PHÂN TÍCH TƯƠNG QUAN CHÍNH TẮC

BÁO CÁO ĐỒ ÁN MÔN PHÂN TÍCH THỐNG KÊ DỮ LIỆU NHIỀU BIẾN

19127120 – Ngô Nhật Du

HCMUS

MỤC LỤC

1. LÝ THUYẾT PHÂN TÍCH TƯƠNG QUAN CHÍNH TẮC

1.1. Giới thiệu:

Phân tích tương quan chính tắc (CCA) (Jordan, 1875; Hotelling, 1936) là một công cụ hữu ích trong việc mở khóa các mối quan hệ phức tạp giữa nhiều biến trong tập dữ liệu lớn. Điểm mạnh chính của CCA là nó có thể đánh giá đồng thời hai bộ biến khác nhau mà không cần giả định bất kỳ hình thức ưu tiên hoặc định hướng cụ thể nào. Ví dụ: CCA cho phép ma trận dữ liệu đo lường não (ví dụ: liên kết kết nối giữa một tập hợp các vùng não) được phân tích đồng thời đối với ma trận dữ liệu thứ hai về các phép đo hành vi. Nói cách khác, CCA xác định các nguồn của biến thể phổ biến trong hai tập hợp các biến số chiều cao. Tương quan chính tắc thích hợp trong các tình huống tương tự như hồi quy bội số, nhưng có nhiều biến kết quả tương quan lẫn nhau

Càng ngày càng có nhiều bộ dữ liệu high-dimensional từ các mẫu giống nhau, do đó Phân tích tương quan chính tắc rất phù hợp cho các tình huống trong đó có hai tập dữ liệu high-dimensional từ các mẫu giống nhau và CCA cho phép học cách xem các tập dữ liệu đồng thời. Một ví dụ cổ điển là tập dữ liệu âm thanh và video từ các cá nhân giống nhau. Cũng có thể nói CCA là một kỹ thuật giảm kích thước khác như Phân tích thành phần chính (PCA), tuy nhiên CCA xử lý hai tập dữ liệu thay vì một như PCA.

Khi một quá trình có thể được mô tả bởi hai tập hợp các biến tương ứng với hai khía cạnh hoặc góc nhìn khác nhau, phân tích mối quan hệ giữa hai góc nhìn này có thể cải thiện sự hiểu biết về hệ thống cơ bản. Trong bối cảnh này, một mối quan hệ là một bản đồ của các quan sát tương ứng với một biến của một dạng xem với các quan sát tương ứng với một biến của chế độ xem khác. Ví dụ trong lĩnh vực y học, một quan điểm có thể bao gồm các biến tương ứng với các triệu chứng của bệnh và quan điểm khác với các yếu tố nguy cơ có thể ảnh hưởng đến tỷ lệ mắc bệnh. Xác định mối quan hệ giữa các triệu chứng và các yếu tố nguy cơ có thể cải thiện sự hiểu biết về phơi nhiễm bệnh và đưa ra chỉ định phòng ngừa và điều trị.

Kể từ khi đề xuất phân tích tương quan chính tắc (CCA) của H. Hotelling [Hotelling 1935; Hotelling 1936], mối quan hệ giữa các biến đã được khám phá trong các lĩnh vực khoa học khác nhau. CCA lần đầu tiên được áp dụng để kiểm tra mối quan hệ của các đặc tính lúa mì với đặc tính bột mì trong một nghiên cứu kinh tế của F. Waugh vào năm 1942 [Waugh 1942]. Kể từ đó, các nghiên cứu trong lĩnh vực tâm lý học [Hopkins 1969; Dunham and Kravetz 1975], địa lý [Monmonier and Finn 1973], y học [Lindsey et al. 1985], vật lý [Wong et al. 1980], hóa học [Tu et al. 1989], sinh học [Sullivan 1982], mô hình chuỗi thời gian [Heij and Roorda 1991], và xử lý tín hiệu [Schell và Gardner 1995] tạo thành các ví dụ về các lĩnh vực ứng dụng ban đầu của CCA.

Vào đầu thế kỷ 21, khả năng ứng dụng của CCA đã được chứng minh trong các lĩnh vực khoa học hiện đại như khoa học thần kinh, học máy và tin sinh học. Các mối quan hệ đã được khám phá để phát triển giao diện não-máy tính [Cao et al. 2015; Nakanishi et al. 2015] và trong lĩnh vực di truyền hình ảnh [Fang et al. 2016]. CCA cũng đã được áp dụng cho lựa chọn đặc điểm [Ogura et al. 2013], trích xuất đặc điểm và hợp nhất [Shen et al. 2013], và giảm kích thước [Wang et al. 2013]. Ví dụ về các nghiên cứu ứng dụng được thực hiện trong các lĩnh vực tin sinh học và sinh học tính toán bao gồm [Rousu et al. 2013; Seoane et al. 2014; Baur và Bozdag 2015; Sarkar and Chakraborty 2015; Cichonska et al. 2016]. Phạm vi rộng lớn của các lĩnh vực ứng dụng nhấn mạnh tiện ích của CCA trong việc trích xuất mối quan hệ giữa các biến.

Ban đầu, CCA được phát triển để trích xuất các mối quan hệ tuyến tính trong các cài đặt được xác định quá mức, đó là khi số lượng quan sát vượt quá số lượng biến trong cả hai dạng xem. Để mở rộng CCA đến các cài đặt chưa được xác định thường xảy ra trong phân tích dữ liệu hiện đại, các phương pháp chính thức hóa đã được đề xuất. Khi kích thước mẫu nhỏ, Bayesian CCA cũng cung cấp một giải pháp thay thế để thực hiện CCA. Khả năng áp dụng của CCA cho các thiết lập chưa được xác định đã được cải thiện hơn nữa thông qua các tiêu chuẩn sparsityinducing tạo điều kiện cho việc giải thích kết quả cuối cùng. Các phương pháp hạt nhân và mạng lưới thần kinh đã được giới thiệu để khám phá các mối quan hệ phi tuyến tính. Hiện tại, các phương pháp tương quan chính tắc có thể được sử dụng để trích xuất các mối quan hệ tuyến tính và phi tuyến tính trong cả hai cài đặt quá mức và chưa được xác định.

Ngoài các biến thể đã được mô tả của CCA, các phần mở rộng thay thế đã được đề xuất, chẳng hạn như CCA bán ghép và đa dạng. Nói chung, các thuật toán CCA giả định sự tương ứng một-một giữa các quan sát trong các quan điểm, nói cách khác, dữ liệu được giả định là được ghép nối. Tuy nhiên, trong các bộ dữ liệu thực tế, một số quan sát có thể bị thiếu trong cả hai chế độ xem, điều đó có nghĩa là các quan sát được bán dẫn. Ví dụ về các thuật toán CCA bán ghép nối bao gồm [Blaschko et al. 2008], [Kimura et al. 2013], [Chen et al. 2012], và [Zhang et al. 2014]. CCA cũng đã được mở rộng đến hơn hai lượt xem bởi [Horst 1961], [Carroll 1968], [Kettenring 1971], và [Van de Geer

1984]. Trong multi-view CCA, các mối quan hệ được tìm kiếm giữa nhiều hơn hai quan điểm. Một số phần mở rộng hiện đại của CCA đa chế độ xem bao gồm các biến thể chính thức [Tenenhaus và Tenenhaus 2011], hạt nhân [Tenenhaus et al. 2015], và các biến thể thưa thớt [Tenenhaus et al. 2014]. Các nghiên cứu ứng dụng về CCA đa dạng xem và các biến thể hiện đại của nó có thể được tìm thấy trong khoa học thần kinh [Kang et al. 2013], [Chen et al. 2014], đặc điểm hợp nhất [Yuan et al. 2011] và giảm kích thước [Yuan et al. 2014]. Tuy nhiên, cả CCA bán ghép và đa dạng xem đều nằm ngoài phạm vi của hướng dẫn này.

Hướng dẫn này bắt đầu với phần giới thiệu về công thức ban đầu của CCA. Các khuôn khổ cơ bản và các giả định thống kê được trình bày. Các kỹ thuật để giải quyết vấn đề tối ưu hóa CCA được thảo luận. Sau khi giải quyết vấn đề CCA, các phương pháp tiếp cận để giải thích và đánh giá kết quả được giải thích. Các biến thể của CCA được minh họa bằng cách sử dụng các ví dụ đã làm việc. Trong số các phiên bản mở rộng của CCA, hướng dẫn tập trung vào các chủ đề CCA chính quy, hạt nhân và thưa thớt. Ngoài ra, các biến thể CCA sâu và Bayesian được xem xét ngắn gọn. Hướng dẫn này làm quen với người đọc với các phương pháp tương quan chính tắc, thảo luận về nơi chúng được áp dụng và loại thông tin nào có thể được trích xuất.

1.2. Mục tiêu:

Phân tích tương quan chính tắc xác định một tập hợp các biến thể chính tắc, kết hợp tuyến tính trực giao của các biến trong mỗi tập hợp để giải thích tốt nhất sự biến đổi cả trong và giữa các tập hợp. Ngoài ra, đây là một phương pháp đa biến nhằm mục đích giảm cấu trúc tương quan giữa hai tập hợp các biến về dạng đơn giản nhất có thể, thông qua các phép biến đổi tuyến tính của các biến trong mỗi tập hợp.

CCA còn được sử dụng để xác định và đo lường mối liên hệ giữa hai tập hợp các biến. Nói một cách đơn giản, với hai tập hợp biến, phương pháp tìm kiếm các hỗn hợp tuyến tính trong mỗi tập hợp, sao cho mỗi hỗn hợp thu được từ một tập hợp này có tương quan tối đa với một hỗn hợp tương ứng từ tập hợp kia, nhưng không tương quan với tất cả các hỗn hợp khác trong một trong hai tập hợp.

1.3. Nguyên lý:

Phân tích tương quan chính tắc dựa trên các chỉ số tuyến tính, như là tổ hợp tuyến tính của các biến ngẫu nhiên. Trong phân tích thống kê đa biến, dữ liệu bao gồm nhiều biến được đo lường trên một tập hợp các quan sát hoặc các cá thể. Trong trường hợp CCA, các biến của một quan sát có thể được phân chia thành hai tập hợp có thể được coi là hai góc nhìn của dữ liệu.

