảm xúc dựa trên từ vựng.
* Tín hiệu sinh học (nếu có):  
  Trong một số hệ thống cao cấp, dữ liệu như nhịp tim, điện não đồ (EEG) cũng có thể được sử dụng để phát hiện phản ứng cảm xúc. Các đặc trưng như tần số alpha, beta, hoặc HRV (Heart Rate Variability) được trích xuất để phục vụ cho phân loại.

### **3.1.2 Tiếp cận dựa trên Học sâu (Deep Learning)**

Việc áp dụng các mô hình Convolutional Neural Network (CNN) để trích xuất đặc trưng không gian từ ảnh khuôn mặt hoặc khung hình video, thường sử dụng các kiến trúc mạng phổ biến như VGG-16, ResNet, MobileNet, hoặc kết hợp với Facial Action Coding System (FACS) để nhận diện các chuyển động cơ mặt. Bên cạnh đó, các mạng hồi tiếp (Recurrent Neural Network - RNN), đặc biệt là LSTM hay GRU, được sử dụng cho dữ liệu dạng chuỗi như âm thanh, cho phép nắm bắt thông tin thời gian và bối cảnh nhằm nâng cao hiệu quả nhận diện cảm xúc. Một số giải pháp kết hợp (hybrid) như CNN-LSTM cũng được phát triển để xử lý song song cả không gian và thời gian, áp dụng cho các đoạn video biểu cảm hoặc chuỗi tín hiệu âm thanh phức tạp. Hướng tiếp cận này tự động hóa quá trình trích xuất đặc trưng, linh hoạt hơn khi mở rộng cho các tập dữ liệu lớn, tuy nhiên đòi hỏi tài nguyên tính toán cao cũng như kỹ năng thiết kế, tinh chỉnh mô hình

### **3.1.3 Tiếp cận Đa phương thức (Multimodal Learning)**

Một trong những lý do thúc đẩy hướng tiếp cận này nằm ở chỗ, cảm xúc con người hiếm khi thể hiện rõ ràng qua một yếu tố duy nhất, mà thường hiện diện đồng thời trong nét mặt, ánh mắt, cử chỉ, giọng điệu, tốc độ nói hay nhịp điệu âm thanh. Khi xử lý bài toán chỉ dựa vào riêng khuôn mặt, mô hình có thể bỏ lỡ những dấu hiệu cảm xúc đến từ sự thay đổi cường độ hay ngữ điệu. Ngược lại, nếu chỉ dựa vào giọng nói, phương pháp cũng khó đạt độ chính xác cao khi người nói có giọng trầm hoặc môi trường ghi âm nhiều nhiễu. Học đa phương thức giúp khắc phục những hạn chế này bằng cách kết hợp đặc trưng từ hai nguồn dữ liệu, cho phép mô hình nhận diện cảm xúc với độ tin cậy cao hơn.

Việc kết hợp (fusion) hai dạng dữ liệu khuôn mặt và giọng nói có thể diễn ra theo nhiều cách khác nhau. Với chiến lược kết hợp sớm (early fusion), mô hình sẽ tiếp nhận đồng thời thông tin từ cả hai modal ngay từ bước đầu, tạo điều kiện cho mạng nơ-ron học mối liên hệ giữa chúng ở giai đoạn trích xuất đặc trưng. Phương pháp này khuyến khích việc nắm bắt tương quan phức tạp giữa hình ảnh và âm thanh, nhưng đồng thời đòi hỏi quá trình đồng bộ và sắp xếp dữ liệu kỹ lưỡng nhằm tránh tình trạng mất cân bằng hoặc chồng chéo thông tin.

Trong khi đó, kết hợp muộn (late fusion) lại tách riêng các luồng xử lý cho mỗi modal, ví dụ một mô hình CNN (hoặc Vision Transformer) chuyên phân tích khuôn mặt và một mô hình LSTM (hoặc Audio Transformer) chuyên phân tích tín hiệu âm thanh. Đến giai đoạn cuối, các đầu ra dự đoán mới được tổ hợp lại để đưa ra kết quả chung. Cách tiếp cận này mang đến sự linh hoạt trong khâu thiết kế, vì mỗi modal có thể được tối ưu hóa theo đặc thù riêng (chẳng hạn xử lý hình ảnh với mạng CNN, xử lý giọng nói với mạng LSTM), nhưng đôi lúc việc gộp kết quả rời rạc không tận dụng hết tiềm năng của việc gắn kết hai modal ngay từ sớm.