Phân tích Tương quan Chính tắc tìm kiếm vectơ $\boldsymbol{a}$ và $\boldsymbol{b}$ sao cho mối quan hệ của hai chỉ số $\boldsymbol{U} = \boldsymbol{a}^T\boldsymbol{X}$ và $\boldsymbol{V} = \boldsymbol{b}^T\boldsymbol{Y}$ được lượng hóa theo một cách nào đó có thể giải thích được. Chính xác hơn, ta đang tìm kiếm các vector hệ số $\boldsymbol{a}$ và $\boldsymbol{b}$ sao cho chúng tối đa hóa mối tương quan giữa 2 chỉ số:

$$\boldsymbol{p}^* = \max_{a,b} \boldsymbol{Corr}(\boldsymbol{U},\boldsymbol{V}) = \max_{a,b} \frac{\boldsymbol{Cov}(\boldsymbol{U},\boldsymbol{V})}{\sqrt{\boldsymbol{Var}(\boldsymbol{U})}\sqrt{\boldsymbol{Var}(\boldsymbol{V})}} = \max_{a,b} \frac{\boldsymbol{a}^T \sum_{\boldsymbol{XY}} \boldsymbol{b}}{\sqrt{\boldsymbol{a}^T \sum_{\boldsymbol{XX}} \boldsymbol{a}}\sqrt{\boldsymbol{b}^T \sum_{\boldsymbol{YY}} \boldsymbol{b}}}$$

1.4. Phát biểu Bài toán:

Phân tích tương quan chính tắc đo lường mối liên hệ giữa hai nhóm biến qua hai vectơ ngẫu nhiên có $\boldsymbol{p}-$ và $\boldsymbol{q}-$ chiều ($\boldsymbol{p} \leq \boldsymbol{q}$):

$$\boldsymbol{X} = \begin{bmatrix} \boldsymbol{X_1} \\ \boldsymbol{X_2} \\ \vdots \\ \boldsymbol{X_p} \end{bmatrix} \in \mathbb{R}^{\boldsymbol{p*n}} \text{ và } \boldsymbol{Y} = \begin{bmatrix} \boldsymbol{Y_1} \\ \boldsymbol{Y_2} \\ \vdots \\ \boldsymbol{Y_q} \end{bmatrix} \in \mathbb{R}^{\boldsymbol{q*n}}$$

với các vectơ ngẫu nhiên, ta có:

- $\boldsymbol{\mu_X} = \boldsymbol{E}(\boldsymbol{X})$ và $\boldsymbol{\mu_Y} = \boldsymbol{E}(\boldsymbol{Y})$ lần lượt là giá trị trung bình của $\boldsymbol{X}$ và $\boldsymbol{Y}$.
- $\sum_{\boldsymbol{XX}} = \boldsymbol{Cov}(\boldsymbol{X})$ và $\sum_{\boldsymbol{YY}} = \boldsymbol{Cov}(\boldsymbol{Y})$ lần lượt là ma trận hiệp phương sai của $\boldsymbol{X}$ và $\boldsymbol{Y}$ và $\sum_{\boldsymbol{XY}} = \boldsymbol{Cov}(\boldsymbol{XY}) = \sum_{\boldsymbol{YX}}{}^{\boldsymbol{T}}$ là ma trận hiệp phương sai giữa $\boldsymbol{X}$ và $\boldsymbol{Y}$.

Liên kết hai ma trận ngẫu nhiên $\boldsymbol{X}$ và $\boldsymbol{Y}$ sử dụng Phân vùng Ma trận Hiệp phương sai:

$$\underset{((p+q)*1)}{XY} = \begin{bmatrix} X \\ Y \end{bmatrix} = \begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_p \\ Y_1 \\ Y_2 \\ \vdots \\ Y_q \end{bmatrix}$$

có vector trung bình:

$$\underset{((p+q)*1)}{\mu} = E(XY) = \begin{bmatrix} E(X) \\ E(Y) \end{bmatrix} = \begin{bmatrix} \mu_X \\ \mu_Y \end{bmatrix}$$

và ma trận hiệp phương sai có $p * q$ entries tương ứng với tất cả các hiệp phương sai riêng giữa các thành phần của X và các thành phần của Y:

$$\underset{((p+q)*(p+q))}{\sum} = E(XY - \mu)(XY - \mu)^T = \begin{bmatrix} E(X-\mu)(X-\mu)^T & E(X-\mu)(Y-\mu)^T \\ E(Y-\mu)(X-\mu)^T & E(Y-\mu)(Y-\mu)^T \end{bmatrix} = \begin{bmatrix} \sum_{XX} & \sum_{XY} \\ \sum_{YX} & \sum_{YY} \end{bmatrix}$$

Giảm thứ nguyên (Dimension Reduction):

Đặt $U = a^T X$ và $V = b^T Y$ lần lượt là tổ hợp tuyến tính của X và Y và tìm vectors $a \in \mathbb{R}^p$, $b \in \mathbb{R}^q$ để $Corr(U,V)$ được tối đa hóa, ta có:

- $Var(U) = a^T \sum_X a$
- $Var(V) = b^T \sum_Y b$
- $Cov(U,V) = a^T \sum_{XY} b$

Từ đó, tương quan giữa U và V là:

$$Corr(U,V) = \frac{Cov(U,V)}{\sqrt{Var(U)}\sqrt{Var(V)}} = \frac{a^T \sum_{XY} b}{\sqrt{a^T \sum_X a}\sqrt{b^T \sum_Y b}}$$

Cặp phương sai chính tắc đầu tiên là cặp kết hợp tuyến tính U_1, V_1 với phương sai đơn vị sao cho $Corr(U_1, V_1)$ là cực đại. Cặp phương sai chính tắc thứ k là cặp kết hợp tuyến tính U_k, V_k với phương sai đơn vị sao cho $Corr(U_k, V_k)$ là cực đại trong tất cả các cặp không tương quan với $k-1$ cặp trước đó.

$$\begin{aligned} U_1 &= a_{11}X_1 + a_{12}X_2 + \cdots + a_{1p}X_p \\ U_2 &= a_{21}X_1 + a_{22}X_2 + \cdots + a_{2p}X_p \\ &\vdots \\ U_p &= a_{p1}X_1 + a_{p2}X_2 + \cdots + a_{pp}X_p \\ V_1 &= b_{11}Y_1 + b_{12}Y_2 + \cdots + b_{1p}Y_q \\ V_2 &= b_{21}Y_1 + b_{22}Y_2 + \cdots + b_{2p}Y_q \\ &\vdots \\ V_p &= b_{p1}Y_1 + b_{p2}Y_2 + \cdots + b_{pq}Y_q \end{aligned}$$

Khi U_k, V_k là cặp biến thể chính tắc thứ k, ta nói rằng $p_k = Corr(U_k, V_k)$ là tương quan chính tắc thứ k đối với $\tilde{a} = \sum_X^{\frac{1}{2}} a$ và $\tilde{b} = \sum_Y^{\frac{1}{2}} b$, thì tương quan giữa U và V là:

$$Corr(U,V) = \frac{Cov(U,V)}{\sqrt{Var(U)}\sqrt{Var(V)}} = \frac{a^T \sum_{XY} b}{\sqrt{a^T \sum_X a}\sqrt{b^T \sum_Y b}} = \frac{\tilde{a}^T \sum_X^{-\frac{1}{2}} \sum_{XY} b \sum_Y^{-\frac{1}{2}} \tilde{b}}{\sqrt{\tilde{a}^T \tilde{a}}\sqrt{\tilde{b}^T \tilde{b}}}$$

Đặt điều kiện biên: với $k, l = 1, 2, \ldots, p$

$$\begin{aligned} Var(U_k) &= Var(V_k) = 1 \\ Cov(U_k, U_l) &= Corr(U_k, U_L) = 0, k \neq l \\ Cov(V_k, V_l) &= Corr(V_k, V_L) = 0, k \neq l \\ Cov(U_k, V_l) &= Corr(U_k, V_L) = 0, k \neq l \end{aligned}$$

Đặt $\boldsymbol{M} = \sum_{X}^{-\frac{1}{2}} \sum_{XY} \sum_{Y}^{-\frac{1}{2}}$, ta có:

$$\max_{a,b} \boldsymbol{Corr}(\boldsymbol{a}^T\boldsymbol{X}, \boldsymbol{b}^T\boldsymbol{Y}) \Leftrightarrow \max_{\tilde{a},\tilde{b}:\|\tilde{a}\|=1,\|\tilde{b}\|=1} \tilde{\boldsymbol{a}}^T\boldsymbol{M}\tilde{\boldsymbol{b}}$$

Giải pháp cho vấn đề tối đa hóa này liên quan đến sự phân tách giá trị đơn lẻ của $\boldsymbol{M}$. Tương tự, nó liên quan đến sự phân tách riêng của $\boldsymbol{MM^T}$

$$\boldsymbol{MM^T} = \sum_{X}^{-\frac{1}{2}} \sum_{XY} \sum_{Y}^{-1} \sum_{YX} \sum_{X}^{-\frac{1}{2}}$$

1.5. Phương pháp:

1.5.1. Phân tích Tương quan Chính tắc Quần thể:

Mặc dù các biến chính tắc là nhân tạo, chúng thường có thể được "xác định" về các biến chủ đề. Những lần nhận dạng này được hỗ trợ bằng cách tính toán mối tương quan giữa các biến chính tắc và các biến gốc. Tuy nhiên, những mối tương quan này phải được giải thích một cách thận trọng vì chúng chỉ cung cấp thông tin đơn biến, theo nghĩa là chúng không chỉ ra cách các biến ban đầu gộp chung cho các phân tích chính tắc. Vì lý do này, các biến ban đầu thường được đánh giá sự đóng góp trực tiếp từ các hệ số tiêu chuẩn hóa.

Cho trước hai nhóm biến ngẫu nhiên có $\boldsymbol{p}-$ và $\boldsymbol{q}-$ chiều ($\boldsymbol{p} \leq \boldsymbol{q}$):

$$\boldsymbol{X} = \begin{bmatrix} \boldsymbol{X_1} \\ \boldsymbol{X_2} \\ \vdots \\ \boldsymbol{X_p} \end{bmatrix} \in \mathbb{R}^{\boldsymbol{p*n}} \text{ và } \boldsymbol{Y} = \begin{bmatrix} \boldsymbol{Y_1} \\ \boldsymbol{Y_2} \\ \vdots \\ \boldsymbol{Y_q} \end{bmatrix} \in \mathbb{R}^{\boldsymbol{q*n}}$$

Đặt $\boldsymbol{\lambda_1} \geq \cdots \geq \boldsymbol{\lambda_p}$ là các giá trị riêng của $\sum_{X}^{-\frac{1}{2}} \sum_{XY} \sum_{Y}^{-1} \sum_{YX} \sum_{X}^{-\frac{1}{2}}$ với $\boldsymbol{e_1}, \ldots, \boldsymbol{e_p}$ là các vector riêng (eigenvector) tương ứng với định mức đơn vị.

Đặt $\boldsymbol{\lambda_1} \geq \cdots \geq \boldsymbol{\lambda_p}$ là các giá trị riêng $\boldsymbol{p}$ lớn nhất của $\sum_{Y}^{-\frac{1}{2}} \sum_{YX} \sum_{X}^{-1} \sum_{XY} \sum_{Y}^{-\frac{1}{2}}$ với $\boldsymbol{f_1}, \ldots, \boldsymbol{f_p}$ là các vector riêng (eigenvector) tương ứng với định mức đơn vị.

Tại cặp thứ $\boldsymbol{k}$ của các biến thể chính tắc: $\boldsymbol{U_k} = \boldsymbol{e_k^T} \sum_{X}^{-\frac{1}{2}} \boldsymbol{X}$ và $\boldsymbol{V_k} = \boldsymbol{f_k^T} \sum_{Y}^{-\frac{1}{2}} \boldsymbol{Y}$, với $\boldsymbol{e_k^T} \sum_{X}^{-\frac{1}{2}}$ và $\boldsymbol{f_k^T} \sum_{Y}^{-\frac{1}{2}}$ là các hướng chính tắc, thì tương quan giữa $\boldsymbol{U_k}$ và $\boldsymbol{V_k}$:

$$\boldsymbol{p_k} = \boldsymbol{Corr}(\boldsymbol{U_k}, \boldsymbol{V_k}) = \sqrt{\boldsymbol{\lambda_k}}$$

CHỨNG MINH:

Giả sử: $\boldsymbol{p_1} = \frac{\tilde{\boldsymbol{a}}^T\boldsymbol{M}\tilde{\boldsymbol{b}}}{\sqrt{\tilde{\boldsymbol{a}}^T\tilde{\boldsymbol{a}}}\sqrt{\tilde{\boldsymbol{b}}^T\tilde{\boldsymbol{b}}}}$

Áp dụng bất đẳng thức Cauchy-Schwartz cho tử số của $\boldsymbol{p_1}^2$:

$$\left(\tilde{\boldsymbol{a}}^T\boldsymbol{M}\tilde{\boldsymbol{b}}\right)^2 \leq \left(\tilde{\boldsymbol{a}}^T\tilde{\boldsymbol{a}}\right)\left(\tilde{\boldsymbol{b}}^T\boldsymbol{M}^T\boldsymbol{M}\tilde{\boldsymbol{b}}\right)$$

hoặc $\left(\tilde{\boldsymbol{a}}^T\boldsymbol{M}\tilde{\boldsymbol{b}}\right)^2 = \left(\tilde{\boldsymbol{a}}^T\tilde{\boldsymbol{a}}\right)\left(\tilde{\boldsymbol{b}}^T\boldsymbol{M}^T\boldsymbol{M}\tilde{\boldsymbol{b}}\right)$ nếu tồn tại một đại lượng $\boldsymbol{C}$ vô hướng sao cho: $\tilde{\boldsymbol{a}} = \boldsymbol{C}\boldsymbol{M}\tilde{\boldsymbol{b}}$

Từ bất đẳng thức Cauchy-Schwartz trên:

$$\boldsymbol{p_1}^2 \leq \frac{\left(\tilde{\boldsymbol{a}}^T\tilde{\boldsymbol{a}}\right)\left(\tilde{\boldsymbol{b}}^T\boldsymbol{M}^T\boldsymbol{M}\tilde{\boldsymbol{b}}\right)}{\left(\tilde{\boldsymbol{a}}^T\tilde{\boldsymbol{a}}\right)\left(\tilde{\boldsymbol{b}}^T\tilde{\boldsymbol{b}}\right)} = \frac{\left(\tilde{\boldsymbol{b}}^T\boldsymbol{M}^T\boldsymbol{M}\tilde{\boldsymbol{b}}\right)}{\left(\tilde{\boldsymbol{b}}^T\tilde{\boldsymbol{b}}\right)}$$

Tối đa hóa tỉ lệ của $\boldsymbol{p_1}^2$ bằng cách chọn $\tilde{\boldsymbol{b}}$ là vector riêng (eigenvector) tương quan với giá trị riêng (eigenvalue) lớn nhất $\boldsymbol{\lambda_1}$ của $\boldsymbol{M^TM}$. Từ đó: $\boldsymbol{MM^T}\tilde{\boldsymbol{a}} = \boldsymbol{MM^T}\left(\boldsymbol{CM}\tilde{\boldsymbol{b}}\right) = \boldsymbol{CM}\left(\boldsymbol{M^TM}\tilde{\boldsymbol{b}}\right) = \boldsymbol{CM}\left(\boldsymbol{\lambda_1}\tilde{\boldsymbol{b}}\right) = \boldsymbol{\lambda_1}\left(\boldsymbol{CM}\tilde{\boldsymbol{b}}\right) = \boldsymbol{\lambda_1}\tilde{\boldsymbol{a}}$. Nói cách khác, khi $\boldsymbol{p_1}^2$ đạt được cực đại, $\tilde{\boldsymbol{a}}$ bằng với vector riêng tương quan với giá trị riêng lớn nhất $\boldsymbol{\lambda_1}$ của $\boldsymbol{MM^T}$. Do đó, nếu $\tilde{\boldsymbol{a}} = \boldsymbol{e_1}$ và $\tilde{\boldsymbol{b}} = \boldsymbol{f_1}$ thì $\boldsymbol{a} = \boldsymbol{e_1} \sum_{X}^{-\frac{1}{2}} \boldsymbol{X}$ và $\boldsymbol{b} = \boldsymbol{f_1} \sum_{Y}^{-\frac{1}{2}} \boldsymbol{Y}$. Chính tắc tiếp theo thu được bằng cách áp đặt một ràng buộc trực giao và lặp lại phân tích này.

1.5.2. Phân tích Tương quan Chính tắc cho Mẫu:

Cho trước hai nhóm biến ngẫu nhiên có $\boldsymbol{p}-$ và $\boldsymbol{q}-$ chiều $(\boldsymbol{p} \leq \boldsymbol{q} < \boldsymbol{n})$ với $(\boldsymbol{X_i}, \boldsymbol{Y_i})$ được lấy mẫu chung, tức là trên cùng một đơn vị thử nghiệm.

$$\boldsymbol{X} = \begin{bmatrix} \boldsymbol{X_1} \\ \boldsymbol{X_2} \\ \vdots \\ \boldsymbol{X_p} \end{bmatrix} \in \mathbb{R}^{\boldsymbol{p*n}} \text{ và } \boldsymbol{Y} = \begin{bmatrix} \boldsymbol{Y_1} \\ \boldsymbol{Y_2} \\ \vdots \\ \boldsymbol{Y_q} \end{bmatrix} \in \mathbb{R}^{\boldsymbol{q*n}}$$

Một mẫu ngẫu nhiên của các quan sát $\boldsymbol{n}$ trên mỗi $(\boldsymbol{p}+\boldsymbol{q})$ nhóm biến $\boldsymbol{X}$ và $\boldsymbol{Y}$ có thể được lắp ráp vào matrix dữ liệu $\boldsymbol{n} * (\boldsymbol{p}+\boldsymbol{q})$.

$$\boldsymbol{XY} = [\boldsymbol{X} \quad \boldsymbol{Y}] = \begin{bmatrix} \boldsymbol{X_{11}} & \boldsymbol{X_{12}} & \cdots & \boldsymbol{X_{1p}} & \boldsymbol{Y_{11}} & \boldsymbol{Y_{12}} & \cdots & \boldsymbol{Y_{1q}} \\ \boldsymbol{X_{21}} & \boldsymbol{X_{22}} & \cdots & \boldsymbol{X_{2p}} & \boldsymbol{Y_{21}} & \boldsymbol{Y_{22}} & \cdots & \boldsymbol{Y_{2q}} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \boldsymbol{X_{n1}} & \boldsymbol{X_{n2}} & \cdots & \boldsymbol{X_{np}} & \boldsymbol{Y_{n1}} & \boldsymbol{Y_{n2}} & \cdots & \boldsymbol{Y_{nq}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{X_1}^T & \boldsymbol{Y_1}^T \\ \vdots & \vdots \\ \boldsymbol{X_n}^T & \boldsymbol{Y_n}^T \end{bmatrix}$$

với mỗi nhóm biến ngẫu nhiên, ta có:

- $\overline{\boldsymbol{X}} = \frac{\boldsymbol{1}}{\boldsymbol{n}} \sum_{\boldsymbol{j=1}}^{\boldsymbol{n}} \boldsymbol{X_j}$ và $\overline{\boldsymbol{Y}} = \frac{\boldsymbol{1}}{\boldsymbol{n}} \sum_{\boldsymbol{j=1}}^{\boldsymbol{n}} \boldsymbol{Y_j}$ lần lượt là giá trị trung bình mẫu của $\boldsymbol{X}$ và $\boldsymbol{Y}$, từ đó có vector trung bình mẫu:

$$\overline{\boldsymbol{XY}} = \begin{bmatrix} \overline{\boldsymbol{X}} \\ \overline{\boldsymbol{Y}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\frac{1}{n} \sum_{j=1}^{n} X_j} \\ \boldsymbol{\frac{1}{n} \sum_{j=1}^{n} Y_j} \end{bmatrix}$$

- $\boldsymbol{S_X}$ và $\boldsymbol{S_Y}$ lần lượt là hiệp phương sai mẫu của $\boldsymbol{X}$ và $\boldsymbol{Y}$.

$$\boldsymbol{S_{XY}} = \boldsymbol{\frac{1}{n-1} \sum_{j=1}^{n} (X_j - \overline{X})(Y_j - \overline{Y})^T}$$

từ đó có ma trận hiệp phương sai mẫu có $\boldsymbol{p} * \boldsymbol{q}$ entries tương ứng với tất cả các hiệp phương sai mẫu riêng giữa các thành phần của $\boldsymbol{X}$ và các thành phần của $\boldsymbol{Y}$:

$$\boldsymbol{S} = \boldsymbol{S}(\boldsymbol{XY} - \overline{\boldsymbol{XY}})(\boldsymbol{XY} - \overline{\boldsymbol{XY}})^T = \begin{bmatrix} \boldsymbol{S}(\boldsymbol{X} - \overline{\boldsymbol{XY}})(\boldsymbol{X} - \overline{\boldsymbol{XY}})^T & \boldsymbol{S}(\boldsymbol{X} - \overline{\boldsymbol{XY}})(\boldsymbol{Y} - \overline{\boldsymbol{XY}})^T \\ \boldsymbol{S}(\boldsymbol{Y} - \overline{\boldsymbol{XY}})(\boldsymbol{X} - \overline{\boldsymbol{XY}})^T & \boldsymbol{S}(\boldsymbol{Y} - \overline{\boldsymbol{XY}})(\boldsymbol{Y} - \overline{\boldsymbol{XY}})^T \end{bmatrix} = \begin{bmatrix} \boldsymbol{S_{XX}} & \boldsymbol{S_{XY}} \\ \boldsymbol{S_{YX}} & \boldsymbol{S_{YY}} \end{bmatrix}$$

Các kết hợp tuyến tính $\widehat{\boldsymbol{U}} = \widehat{\boldsymbol{a}}^{\boldsymbol{T}} \boldsymbol{X}$ và $\widehat{\boldsymbol{V}} = \widehat{\boldsymbol{b}}^{\boldsymbol{T}} \boldsymbol{Y}$ có mối tương quan mẫu:

$$r_{\widehat{U},\widehat{V}} = \frac{\widehat{\boldsymbol{a}}^{\boldsymbol{T}} \boldsymbol{S_{XY}} \widehat{\boldsymbol{b}}}{\sqrt{\widehat{\boldsymbol{a}}^{\boldsymbol{T}} \boldsymbol{S_{XX}} \widehat{\boldsymbol{a}}} \sqrt{\widehat{\boldsymbol{b}}^{\boldsymbol{T}} \boldsymbol{S_{YY}} \widehat{\boldsymbol{b}}}}$$

Cặp phương sai mẫu chính tắc đầu tiên là cặp kết hợp tuyến tính $\widehat{\boldsymbol{U_1}}, \widehat{\boldsymbol{V_1}}$ với phương sai đơn vị sao cho $\boldsymbol{r}(\widehat{\boldsymbol{U_1}}, \widehat{\boldsymbol{V_1}})$ là cực đại. Cặp phương sai chính tắc thứ $\boldsymbol{k}$ là cặp kết hợp tuyến tính $\widehat{\boldsymbol{U_k}}, \widehat{\boldsymbol{V_k}}$ với phương sai đơn vị sao cho $\boldsymbol{r}(\widehat{\boldsymbol{U_k}}, \widehat{\boldsymbol{V_k}})$ là cực đại trong tất cả các cặp không tương quan với $\boldsymbol{k-1}$ cặp trước đó.