A diagram of a block diagram

AI-generated content may be incorrect.

Các phương pháp kết hợp fusion trong mutil modal

Kết hợp lai (hybrid fusion) là một lựa chọn dung hòa giữa hai trường phái trên khi mô hình vừa trích xuất đặc trưng chuyên biệt cho từng loại dữ liệu, vừa chia sẻ thông tin ở một vài tầng trung gian. Sự pha trộn này cho phép mô hình tận dụng lợi thế của cả hai chiến lược, vừa giữ được khả năng quan sát chi tiết trong từng luồng, vừa hình thành mạch trao đổi thông tin sớm hơn so với kết hợp muộn.

Trong học sâu đa phương thức, nhiều kiến trúc tiên tiến đang được áp dụng, chẳng hạn các biến thể của Transformer (như Multimodal Transformers, MMTransformer, VisualBERT, VideoBERT) hoặc những mô hình dựa trên Graph Neural Networks (GNNs). Các kiến trúc Transformer thường khai thác cơ chế self-attention để tìm ra vùng thông tin quan trọng trong từng modal, rồi tiếp tục xác định cách những vùng quan trọng này liên kết qua lại. Cách tiếp cận này giúp mô hình nắm bắt mối tương quan đa chiều giữa hình ảnh và âm thanh mà không bị bó buộc bởi thứ tự hay cấu trúc rời rạc. Trong khi đó, GNNs cho phép biểu diễn dữ liệu dưới dạng đồ thị, mà mỗi nút (node) có thể tượng trưng cho một đặc trưng hoặc một khung hình, còn cạnh (edge) thể hiện mối liên hệ về vị trí, thời gian hoặc nội dung. Phương pháp đồ thị thích hợp cho những kịch bản đòi hỏi suy luận phức tạp, trong đó một hoặc nhiều đoạn âm thanh có thể liên quan đến một tập hợp khung hình cụ thể, và ngược lại.

Tuy vậy, học đa phương thức cũng đặt ra những yêu cầu cao hơn về dữ liệu và tính toán. Trước hết, khuôn mặt và giọng nói phải được căn chỉnh thời gian để đảm bảo tính thống nhất, nhất là khi nguồn dữ liệu đến từ nhiều camera, micro hoặc định dạng khác nhau. Việc đồng bộ này đôi khi gặp khó khăn trong điều kiện thực tế với âm thanh nhiều tạp, độ nhiễu cao, hoặc khi video bị rung, mờ. Thêm vào đó, chi phí huấn luyện cũng dễ tăng lên đáng kể do mô hình phải xử lý và học đặc trưng từ hai modal song song, khiến bộ tham số phình to. Một số nghiên cứu mới đây đang tập trung giảm tải chi phí này bằng cách trích xuất trước đặc trưng rồi mới kết hợp, hoặc sử dụng các cơ chế chia sẻ trọng số ở những lớp đầu để hạn chế tình trạng “thừa” tham số.

## 3.2 Chiến lược và xây dựng mô hình sử dụng cho phương pháp đa phương thức

#### 3.2.1.1 Late Transformer Fusion

A diagram of a block diagram

AI-generated content may be incorrect.

Kiến trúc tổng quan

Hình trên minh họa kiến trúc Late Transformer Fusion cho bài toán nhận diện cảm xúc đa phương thức (multi-modal emotion recognition) sử dụng hai nguồn dữ liệu chính là video (thường biểu diễn khuôn mặt hoặc biểu cảm) và audio (giọng nói).