Đặt $\widehat{\boldsymbol{\lambda_1}} \geq \cdots \geq \widehat{\boldsymbol{\lambda_p}}$ là các giá trị riêng của $\boldsymbol{S_X^{-\frac{1}{2}} S_{XY} S_Y^{-1} S_{YX} S_X^{-\frac{1}{2}}}$ với $\widehat{\boldsymbol{e_1}}, \ldots, \widehat{\boldsymbol{e_p}}$ là các vector riêng (eigenvector) tương ứng với định mức đơn vị.

Đặt $\widehat{\boldsymbol{\lambda_1}} \geq \cdots \geq \widehat{\boldsymbol{\lambda_p}}$ là các giá trị riêng $\boldsymbol{p}$ lớn nhất của $\boldsymbol{S_Y^{-\frac{1}{2}} S_{YX} S_X^{-1} S_{XY} S_Y^{-\frac{1}{2}}}$ với $\widehat{\boldsymbol{f_1}}, \ldots, \widehat{\boldsymbol{f_p}}$ là các vector riêng (eigenvector) tương ứng với định mức đơn vị.

Tại cặp thứ $\boldsymbol{k}$ của các biến thể chính tắc trên mẫu: $\widehat{\boldsymbol{U_k}} = \boldsymbol{X S_X^{-\frac{1}{2}} \widehat{e_k}}$ và $\widehat{\boldsymbol{V_k}} = \boldsymbol{Y S_Y^{-\frac{1}{2}} \widehat{f_k}}$, với $\boldsymbol{S_X^{-\frac{1}{2}} \widehat{e_k}}$ và $\boldsymbol{S_Y^{-\frac{1}{2}} \widehat{f_k}}$ là các hướng chính tắc, thì tương quan giữa $\widehat{\boldsymbol{U_k}}$ và $\widehat{\boldsymbol{V_k}}$:

$$\widehat{\boldsymbol{p_k}} = \boldsymbol{Corr}(\widehat{\boldsymbol{U_k}}, \widehat{\boldsymbol{V_k}}) = \sqrt{\widehat{\boldsymbol{\lambda_k}}}$$

1.5.3. Suy diễn mẫu lớn:

CCA giải thích cho mối tương quan giữa $\boldsymbol{X}$ và $\boldsymbol{Y}$, nếu không có tương quan thì $\boldsymbol{\Sigma_{XY} = 0}$. Để đánh giá tính độc lập của X và Y, sử dụng Likelihood-ratio test.

Likelihood-ratio test cho $\boldsymbol{\Sigma_{XY} = 0}$:

Đặt $(\boldsymbol{X_i, Y_i})$ với $\boldsymbol{i = 1, 2, \dots, n}$ là mẫu ngẫu nhiên từ phân phối chuẩn $\boldsymbol{N_{p+q}(\mu, \Sigma)}$ với $\boldsymbol{\Sigma = \begin{bmatrix} \Sigma_{XX} & \Sigma_{XY} \\ \Sigma_{YX} & \Sigma_{YY} \end{bmatrix}}$.

Đặt $\boldsymbol{S_X}$ và $\boldsymbol{S_Y}$ lần lượt là hiệp phương sai mẫu của $\boldsymbol{X_1, X_2, .., X_n}$ và $\boldsymbol{Y_1, Y_2, \dots, Y_n}$, với $\boldsymbol{S_n}$ là hiệp phương sai mẫu có $\boldsymbol{(p+q)}$ − chiều của $\boldsymbol{(X_i, Y_i)}$.

Do đó Likelihood-ratio test $\boldsymbol{H_0: \Sigma_{XY} = 0}$ so với $\boldsymbol{H_1: \Sigma_{XY} \neq 0}$ từ chối $\boldsymbol{H_0}$ cho giá trị lớn hơn của

$$\boldsymbol{-2 \log \Lambda = n \log \frac{|S_X||S_Y|}{|S_n|} = -n \log \prod_{i=1}^{p} (1 - \hat{p}_i^2)}$$

với $\boldsymbol{\hat{p}_1, \hat{p}_2, \dots, \hat{p}_p}$ là tương quan chính tắc mẫu và $\boldsymbol{S_n = \begin{bmatrix} S_{XX} & S_{XY} \\ S_{YX} & S_{YY} \end{bmatrix}}$ là ước tính không thiên vị của $\boldsymbol{\Sigma}$:

$$\boldsymbol{|S_n| = |S_Y||S_X - S_{XY} S_Y^{-1} S_{YX}|}$$

Thống kê likelihood-ratio so sánh phương sai tổng quát mẫu dưới $\boldsymbol{H_0}$ như là $\boldsymbol{\begin{vmatrix} S_{XX} & 0 \\ 0^T & S_{YY} \end{vmatrix} = |S_{XX}||S_{YY}|}$ với phương sai tổng quát không giới hạn $\boldsymbol{|S_n|}$:

$$\boldsymbol{\Lambda^{\frac{2}{n}} = \frac{|S_n|}{|S_X||S_Y|} = \frac{|S_Y||S_X - S_{XY} S_Y^{-1} S_{YX}|}{|S_X||S_Y|} = \frac{|S_X - S_{XY} S_Y^{-1} S_{YX}|}{|S_X|} = |I_p - S_{XY} S_Y^{-1} S_{YX} S_X^{-1}|}$$

$$\boldsymbol{= \left| I_p - S_X^{-\frac{1}{2}} S_{XY} S_Y^{-1} S_{YX} S_X^{-\frac{1}{2}} \right| = |I_p - \hat{M}\hat{M}^T|}$$

vì giá trị riêng của $\boldsymbol{\hat{M}\hat{M}^T}$ là $\boldsymbol{\hat{p}_1^2 > \hat{p}_2^2 > \cdots > \hat{p}_{p^T}^2}$, từ đó ta có:

$$\boldsymbol{\Lambda^{\frac{2}{n}} = \prod_{i=1}^{p} (1 - \hat{p}_i^2)}$$

Phân phối rỗng (Null Distribution)

Đối với một số n lớn, thống kê thử nghiệm được phân bố xấp xỉ dưới dạng một biến ngẫu nhiên chi-square:

$$\boldsymbol{\frac{(p+q)(p+q+1)}{2} - \frac{p(p+1)}{2} + \frac{q(q+1)}{2} = pq}$$

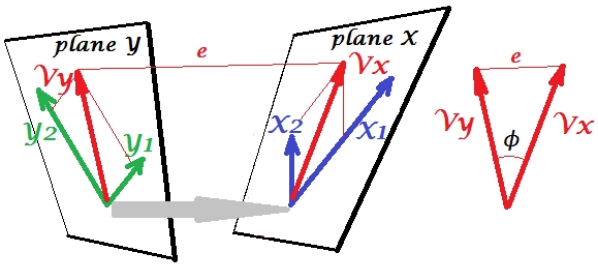
Bartlett's sử dụng một phân phối khác:

$$\boldsymbol{-(n - 1 - \frac{1}{2}(p+q+1)) \, log \prod_{i=1}^{p} (1 - \hat{p}_i^2)}$$

Từ đó, kiểm tra giả định thứ $\boldsymbol{k}$ sử dụng một cách ngắn hơn của thống kê likelihood-ratio test trên $\boldsymbol{(p-k)(q-k)}$ bậc tự do:

$$\boldsymbol{LRT_k = -(n - 1 - \frac{1}{2}(p+q+1)) \, log \prod_{i=k+1}^{p} (1 - \hat{p}_i^2)}$$

1.5.4. Ý nghĩa Hình học:



Xét trên mặt phẳng X và mặt phẳng Y, mặt phẳng X và Y chỉ cắt nhau tại một điểm của góc toạ độ. Mỗi biến thể chính tắc là sự kết hợp tuyến tính của các biến tương ứng. Cụ thể, V_x là một hình chiếu của V_y trên mặt phẳng X và đồng thời V_y là một hình chiếu của V_x trên mặt phẳng Y, nhưng chúng không phải là phép chiếu trực giao. Thay vào đó, chúng được tìm thấy (trích xuất) để giảm thiểu góc ϕ giữa chúng. Cosine của ϕ là tương quan chính tắc.

Vì các phép chiếu không cần phải trực giao, độ dài (do đó là phương sai) của các biến thể chuẩn không được xác định tự động bởi thuật toán so khớp và phải tuân theo các quy ước / ràng buộc có thể khác nhau trong các cách triển khai khác nhau. Trong CCA, các cặp biến thiên tương quan tối đa tương quan lẫn nhau được trích xuất cho đến khi tất cả biến thiên đa biến có thể được dự đoán trong tập hợp nhỏ hơn được thiết lập

Cho trước hai tập $\boldsymbol{A}$ và $\boldsymbol{B}$ với các dòng là hướng chính tắc:

$$\boldsymbol{A} = \begin{pmatrix} \boldsymbol{e_1^T \sum_X^{-\frac{1}{2}}} & \dots & \boldsymbol{e_p^T \sum_X^{-\frac{1}{2}}} \end{pmatrix}^{\boldsymbol{T}} \text{ và } \boldsymbol{B} = \begin{pmatrix} \boldsymbol{f_1^T \sum_Y^{-\frac{1}{2}}} & \dots & \boldsymbol{f_q^T \sum_Y^{-\frac{1}{2}}} \end{pmatrix}^{\boldsymbol{T}}$$

Sử dụng một phép biến đổi tuyến tính $\boldsymbol{U = AX}$ và $\boldsymbol{V = BY}$ có $\boldsymbol{p}-$ chiều, từ đó có:

$$\boldsymbol{Cov(U,V) = Cov(AX,X) = A \sum_X}$$

với $\boldsymbol{I_p}$ là ma trận đơn vị, $\boldsymbol{\sigma_i^2}$ là phương sai của $\boldsymbol{X_i}$ và $\boldsymbol{Cov(U) = I_p}$, ta có:

$$\boldsymbol{Corr(U_k, X_i) = Cov\left(U_k, \sigma_i^{-1} X_i\right)}$$

Đặt $\boldsymbol{D_X}$ là ma trận đường chéo của với nguyên tố thứ $\boldsymbol{i}$ của đường chéo là $\boldsymbol{\sigma_i = \sqrt{Var(X_i)}}$, ta có

$$\boldsymbol{Corr(U,X) = A \sum_X D_X^{-1}}$$

tương tự, có được:

$$\boldsymbol{Corr(V,Y) = B \sum_Y D_Y^{-1}}$$

$$\boldsymbol{Corr(U,Y) = A \sum_{XY} D_Y^{-1}}$$

$$\boldsymbol{Corr(V,X) = B \sum_{YX} D_X^{-1}}$$

Những đại lượng này cung cấp thông tin về sự đóng góp của các biến gốc vào các biến chính tắc.

Giải thích theo Ý nghĩa Hình học:

Cho $\boldsymbol{A}$ là ma trận có $\boldsymbol{k}$ dòng là hướng chính tắc thứ $\boldsymbol{k}$ của $\boldsymbol{e_k^T \sum_X^{-\frac{1}{2}}}$ và $\boldsymbol{E}$ là ma trận có **k** cột là vector riêng $\boldsymbol{e_k}$, ta có: $\boldsymbol{A = E^T \sum_X^{-\frac{1}{2}}}$, với $\boldsymbol{E^T E = I_p}$. Nhận được các biến chính tắc $\boldsymbol{U_k}$ bằng cách biến đổi $\boldsymbol{X}$ bằng $\boldsymbol{A}$: $\boldsymbol{U = AX}$.