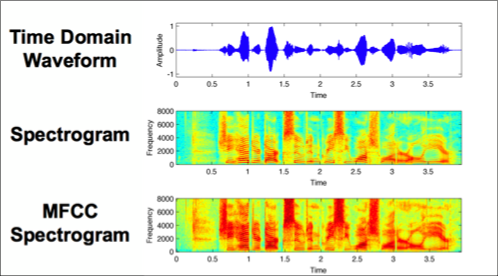
Ở nhánh video, mô hình đầu tiên sử dụng một mạng trích xuất đặc trưng hình ảnh tiên tiến là EfficientFace để lấy ra các đặc trưng khuôn mặt từ chuỗi khung hình đầu vào, sau đó đưa các đặc trưng này qua bốn tầng Conv1D block để mã hóa thông tin theo chuỗi thời gian nhằm nắm bắt động thái biểu cảm. A diagram of a person's work flow

AI-generated content may be incorrect.

Kiến trúc của EfficientFace

Song song, nhánh audio cũng được xử lý qua bốn tầng Conv1D block, từ các tín hiệu giọng nói đầu vào (đã được chuyển đổi dạng như spectrogram hoặc MFCC), nhằm trích xuất và nắm bắt thông tin thay đổi âm thanh theo thời gian.

Song song với nhánh hình ảnh, nhánh âm thanh được thiết kế nhằm khai thác những thông tin hữu ích từ giọng nói, ngữ điệu, âm lượng và các đặc điểm âm thanh khác của người nói.



Phổ âm thanh MFCC

Ban đầu, tín hiệu âm thanh được chuyển đổi sang các hệ số cepstral tần số mel (MFCC), đây là phương pháp phổ biến trong xử lý giọng nói vì MFCC mô phỏng khá sát quá trình nghe của con người và giúp trích xuất tương đối đầy đủ các thành phần tần số quan trọng. Sau khi thu được ma trận MFCC, hệ thống đi qua một chuỗi các khối tích chập 1D, trong đó mỗi khối thường bao gồm bước chuẩn hóa hàng loạt (Batch Normalization), hàm kích hoạt ReLU và thao tác pooling như MaxPooling. Cách bố trí này giúp mạng khái quát hóa tốt hơn và tránh tình trạng biến đổi cục bộ quá đột ngột trong tập dữ liệu. Bởi vì dữ liệu âm thanh mang tính chuỗi theo thời gian, các kernel tích chập 1D trượt dọc theo trục thời gian để nhận biết những biến thiên trong đặc trưng âm thanh. Nếu như khung hình khuôn mặt tập trung vào việc phát hiện các cử động cơ mặt, thì các đặc trưng âm thanh ở đây có thể được dùng để xác định sự chuyển biến trong cao độ, biên độ hay sắc thái giọng nói. Việc duy trì một chuỗi bốn khối tích chập 1D cũng giúp mô hình có đủ độ sâu để học các đặc trưng từ thô (low-level features) đến cao (high-level features), hỗ trợ phân biệt tốt hơn giữa các sắc thái giọng nói khác nhau.

##### **Nhánh hình ảnh**

|  |  |
| --- | --- |
| Component | Description |
| EfficientFace Module |  |
| Stage 1 | Reshape |
|  | Conv1D [k=3, d=64, s=1] + BN1D + ReLU |
|  | Conv1D [k=3, d=64, s=1] + BN1D + ReLU |
| Stage 2 | Conv1D [k=3, d=128, s=1] + BN1D + ReLU |
| Predict | Conv1D [k=3, d=128, s=1] + BN1D + ReLU |
|  | Global Average Pooling + Linear |