Sử dụng phân tách quang phổ của $\boldsymbol{\sum_X}$ ta có: $\boldsymbol{A = E^T \sum_X^{-\frac{1}{2}} = E^T P_X \wedge_X^{-\frac{1}{2}} P_X^T}$, với $\boldsymbol{P_X}$ bao gồm vector riêng $\boldsymbol{\Sigma_X}$ và $\wedge_X$ là ma trận đường chéo chứa các giá trị riêng. Từ đó,

$$\boldsymbol{U = AX = E^T P_X \wedge_X^{-\frac{1}{2}} P_X^T X}$$

$\boldsymbol{P_X^T X}$: matrix của thành phần chính của $\boldsymbol{X}$.

$\boldsymbol{\wedge_X^{-\frac{1}{2}} P_X^T X}$: chuẩn hóa các thành phần chính để có phương sai đơn vị.

$\boldsymbol{P_X \wedge_X^{-\frac{1}{2}} P_X^T}$: xoay các thành phần chính đã được tiêu chuẩn hóa bằng cách sử dụng một phép biến đổi chỉ liên quan đến $\boldsymbol{\sum_X}$

$\boldsymbol{E^T P_X \wedge_X^{-\frac{1}{2}} P_X^T X}$: xoay kết quả bằng cách sử dụng một phép biến đổi liên quan đến toàn bộ ma trận hiệp phương sai $\sum$

1.6. Giải thuật:

Cho trước tập dữ liệu của 2 tập biến $\boldsymbol{X}$ và $\boldsymbol{Y}$. Các bước thực hiện:

- Bước 0: chuẩn háo các biến mà được phân tích
- Bước 1: tính ma trận $\boldsymbol{\sum_X^{-\frac{1}{2}} \sum_{XY} \sum_Y^{-1} \sum_{YX} \sum_X^{-\frac{1}{2}}}$ hoặc $\boldsymbol{\sum_Y^{-\frac{1}{2}} \sum_{YX} \sum_X^{-1} \sum_{XY} \sum_Y^{-\frac{1}{2}}}$
- Bước 2: tìm $\boldsymbol{\rho_1^{*2} \geq \rho_2^{*2} \geq \cdots \geq \rho_p^{*2}}$
- Bước 3: tính vector hệ số $\boldsymbol{a_k = e_k^T \sum_X^{-\frac{1}{2}}}$ v$\boldsymbol{b_k = f_k^T \sum_Y^{-\frac{1}{2}}}$ à

Cuối cùng, tổ hợp tuyến tính: $\boldsymbol{U_k} = \boldsymbol{a^T X}$ và $\boldsymbol{V_k} = \boldsymbol{b^T Y}$.

Việc thực hiện bước 0 là tuỳ ý, do chuẩn hoá sẽ không ảnh hưởng đến kết quả tương quan chính tắc của phương pháp. Nếu như cần các biến $\boldsymbol{U_k}, \boldsymbol{V_k}$ mới và trực quan hoá chúng thì để cho dễ đọc và dễ hiểu đươc các biến mới đó, nên cần thực hiện chuẩn hoá lại.

1.7. Phân tích Tương quan (CCA vs. PCA)

PCA liên quan đến việc giải thích cấu trúc phương sai-hiệp phương sai của một tập hợp các biến thông qua một vài kết hợp tuyến tính của các biến này. Mục tiêu chung của nó là giảm dữ liệu và giải thích.

PCA có những điểm tương đồng nhất định với CCA, mặc dù PCA thực hiện phân rã ma trận không được giám sát của một tập biến. Thuộc tính chung của PCA và CCA là ràng buộc trực giao được áp đặt trong quá trình khám phá cấu trúc. Tuy nhiên, xét sự khác biệt đáng kể, các công thức PCA giúp giảm thiểu sai số tái tạo giữa tập biến ban đầu và phép chiếu lùi của mỗi quan sát từ các chiều tiềm ẩn của biến thể (Hastie et al., 2015). Thay vào đó, CCA trực tiếp tối ưu hóa sự tương ứng giữa các kích thước tiềm ẩn trực tiếp trong không gian nhúng, hơn là tổn thất tái cấu trúc trong các biến ban đầu. Hơn nữa, PCA có thể được sử dụng để giảm kích thước như một bước tiền xử lý trước CCA.

Ngoài ra, PCA phân tích các tập dữ liệu nhiều biến để tìm các mẫu, trong khi theo CCA, các mẫu được chọn theo cách là dữ liệu được chiếu trên những mẫu đó biểu diễn tối đa mối tương quan.

1.8. Áp dụng:

Phân tích tương quan chính tắc sử dụng tập dữ liệu **PALMER PENGUIN**:

Điều kiện cần: thư viện Pandas, Matplotlib, numpy và seaborn

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
```

Tiếp đến, tải bộ dữ liệu Palmer Penguin và làm bước dọn dẹp là xoá hàng nào có dữ liệu bị thiếu. Bộ dữ liệu Palmer Penguin được tải bằng đường link này.

```
link2data = "https://raw.githubusercontent.com/mwaskom/seaborn-
data/master/penguins.csv"
df = pd.read_csv(link2data)
df =df.dropna()
df.head()
```

	species	island	bill_length_mm	bill_depth_mm	flipper_length_mm	body_mass_g	sex
0	Adelie	Torgersen	39.1	18.7	181.0	3750.0	MALE
1	Adelie	Torgersen	39.5	17.4	186.0	3800.0	FEMALE
2	Adelie	Torgersen	40.3	18.0	195.0	3250.0	FEMALE
4	Adelie	Torgersen	36.7	19.3	193.0	3450.0	FEMALE
5	Adelie	Torgersen	39.3	20.6	190.0	3650.0	MALE

Tập dữ liệu Palmer Peguin có 7 biến:

- Species – 3 loài (Adelie, Chinstrap, Gentoo)
- Island – 3 đảo (Torgersen, Biscoe, Dream)
- Bill length – chiều dài mỏ (mm)
- Bill depth – chiều sâu mỏ (mm)
- Flipper length – chiều dài cánh (mm)
- Body mass (g) – cân nặng
- Sex – giới tính (male, female)

Vì CCA liên quan đến hai tập dữ liệu nhiều biến, thực hiện chia tập dữ liệu thành hai tập liệu X và Y. Tập dữ liệu X có hai biến tương ứng với chiều dài và chiều sâu của mỏ chim cánh cụt.

```
X = df[['bill_length_mm','bill_depth_mm']]
X.head()
```

	bill_length_mm	bill_depth_mm
0	39.1	18.7
1	39.5	17.4
2	40.3	18.0
4	36.7	19.3
5	39.3	20.6

Thực hiện chuẩn hóa các biến bằng cách trừ đi với giá trị trung bình và chia cho độ lệch chuẩn.

```
X_mc = (X-X.mean())/(X.std())
X_mc.head()
```

	bill_length_mm	bill_depth_mm
0	-0.894695	0.779559
1	-0.821552	0.119404
2	-0.675264	0.424091
4	-1.333559	1.084246
5	-0.858123	1.744400

Tập dữ liệu thứ hai Y chứa chiều dài cánh và trọng lượng cơ thể của chim cánh cụt. Đồng thời thực hiện chuẩn hoá tập Y.

```
Y = df[['flipper_length_mm','body_mass_g']]
Y.head()
```

	flipper_length_mm	body_mass_g
0	181.0	3750.0
1	186.0	3800.0
2	195.0	3250.0
4	193.0	3450.0
5	190.0	3650.0

```
Y_mc = (Y-Y.mean())/(Y.std())
Y_mc.head()
```

	flipper_length_mm	body_mass_g
0	-1.424608	-0.567621
1	-1.067867	-0.505525
2	-0.425733	-1.188572
4	-0.568429	-0.940192
5	-0.782474	-0.691811

CCA với Scikit-learn

```
from sklearn.cross_decomposition import CCA
```

Với hai bộ dữ liệu từ Chim cánh cụt, các đặc điểm của Chim cánh cụt rất khác biệt so với loài khác dựa trên loài đó là gì. Và trong ví dụ, loài là một biến tiềm ẩn chung của hai tập dữ liệu X và Y.

Xét module CCA từ sklearn.cross_decomposition, thực hiện khởi tạo object CCA, và sử dụng hai hàm fit() và transform() với hai ma trận được chuẩn hoá để thực hiện CCA. Và kết quả là hai ma trận tương quan chính tắc.

```
ca = CCA()
ca.fit(X_mc, Y_mc)
X_c, Y_c = ca.transform(X_mc, Y_mc)
print(X_c.shape)
print(Y_c.shape)
```

(333, 2)
(333, 2)

Để hiểu rõ hơn về kết quả, đầu tiên sử dụng cặp hiệp biến chính tắc (canonical covariates) và xem chúng tương quan với nhau như thế nào. Thêm kết quả từ CCA với dữ liệu tương ứng với dữ liệu Chim cánh cụt vào khung dữ liệu Pandas.

```
cc_res = pd.DataFrame({"CCX_1":X_c[:, 0],
                       "CCY_1":Y_c[:, 0],
                       "CCX_2":X_c[:, 1],
                       "CCY_2":Y_c[:, 1],
                       "Species":df.species.tolist(),
                       "Island":df.island.tolist(),
                       "sex":df.sex.tolist()})
cc_res.head()
```

	CCX_1	CCY_1	CCX_2	CCY_2	Species	Island	sex
0	-1.186252	-1.408795	-0.010367	0.682866	Adelie	Torgersen	MALE
1	-0.709573	-1.053857	-0.456036	0.429879	Adelie	Torgersen	FEMALE
2	-0.790732	-0.393550	-0.130809	-0.839620	Adelie	Torgersen	FEMALE
3	-1.718663	-0.542888	-0.073623	-0.458571	Adelie	Torgersen	FEMALE
4	-1.772295	-0.763548	0.736248	-0.014204	Adelie	Torgersen	MALE

Kiểm tra xem cặp hiệp biến chính tắc đầu tiên có tương quan như thế nào. Sử dụng hàm corrcoef () của NumPy để tính toán mối tương quan. Có thể thấy rằng cặp hiệp biến chính tắc đầu tiên có tương quan cao.

```
np.corrcoef(X_c[:, 0], Y_c[:, 0])
```

array([[1. , 0.78763151], [0.78763151, 1.]])

Ngoài ra, có thể tính toán mối tương quan giữa cặp hiệp biến thứ hai, thấy rằng mối tương quan không cao như cặp đầu tiên.

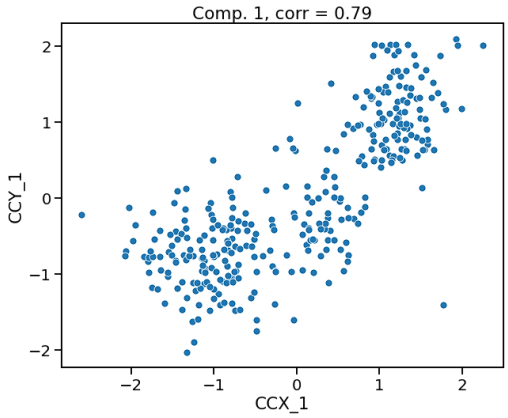
```
np.corrcoef(X_c[:, 1], Y_c[:, 1])
```

array([[1. , 0.08638695], [0.08638695, 1.]])

Để hiểu sâu hơn về mối quan hệ giữa các cặp hiệp biến chính tắc, thực hiện lập biểu đồ phân tán với cặp biến chính quy đầu tiên. Có thể thấy mối tương quan cao giữa cặp biến chính quy đầu tiên.

```
sns.set_context("talk", font_scale=1.2)
plt.figure(figsize=(10,8))
sns.scatterplot(x="CCX_1", y="CCY_1", data=cc_res)
plt.title('Comp. 1, corr = %.2f' %
```

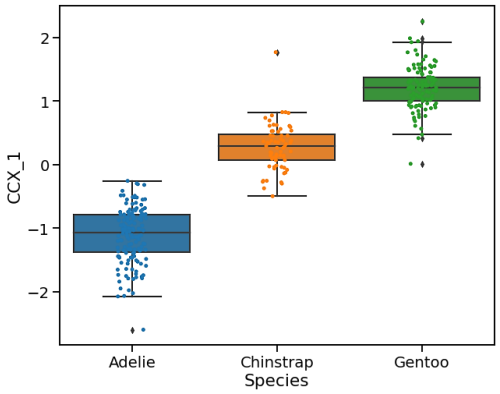
```
        np.corrcoef(X_c[:, 0], Y_c[:, 0])[0, 1])
```



Suy diễn các biến số chính tắc với Heatmap

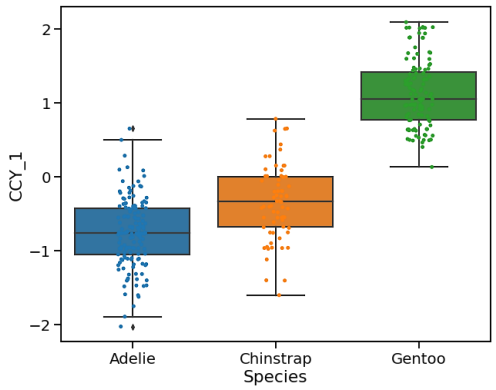
Trong ví dụ này, biết rằng biến loài trong tập dữ liệu là biến ẩn. Cùng xem biến tiềm ẩn có tương quan như thế nào với cặp hiệp biến chính tắc đầu tiên. Đầu tiên, thực hiện tạo một biểu đồ hình hộp giữa biến tiềm ẩn và từng cặp hiệp biến chính tắc đầu tiên.

```
plt.figure(figsize=(10,8))
sns.boxplot(x="Species",
                y="CCX_1",
               data=cc_res)
sns.stripplot(x="Species",
                y="CCX_1",
                 data=cc_res)
```



Hình 1 Plot của tương quan chính tắc giữa X và biến ẩn

```
plt.figure(figsize=(10,8))
sns.boxplot(x="Species", y="CCY_1", data=cc_res)
sns.stripplot(x="Species", y="CCY_1", data=cc_res)
```

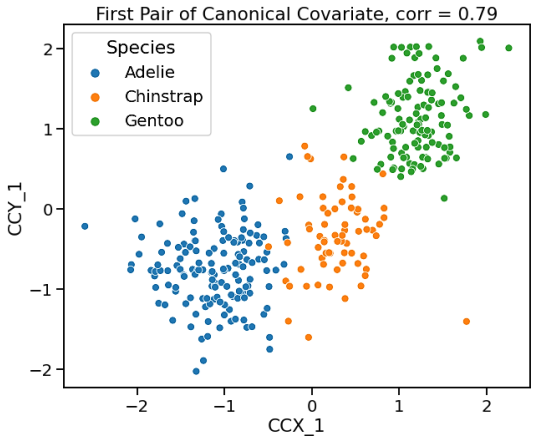


Hình 2 Plot của tương quan chính tắc giữa Y và biến ẩn

Bằng cách tô màu các biểu đồ phân tán giữa cặp hiệp biến chính tắc đầu tiên với biến loài và xem cách các hiệp biến chính tắc nắm bắt biến cơ bản đằng sau tập dữ liệu.

```
plt.figure(figsize=(10,8))
sns.scatterplot(x="CCX_1", y="CCY_1", hue="Species", data=cc_res)
plt.title('First Pair of Canonical Covariate, corr = %.2f' %
          np.corrcoef(X_c[:, 0], Y_c[:, 0])[0, 1])
```

```
Text(0.5, 1.0, 'First Pair of Canonical Covariate, corr = 0.79')
```



Hình 3 Mối tương quan cao giữa cặp biến chính quy đầu tiên

Từ các mối tương quan giữa cặp hiệp biến chính tắc đầu tiên và biến loài trong tập dữ liệu, có thể thấy rằng CCA đã nắm bắt được sự biến đổi chung giữa hai tập dữ liệu. Trong ví dụ này, biến chia sẻ hay là biến tiềm ẩn đằng sau cặp hiệp biến chính tắc đầu tiên là biến loài.

Bằng cách xem xét cẩn thận mối liên hệ giữa CC và một biến ở đây là biến tiềm ẩn, chắc chắn có thể hiểu được kết quả từ kết quả CCA. Thực hiện một bước xa hơn và tạo một bản đồ nhiệt về mối tương quan giữa các hiệp biến chính tắc từ mỗi tập dữ liệu so với tập dữ liệu đầu vào, bao gồm cả dữ liệu meta được liên kết với nó,

Tự lặp lại bằng cách tạo khung dữ liệu với dữ liệu gốc và các hiệp biến chính tắc từ tập dữ liệu đầu tiên X. Để tính toán sự tương quan, chuyển đổi các biến ký tự thành các biến phân loại và chuyển chúng thành mã 0/1/2.

```
ccX_df = pd.DataFrame({"CCX_1":X_c[:, 0],
                       "CCX_2":X_c[:, 1],
                       "Species":df.species.astype('category').cat.codes,
                       "Island":df.island.astype('category').cat.codes,
                       "sex":df.sex.astype('category').cat.codes,
                       "bill_length":X_mc.bill_length_mm,
                       "bill_depth":X_mc.bill_depth_mm})
```

Với hàm corr () của Pandas, có thể tính toán mối tương quan của tất cả các biến trong khung dữ liệu.

```
corr_X_df= ccX_df.corr(method='pearson')
corr_X_df.head()
```

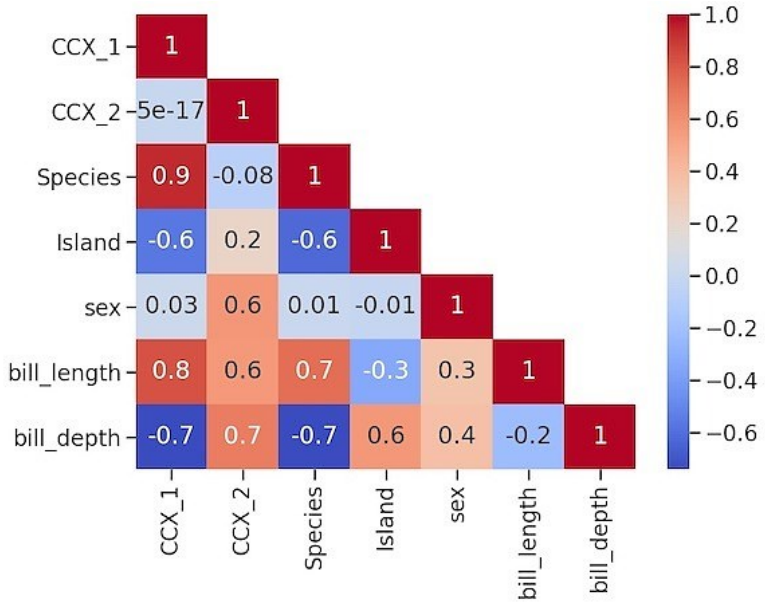
	CCX_1	CCX_2	Species	Island	sex	bill_length	bill_depth
CCX_1	1.000000e+00	7.728804e-18	0.935057	-0.561781	0.025383	0.828437	-0.734650
CCX_2	7.728804e-18	1.000000e+00	-0.078719	0.228933	0.576790	0.560082	0.678447
Species	9.350575e-01	-7.871884e-02	1.000000	-0.622428	0.010964	0.730548	-0.740346
Island	-5.617810e-01	2.289327e-01	-0.622428	1.000000	-0.012435	-0.337179	0.568031
sex	2.538332e-02	5.767897e-01	0.010964	-0.012435	1.000000	0.344078	0.372673

Tạo một heatmap với ma trận tương quan tam giác dưới. Để làm điều đó, thực hiện đặt ma trận tương quan ban đầu với hàm tril ().

```
plt.figure(figsize=(10,8))
X_df_lt = corr_X_df.where(np.tril(np.ones(corr_X_df.shape)).astype(np.bool))
```

Dùng hàm heatmap của Seaborn để tạo ma trận tam giác dưới.

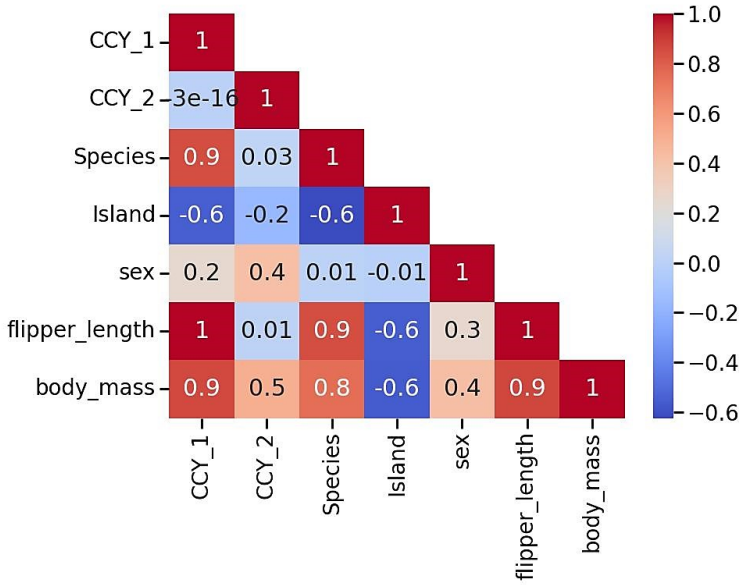
```
sns.heatmap(X_df_lt,cmap="coolwarm",annot=True,fmt='.1g')
plt.tight_layout()
plt.savefig("Heatmap_Canonical_Correlates_from_X_and_data.jpg", format='jpeg',dpi=100)
```



Hình 4 Sơ đồ nhiệt tương quan của Tương quan Chính tắc từ Dữ liệu X và bộ dữ liệu

Làm điều tương tự với tập dữ liệu Y

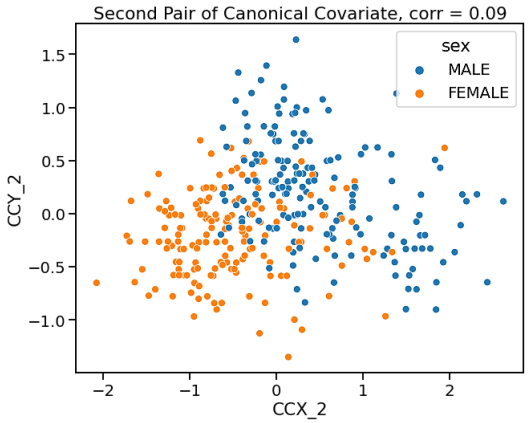
```
ccY_df = pd.DataFrame({"CCY_1":Y_c[:, 0],
                       "CCY_2":Y_c[:, 1],
                       "Species":df.species.astype('category').cat.codes,
                       "Island":df.island.astype('category').cat.codes,
                       "sex":df.sex.astype('category').cat.codes,
                       "flipper_length":Y_mc.flipper_length_mm,
                       "body_mass":Y_mc.body_mass_g})
# compute correlation with Pandas corr()
corr_Y_df= ccY_df.corr(method='pearson')
# Get lower triangular correlation matrix
Y_df_lt = corr_Y_df.where(np.tril(np.ones(corr_Y_df.shape)).astype(np.bool))
# make a lower triangular correlation heatmap with Seaborn
plt.figure(figsize=(10,8))
sns.heatmap(Y_df_lt,cmap="coolwarm",annot=True,fmt='.1g')
plt.tight_layout()
plt.savefig("Heatmap_Canonical_Correlates_from_Y_and_data.jpg", format='jpeg',
dpi=100)
```



Hình 5 Sơ đồ nhiệt tương quan của Tương quan Chính tắc từ Dữ liệu Y và bộ dữ liệu

Hai bản heatmap thể hiện mối tương quan của hiệp biến chính tắc và tập dữ liệu cho thấy giới tính là một biến khác có ảnh hưởng đến cả tập dữ liệu và CCA có thể nắm bắt được điều đó. Để xem ảnh hưởng của giới tính, có thể tạo biểu đồ phân tán với cặp hiệp biến chính tắc thứ hai và tô màu nó theo biến giới tính.