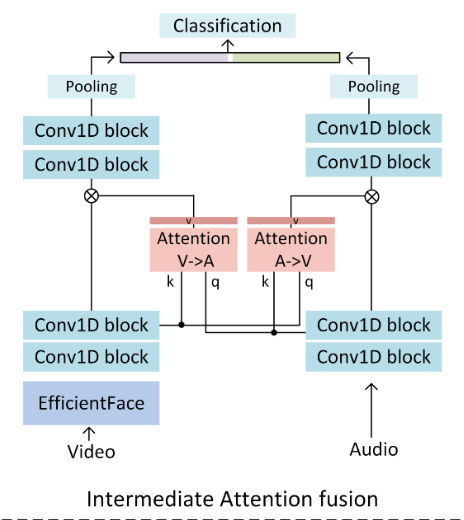
##### **Nhánh âm thanh**

|  |  |
| --- | --- |
| Component | Description |
| Stage 1 | Conv1D [k=3, d=64] + BN1D + ReLU + MaxPool1d [2x1] |
|  | Conv1D [k=3, d=128] + BN1D + ReLU + MaxPool1d [2x1] |
| Stage 2 | Conv1D [k=3, d=256] + BN1D + ReLU + MaxPool1d [k=2] |
|  | Conv1D [k=3, d=128] + BN1D + ReLU + MaxPool1d [k=2] |
| Predict | Global Average Pooling + Linear |

Sau khi trích xuất đặc trưng riêng biệt, kiến trúc tiến hành trộn trễ (late fusion) bằng cách sử dụng hai khối Transformer – một khối lấy audio làm query và video làm key-value (Transformer Block A→V) và ngược lại với Transformer Block V→A, cho phép mỗi modal học được mối liên hệ chéo với modal còn lại thông qua cơ chế attention (chú ý). Sau mỗi khối Transformer là một bước pooling để giảm chiều và cô đọng thông tin quan trọng trước khi ghép các vector lại và đưa vào tầng phân loại cuối cùng (Classification) nhằm xác định trạng thái cảm xúc.

Việc sử dụng Transformer ở giai đoạn trễ giúp mô hình không chỉ học đặc trưng riêng biệt của từng modal mà còn nắm được các mối tương quan chéo phức tạp giữa âm thanh và hình ảnh, từ đó cải thiện hiệu quả nhận diện cảm xúc đặc biệt trong các tình huống không đồng bộ hoặc có sự mâu thuẫn giữa các modal. Kiến trúc này đại diện cho một xu hướng hiện đại trong học sâu đa phương thức, kết hợp khả năng mô hình hóa chuỗi mạnh mẽ của Conv1D với cơ chế attention của Transformer nhằm đạt được tính chính xác cao mà vẫn duy trì tính mở rộng linh hoạt với các nguồn dữ liệu khác nhau.

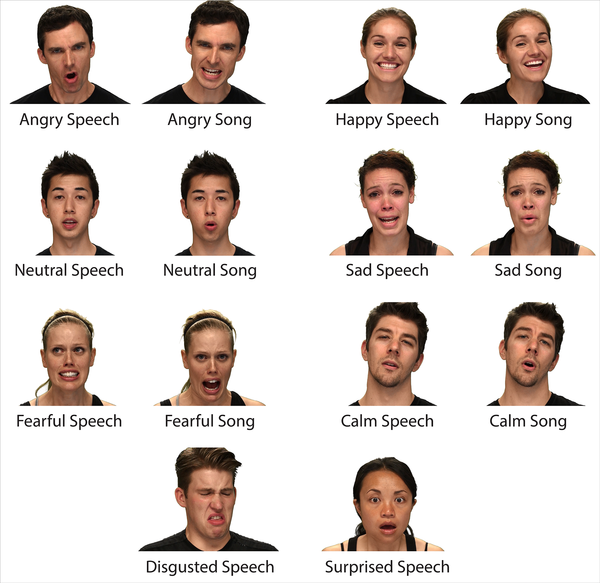
#### 3.2.1.2 Intermediate Attention Fusion