```
plt.figure(figsize=(10,8))
sns.scatterplot(x="CCX_2", y="CCY_2", hue="sex", data=cc_res)
plt.title('Second Pair of Canonical Covariate, corr = %.2f' %
          np.corrcoef(X_c[:, 1], Y_c[:, 1])[0, 1])
```



Hình 6 Mối tương quan cao giữa cặp biến chính quy thứ 2

2. ỨNG DỤNG PHÂN TÍCH TƯƠNG QUAN CHÍNH TẮC

2.1. Giới thiệu Depth Completion:

Depth measurement được sử dụng được sử dụng rộng rãi trong các ứng dụng thị giác máy tính. Tuy nhiên, hầu hết các kỹ thuật hiện có để thu thập độ sâu đều tạo ra depth map với dữ liệu không đầy đủ. Ví dụ, máy ảnh có cấu trúc ánh sáng không thể chụp và có được depth measurement ở những nơi bề mặt quá sáng bóng.

Gần đây, các nhà nghiên cứu đã giới thiệu Depth completion, nhằm mục đích lấp đầy depth measurement còn thiếu bằng cách sử dụng các phương pháp dựa trên học sâu. Những nghiên cứu đó tạo ra depth map dày đặc bằng cách kết hợp các đặc điểm của phép đo sparse depth và hình ảnh RGB tương ứng. Tuy nhiên, họ thường coi việc trích xuất đặc điểm của hai loại thông tin này là các quá trình độc lập, trên thực tế, việc này biến nhiệm vụ mà họ thực hiện thành "multi-modality depth prediction" chứ không phải "depth completion". Mặc dù multi-modality depth prediction có thể tạo ra kết quả đầu ra dày đặc, nhưng chúng không sử dụng được đầy đủ dữ liệu có thể quan sát được. Nhiệm vụ depth completion là duy nhất ở chỗ phần đầu ra của nó đã có thể quan sát được trong đầu vào.

Để hoàn thành nhiệm vụ depth completion, bài viết đề xuất một khuôn khổ dựa trên học sâu, CFCNet. Hình ảnh RGB đầy đủ tương ứng của depth map có thể được phân tách thành hai phần, một phần được gọi là sparse RGB, giữ các giá trị RGB tương ứng tại các vị trí có thể quan sát được trong sparse depth. Phần khác là RGB bổ sung, là phần trừ đi sparse RGB khỏi các hình ảnh RGB đầy đủ. Trong giai đoạn đào tạo, CFCNet tìm hiểu mối quan hệ giữa sparse depth và sparse RGB và sử dụng kiến thức đã học để khôi phục độ sâu không thể quan sát được từ RGB bổ sung.

Để tìm hiểu mối quan hệ giữa hai phương thức, bài viết đề xuất phân tích tương quan chính tắc sâu 2D (2DDCCA). Trong phương pháp được đề xuất, 2DDCCA cố gắng tìm hiểu các phép chiếu phi tuyến tính trong đó các đối tượng được chiếu từ RGB và miền độ sâu có tương quan tối đa. Sử dụng 2DDCCA như một hàm mục tiêu có thể nắm bắt các đặc điểm tương quan về mặt ngữ nghĩa từ miền RGB và miền độ sâu.

Ngoài ra, mục tiêu depth completion còn là xây dựng một khuôn khổ cho sparse depth completion, thúc đẩy mối quan hệ giữa sparse depth và hình ảnh RGB tương ứng của nó, sử dụng thông tin RGB bổ sung để bổ sung cho thông tin sparse depth còn thiếu.

2.2. Động lực:

2.2.1. Khoa học:

Để cải thiện hơn nữa độ chính xác và mạnh mẽ của các thuật toán dự đoán độ sâu, các nhà nghiên cứu đang tìm kiếm các cải tiến bằng cách sử dụng các mẫu sparse depth, vốn đã tồn tại trong nhiều ứng dụng. Tận dụng mối quan hệ của sparse depth và các pixel tương ứng của chúng trong hình ảnh RGB để tối ưu hóa hiệu suất của nhiệm vụ của depth completion. Vì CCA có thể tìm hiểu không gian con được chia sẻ với các đặc điểm dự đoán của nó, bài nghiên cứu ước tính thành phần độ sâu bị thiếu bằng cách sử dụng các đặc điểm từ miền RGB thông qua CCA. Tuy nhiên, CCA truyền thống có vấn đề trong nhiệm vụ thị giác máy tính hiện đại. 2D^{2}CCA được đề xuất thêm để nắm bắt các ngữ nghĩa tương tự từ cả hai bộ mã hóa RGB và chiều sâu. Dựa trên cấu trúc này, decoder trong CFCNet có khả năng sử dụng các đặc điểm độ sâu được tái tạo cùng với các đặc điểm độ sâu quan sát được để khôi phục dense depth map.

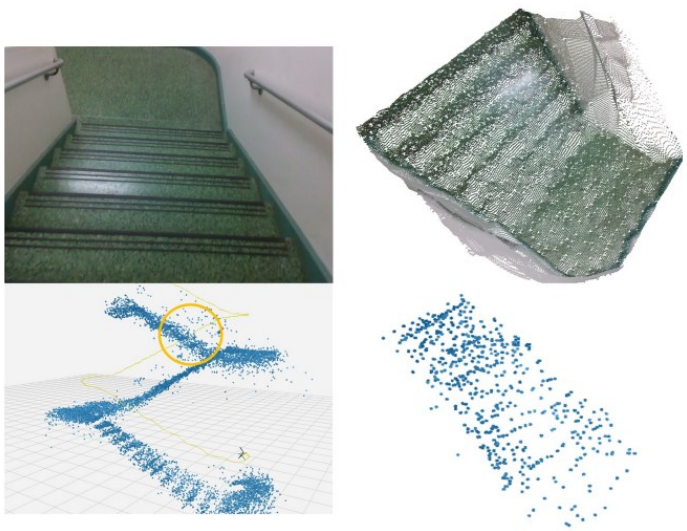
2.2.2. Thực tiễn:

Depth completion nhằm mục đích khôi phục dense depth map từ sparse depth map thu được từ Lidar hoặc các cảm biến độ sâu khác. Ngoài ra, (Xu và cộng sự, 2019) kết hợp mạng dự đoán với một mạng tinh chế khuếch tán cụ thể để điều chỉnh các ràng buộc giữa độ sâu và mức bình thường cho depth completion. (Mal và Karaman, 2018) giới thiệu một mạng hồi quy sâu duy nhất dựa trên depth completion để học trực tiếp từ dữ liệu thô RGB-D, giúp biểu diễn phương thức chéo tốt hơn cho RGB và sparse, đồng thời đạt được độ chính xác tốt nhất cho đến nay. Mục đích của Depth completion là để lấp đầy các pixel bị thiếu trong các hình ảnh có độ sâu tương đối dày đặc hoặc thưa thớt. Bất chấp những tiến bộ gần đây trong dự đoán độ sâu từ hình ảnh RGB đơn lẻ với mạng nơ-ron sâu hơn, các phương pháp hiện có không mang lại kết quả đáng tin cậy cho việc sử dụng thực tế.

2.3. Ứng dụng Depth Completion

Gần đây do sự phổ biến và phát triển đáng kể của học sâu và tài nguyên máy tính, depth completion ngày càng được phổ biến rộng rãi, một số ứng dụng vào thực tiễn hiện tại.

2.3.1. Đo tỷ lệ quán tính trực quan (Visual Inertial Odometry)



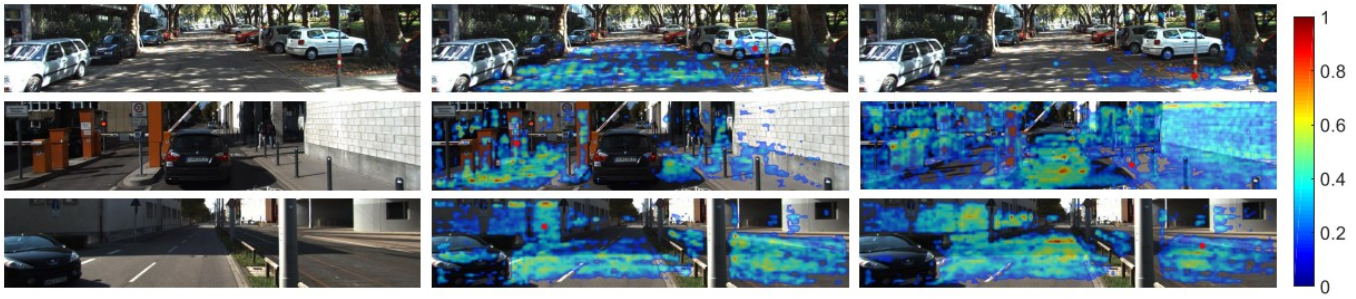
Phương pháp để suy ra dense depth từ chuyển động của máy ảnh và sparse depth được ước tính bằng cách sử dụng hệ thống Visual Inertial Odometry (VIO). Chuỗi hình ảnh là một nguồn thông tin phong phú về cả hình dạng ba chiều (3D) của môi trường và chuyển động của cảm biến bên trong. Chuyển động có thể được suy ra tối đa theo tỷ lệ và hệ quy chiếu Euclid toàn cầu, được cung cấp đủ thị sai và một số vùng Lambertian phân biệt trực quan được cố định trong môi trường và có thể nhìn thấy từ máy ảnh. VIO đóng vai trò như một nền tảng để phát triển các mô hình có thể xử lý chuyển động đầy thử thách và thúc đẩy hơn nữa sự tổng hợp dữ liệu đa chiều và đa giác quan.

2.3.2. Radar-Camera Pixel

(a) (b) (c)

Mặc dù dữ liệu radar và video có thể dễ dàng được hợp nhất ở cấp độ phát hiện, nhưng việc hợp nhất chúng ở cấp độ pixel có khả năng có lợi hơn. Điều này cũng khó khăn hơn một phần do sự thưa thớt của radar, nhưng cũng bởi vì chùm tia radar trên ô tô rộng hơn nhiều so với một pixel thông thường kết hợp với đường cơ sở lớn giữa máy ảnh và radar, dẫn đến sự liên kết kém giữa pixel radar và pixel màu. Sử dụng depth completion được thiết kế cho LiDAR tạo sự hiệu quả bằng hình ảnh đối với radar và video

2.3.3. Hệ thống tự động và tái tạo 3D (Autonomous Systems and 3D Reconstruction)



(a) raw RGB image (b) point A (c) point B

Depth completion là một phần thiết yếu cho các hệ thống tự động và tái tạo 3D. Dense depth rất cần thiết cho người máy, phương tiện tự hành và các nhiệm vụ về thị giác máy tính, bao gồm tránh chướng ngại vật, lập bản đồ và tái tạo 3D.

2.3.4. Self-supervised learning trong Robotics

Depth map là một cách để trực quan hóa khoảng cách của các đối tượng xung quanh cảm biến và rất cần thiết để cho phép rô bốt xây dựng bản đồ chính xác về không gian và lập kế hoạch đường đi của chúng để điều hướng an toàn. Biết liệu một vật có ở gần hay không và chính xác là bao nhiêu, là điều rất quan trọng để tránh va chạm. Đó là điều mà con người và động vật thực hiện một cách tự nhiên. Đối với robot, quá trình đào tạo AI được giám sát đạt được kết quả tương tự bằng cách hiển thị hàng triệu hình ảnh, tất cả đều được gắn nhãn chính xác với 'sự thật mặt đất' để AI học cách nhận ra các vật thể ở các khoảng cách khác nhau. Với việc kết hợp AI với depth map sẽ cải thiện đáng kể chất lượng của máy ảnh độ sâu, đặc biệt khi đo khoảng cách bên ngoài đường được xác định chặt chẽ của nó.