Kiến trúc Intermediate Attention Fusion trong hình thứ hai có cấu trúc tương tự với kiến trúc Late Transformer Fusion trước đó, nhưng điểm khác biệt chính nằm ở vị trí và cách thức tích hợp thông tin giữa các modal. Thay vì thực hiện kết hợp sau khi đã hoàn tất toàn bộ quá trình trích xuất đặc trưng và chỉ dùng Transformer ở tầng cuối như trong Late Fusion, ở kiến trúc Intermediate này, các khối Attention V→A và A→V được chèn vào giữa pipeline, ngay sau các tầng Conv1D block đầu tiên ở mỗi modal. Cụ thể, sau khi video được xử lý qua EfficientFace và hai tầng Conv1D đầu tiên, và audio cũng qua hai tầng Conv1D tương ứng, thông tin từ hai modal sẽ được trao đổi chéo thông qua hai khối cross-modal attention, rồi tiếp tục được đưa vào các khối Conv1D bổ sung trước khi pooling và classification. Cách tích hợp này cho phép mô hình học được các mối tương quan giữa video và audio ngay trong quá trình trích xuất đặc trưng, thay vì chờ đến giai đoạn cuối, giúp tăng khả năng đồng bộ hóa thông tin và cải thiện nhận diện cảm xúc trong các trường hợp tín hiệu cảm xúc phân bố không đều giữa các modal. Đây là điểm khác biệt cốt lõi làm cho Intermediate Fusion có thể cân bằng tốt hơn giữa học đặc trưng riêng và đặc trưng liên modal so với Late Fusion.

# Chương 4.Thực nghiệm đánh giá kết quả phương pháp Intermediate Attention Fusion

## 4.1 Tập dữ liệu sử dụng



Mẫu dữ liệu ravdess

Bộ dữ liệu được lựa chọn trong nghiên cứu này là **Ryerson Audio-Visual Database of Emotional Speech and Song (RAVDESS)**, một trong những tập dữ liệu phổ biến và được sử dụng rộng rãi trong các nghiên cứu về nhận dạng và phân tích cảm xúc. Bộ dữ liệu RAVDESS đặc biệt nổi bật nhờ tính toàn diện và sự đa dạng trong cách thể hiện, giúp các mô hình học máy đa phương thức có thể khai thác hiệu quả thông tin đồng thời từ biểu cảm khuôn mặt và tín hiệu giọng nói. Tổng thể, bộ dữ liệu RAVDESS bao gồm tới **7,356 file** với dung lượng dữ liệu lớn lên tới khoảng **24.8 GB**, cho phép triển khai các mô hình học sâu và đa phương thức quy mô lớn một cách hiệu quả.

Đặc biệt, bộ dữ liệu này được thu thập từ sự tham gia của **24 diễn viên chuyên nghiệp**, trong đó có sự cân bằng về giới tính với **12 diễn viên nam và 12 diễn viên nữ**, giúp bộ dữ liệu đảm bảo tính đa dạng và khả năng khái quát hóa cho mô hình trên nhiều đối tượng khác nhau. Các diễn viên này được yêu cầu thể hiện một cách rõ ràng, chân thực các trạng thái cảm xúc đa dạng như vui vẻ, buồn bã, giận dữ, sợ hãi, ngạc nhiên, chán ghét và trạng thái trung lập. Sự chuyên nghiệp trong diễn xuất của các diễn viên giúp đảm bảo rằng mỗi cảm xúc đều được thể hiện một cách đáng tin cậy và nhất quán, từ đó mô hình có thể học các đặc trưng cảm xúc chính xác và rõ ràng hơn.

Ngoài yếu tố đa dạng về người tham gia (gồm 12 diễn viên nữ và 12 diễn viên nam), bộ dữ liệu RAVDESS còn đặc biệt hấp dẫn bởi tính đa dạng và phong phú về nội dung thể hiện. Cụ thể, mỗi diễn viên tham gia đã được hướng dẫn phát âm hai câu văn bản cố định, trong đó họ thể hiện mỗi câu với nhiều mức độ cảm xúc khác nhau và với các hình thức khác nhau như nói thông thường hoặc hát. Tổng cộng mỗi cảm xúc được thể hiện qua nhiều mức độ cảm xúc khác nhau (cường độ bình thường hoặc cao độ hơn), và ngoài ra còn có thêm một trạng thái trung lập. Điều này không chỉ cho phép nghiên cứu khả năng nhận diện cảm xúc cơ bản, mà còn hỗ trợ việc phát hiện các sắc thái tinh tế hơn của cảm xúc thông qua cả tín hiệu âm thanh lẫn hình ảnh.

Một điểm đáng chú ý nữa của RAVDESS chính là việc dữ liệu được ghi lại và cung cấp dưới ba dạng phương thức (modal) khác nhau bao gồm:

* **Audio-only**: dạng file âm thanh độc lập (.wav) với chất lượng ghi âm chuyên nghiệp, được mã hóa ở chuẩn 16-bit PCM, tần số lấy mẫu cao 48kHz. Điều này cho phép các mô hình xử lý âm thanh hiệu quả, với đầy đủ thông tin về ngữ điệu, âm lượng, nhịp điệu và sắc thái giọng nói để có thể học được các đặc trưng hữu ích trong việc nhận diện cảm xúc.
* **Audio-Video**: dữ liệu âm thanh và hình ảnh được đồng bộ chặt chẽ trong từng video clip định dạng .mp4 độ phân giải HD (720p, codec H.264 cho video, AAC 48kHz cho âm thanh), giúp mô hình học được mối quan hệ thời gian chuẩn xác giữa từng cử chỉ khuôn mặt và tín hiệu âm thanh tương ứng. Đây chính là nền tảng quan trọng để xây dựng các mô hình đa phương thức, cho phép kết hợp thông tin hiệu quả từ hai nguồn dữ liệu này.
* **Video-only**: cho phép nghiên cứu chuyên sâu riêng về nhận dạng cảm xúc qua biểu cảm khuôn mặt khi bỏ qua sự tác động của âm thanh. Định dạng này rất hữu ích khi nghiên cứu muốn tập trung phân tích sâu về thị giác và có thể làm cơ sở đối chiếu so sánh với các phương thức tích hợp đa phương thức khác.

Ngoài ra, việc dữ liệu được ghi lại ở định dạng chất lượng cao cũng là điểm mạnh lớn của bộ RAVDESS. Các tín hiệu âm thanh đều được thu lại với độ phân giải cao (16bit, 48kHz, định dạng WAV), mang lại độ chi tiết âm thanh rõ nét, hỗ trợ trích xuất các đặc trưng tần số hiệu quả như hệ số cepstral tần số mel (MFCC) hay mel-spectrogram. Đối với dữ liệu hình ảnh, các file video đều được quay lại dưới chuẩn video HD 720p với định dạng nén H.264, giúp hình ảnh khuôn mặt rõ ràng và sắc nét, thuận lợi cho việc trích xuất các đặc trưng không gian, cũng như đảm bảo được chất lượng tốt khi sử dụng các mạng CNN hoặc EfficientFace.

## 4.2 Huấn luyện mô hình

Quá trình huấn luyện mô hình nhận diện cảm xúc đa phương thức trong nghiên cứu này được thực hiện trên môi trường Google Colab, một nền tảng tính toán đám mây miễn phí phổ biến của Google, cung cấp khả năng sử dụng GPU giúp tăng tốc đáng kể quá trình xử lý và huấn luyện các mô hình học sâu phức tạp. Cụ thể, mã nguồn và dữ liệu được tải lên Colab,

trong đó các tác vụ chính bao gồm:

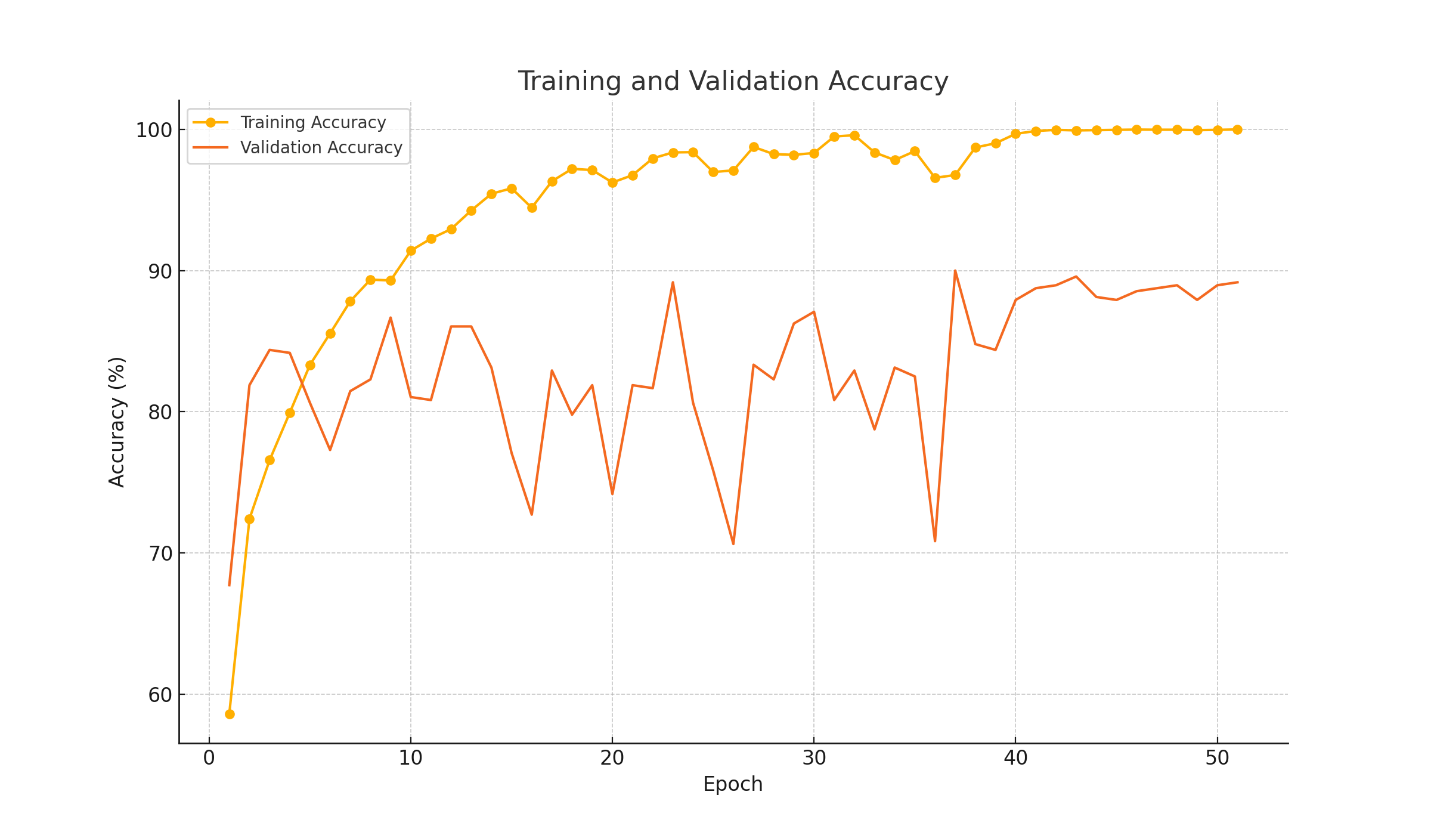
(1) Mã nguồn có sẵn

(2) xử lý tiền dữ liệu như trích xuất khuôn mặt từ video, chuyển đổi âm thanh bằng thư viện Librosa, và tạo các tập chú thích dữ liệu (annotations). Sau đó, toàn bộ mã nguồn và dữ liệu xử lý hoàn chỉnh được lưu trữ lại trên Google Drive nhằm bảo quản dữ liệu và tiện lợi cho các phiên làm việc tiếp theo. Tổng thời gian huấn luyện mô hình kéo dài khoảng 6 tiếng đồng hồ trên môi trường Colab sử dụng GPU miễn phí. Điều này giúp cân bằng giữa hiệu suất tính toán và hiệu quả về mặt chi phí, đồng thời tận dụng lợi thế của các tài nguyên GPU trên Google Colab để huấn luyện các mạng học sâu một cách hiệu quả và nhanh chóng.



Mẫu video chỉ có face đc trích xuất

## 4.3 Đánh giá mô hình



Kết quả huấn luyện và xác thực của mô hình nhận diện cảm xúc đa phương thức đã cho thấy sự hiệu quả và khả năng học tốt từ dữ liệu trong quá trình huấn luyện. Xét trên toàn bộ quá trình huấn luyện kéo dài 50 epochs, có thể nhận thấy rõ ràng sự tiến bộ và cải thiện đáng kể về hiệu suất của mô hình theo thời gian. Điều này được thể hiện qua việc hàm mất mát (loss) đã giảm đáng kể từ mức khá cao ban đầu (khoảng **1.3742**) ở epoch đầu tiên xuống mức thấp (dưới **0.1**) sau khoảng hơn 20 epochs. Việc giảm nhanh hàm mất mát này cho thấy mô hình có khả năng học tốt các đặc trưng cảm xúc quan trọng từ dữ liệu đa phương thức (hình ảnh và âm thanh), giúp cho quá trình huấn luyện diễn ra thuận lợi.

Độ chính xác (accuracy) trên tập huấn luyện đạt được sự cải thiện đáng kể và nhanh chóng. Cụ thể, ở epoch đầu tiên, độ chính xác chỉ đạt khoảng **58.59%**, nhưng đã tăng đáng kể chỉ sau một vài epochs, đạt mức trên **90%** sau khoảng 10 epoch đầu tiên, và tiếp tục tăng ổn định, đạt tới trên **98%** ở các epoch sau cùng. Đặc biệt, top-5 accuracy luôn ở mức rất cao (gần như tuyệt đối, trên **99%** ở hầu hết các epoch), cho thấy rằng mô hình không chỉ dự đoán chính xác cảm xúc mà còn có khả năng phân biệt rõ ràng các cảm xúc rất gần nhau hoặc dễ nhầm lẫn. Đây là một dấu hiệu tích cực, thể hiện tính ổn định cao trong khả năng dự đoán và phân loại các trạng thái cảm xúc của mô hình.

Tuy nhiên, khi đánh giá kết quả trên tập xác thực (validation set), chúng ta nhận thấy một vài điểm đáng chú ý. Ban đầu, độ chính xác trên tập xác thực cũng tăng nhanh chóng từ khoảng **67.71%** ở epoch đầu lên mức cao nhất khoảng **86.67%** (epoch thứ 9), cho thấy mô hình có khả năng khái quát hóa tốt ở giai đoạn đầu của quá trình huấn luyện. Nhưng sau giai đoạn này, chúng ta bắt đầu nhận thấy sự dao động khá rõ nét trong kết quả validation accuracy, cụ thể từ epoch thứ 10 trở đi, độ chính xác dao động liên tục trong khoảng từ **70.63% đến 90.00%**, đôi lúc giảm khá rõ nét ở một số epoch, điều này thể hiện mô hình có xu hướng gặp phải hiện tượng quá khớp (overfitting), đặc biệt là khi đạt tới những epoch cao hơn (trên epoch thứ 20). Hiện tượng này xảy ra khi mô hình quá tập trung vào các đặc trưng cụ thể trong tập huấn luyện mà không có khả năng thích nghi tốt với các trường hợp mới chưa gặp trước đó, làm giảm hiệu quả mô hình khi áp dụng vào tập dữ liệu xác thực hay kiểm thử.

Ngoài ra, sự dao động và biến thiên trong accuracy trên tập xác thực cũng cho thấy rằng tập dữ liệu RAVDESS được sử dụng khá phức tạp và đa dạng về đặc tính cảm xúc cũng như điều kiện thu thập dữ liệu. Các yếu tố như điều kiện ánh sáng, chất lượng ghi âm, sự đa dạng về cách thể hiện cảm xúc của các diễn viên khác nhau… có thể là nguyên nhân dẫn đến sự biến động này. Điều này ngụ ý rằng mô hình vẫn cần được cải tiến thêm, ví dụ thông qua các biện pháp tăng cường dữ liệu (data augmentation), áp dụng các kỹ thuật chính quy hóa (regularization), điều chỉnh tốc độ học (learning rate), hoặc kỹ thuật dừng sớm (early stopping) để cải thiện khả năng khái quát hóa tốt hơn trên dữ liệu thực tế, hạn chế hiện tượng overfitting.

# KẾT LUẬN