2.4. Nghiên cứu liên quan

2.4.1. Deep Canonical Correlation Analysis (DCCA)

Canonical correlation analysis - Phân tích tương quan chính tắc (CCA) (Hotelling, 1936; Anderson, 1984) là một kỹ thuật thống kê tiêu chuẩn để tìm các phép chiếu tuyến tính của hai vectơ ngẫu nhiên có tương quan tối đa. Kernel Canonical correlation analysis - Phân tích tương quan chính tắc của hạt nhân (KCCA) (Akaho, 2001; Melzer và cộng sự, 2001; Bach & Jordan, 2002; Hardoon và cộng sự, 2004) là một phần mở rộng của CCA trong đó các phép chiếu phi tuyến tương quan tối đa, được giới hạn để tái tạo các không gian Hilbert trong hạt nhân với các hạt nhân tương ứng. Cả CCA và KCCA đều là các kỹ thuật để học các biểu diễn của hai chế độ xem (view) dữ liệu, sao cho biểu diễn của mỗi chế độ xem đồng thời là dự đoán chính xác nhất và dễ đoán nhất đối với chế độ xem khác.

CCA và KCCA đã được sử dụng để phân tích dữ liệu không giám sát khi có nhiều chế độ xem khác (Hardoon và cộng sự, 2007; Vinokourov và cộng sự, 2003; Dhillon và cộng sự, 2011); học các đặc điểm cho nhiều phương thức sau đó

được hợp nhất để dự đoán (Sargin và cộng sự, 2007); đặc điểm học tập cho một chế độ xem duy nhất khi một chế độ xem khác có sẵn để học biểu diễn nhưng không có tại thời điểm dự đoán (Blaschko & Lampert, 2008; Chaudhuri et al., 2009; Arora & Livescu, 2012); và giảm độ phức tạp mẫu của các vấn đề dự đoán bằng cách sử dụng dữ liệu không được gắn nhãn (Kakade & Foster, 2007). Các ứng dụng rộng rãi trên một số lĩnh vực, bao gồm y học, khí tượng học (Anderson, 1984), hóa học (Montanarella và cộng sự, 1995), sinh học và thần kinh học (Vert & Kanehisa, 2002; Hardoon và cộng sự, 2007), ngôn ngữ tự nhiên xử lý (Vinokourov và cộng sự, 2003; Haghighi và cộng sự, 2008; Dhillon và cộng sự, 2011), xử lý giọng nói (Choukri & Chollet, 1986; Rudzicz, 2010; Arora & Livescu, 2013), thị giác máy tính (Kim và cộng sự ., 2007), và xử lý tín hiệu đa phương thức (Sargin và cộng sự, 2007; Slaney & Covell, 2000). Một thuộc tính hấp dẫn của CCA đối với các nhiệm vụ dự đoán là, nếu có nhiễu trong một trong hai chế độ xem không tương quan với chế độ xem kia, thì các biểu diễn đã học không được chứa nhiễu trong các kích thước không tương quan.

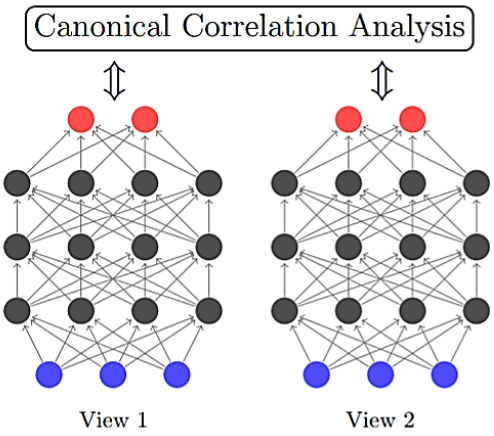
Trong khi hạt nhân CCA cho phép học các biểu diễn phi tuyến, nó có nhược điểm là biểu diễn bị giới hạn bởi hạt nhân cố định. Ngoài ra, vì nó là một phương pháp phi tham số, thời gian cần thiết để đào tạo KCCA hoặc tính toán các đại diện của các điểm dữ liệu mới sẽ có tỷ lệ thấp hơn với kích thước của tập hợp đào tạo. Thực hiện xem xét việc học các biểu diễn phi tuyến linh hoạt thông qua mạng sâu. Mạng sâu không mắc phải những nhược điểm đã nói ở trên của mô hình phi tham số và dựa trên sự thành công trên thực nghiệm của các mô hình sâu trên nhiều nhiệm vụ, có thể mong đợi có thể học được nhiều cách biểu diễn có tương quan cao hơn. Mạng sâu đã được sử dụng rộng rãi để học các biểu diễn, ví dụ sử dụng máy Boltzmann sâu (deep Boltzmann machines) (Salakhutdinov & Hinton, 2009), máy mã tự động sâu (deep autoencoders) (Hinton & Salakhutdinov, 2006) và mạng truyền thẳng phi tuyến sâu (deep nonlinear feedforward networks) (Hinton và cộng sự, 2006). Chúng đã rất thành công để học các biểu diễn của một chế độ xem dữ liệu duy nhất. Deep CCA (DCCA) học đồng thời hai ánh xạ phi tuyến sâu của hai chế độ xem có tương quan tối đa. Điều này có thể được coi là học một hạt nhân cho KCCA, nhưng chức năng ánh xạ không bị giới hạn trong không gian Hilbert hạt nhân tái tạo.

Công trình liên quan chặt chẽ nhất là của Ngiam et al. trên máy mã tự động đa phương thức (multimodal autoencoders) (Ngiam và cộng sự, 2011) và của Srivastava và Salakhutdinov trên máy Boltzmann hạn chế đa phương thức (multimodal restricted Boltzmann machines) (Srivastava & Salakhutdinov, 2012). Trong những cách tiếp cận này, có một mạng duy nhất đang được học với một hoặc nhiều lớp được kết nối với cả hai chế độ xem (phương thức); trong trường hợp không có một trong các chế độ xem, nó có thể được dự đoán từ chế độ xem khác bằng cách sử dụng mạng đã học. Sự khác biệt chính là trong cách tiếp cận của chúng tôi, chúng tôi học hai mã hóa sâu riêng biệt, với mục tiêu là các mã hóa đã học càng tương quan càng tốt. Các mục tiêu khác nhau này có thể có lợi thế trong các cài đặt khác nhau. Trong công việc hiện tại, chúng tôi đặc biệt quan tâm đến mục tiêu tương quan, đó là mở rộng CCA với các ánh xạ phi tuyến đã học. Do đó, cách tiếp cận của chúng tôi có thể áp dụng trực tiếp trong tất cả các cài đặt nơi CCA và KCCA được sử dụng và chúng tôi so sánh khả năng của nó so với CCA và KCCA để tổng quát hóa mục tiêu tương quan với dữ liệu mới, cho thấy rằng DCCA đạt được kết quả tốt hơn nhiều.

Để khắc phục những hạn chế của CCA truyền thống trong đó các dự báo phải là tuyến tính, phân tích tương quan kinh điển sâu (DCCA) đã được đề xuất. DCCA sử dụng mạng nơ ron sâu để tìm hiểu các phép chiếu phi tuyến tính phức tạp hơn giữa nhiều phương thức. CCA, DCCA và các biến thể khác đã được sử dụng rộng rãi trong các bài toán học biểu diễn đa phương thức

DCCA tính toán các biểu diễn của hai khung nhìn bằng cách chuyển chúng qua nhiều lớp biến đổi phi tuyến xếp chồng lên nhau (xem Hình dưới). Giả sử đơn giản rằng mỗi lớp trung gian trong mạng cho chế độ xem đầu tiên có $\boldsymbol{c_1}$ đơn vị và lớp cuối cùng (đầu ra) có o đơn vị. Gọi $\boldsymbol{x_1} \in \boldsymbol{R^{n_1}}$ là một thể hiện của khung nhìn thứ nhất. Kết quả đầu ra của lớp đầu tiên cho ví dụ $\boldsymbol{x_1}$ là $\boldsymbol{h_1} = \boldsymbol{s}\left(\boldsymbol{W_1^1 x_1} + \boldsymbol{b_1^1}\right) \in \boldsymbol{R^{c_1}}$, trong đó $\boldsymbol{W_1^1} \in \boldsymbol{R^{c_1 \times n_1}}$ là ma trận có trọng số, $\boldsymbol{b_1^1} \in \boldsymbol{R_1^c}$ là vectơ của các thành phần, và $\boldsymbol{s}$: $\boldsymbol{R} \rightarrow \boldsymbol{R}$ là một hàm phi tuyến được áp dụng theo từng thành phần. Các kết quả đầu ra h_1 sau đó có thể được sử dụng để tính toán các kết quả đầu ra của lớp tiếp theo là$\boldsymbol{h_2} = \boldsymbol{s}\left(\boldsymbol{W_2^1 h_1} + \boldsymbol{b_2^1}\right) \in \boldsymbol{R^{c_1}}$, và cứ tiếp tục như vậy cho đến khi biểu diễn cuối cùng $\boldsymbol{f_1(x_1)} = \boldsymbol{s}\left(\boldsymbol{W_d^1 h_{d-1}} + \boldsymbol{b_d^1}\right) \in \boldsymbol{R^o}$ được tính cho một mạng có $\boldsymbol{d}$ lớp. Cho một thể hiện $\boldsymbol{x_2}$ của khung nhìn thứ hai, biểu diễn $\boldsymbol{f_2(x_2)}$ được tính theo cùng một cách, với các tham số khác nhau $\boldsymbol{W_l^2}$ và $\boldsymbol{b_l^2}$ (và các tham số kiến trúc có khả năng khác nhau $\boldsymbol{c_2}$ và d). Mục đích là để cùng tìm hiểu các tham số cho cả hai khung nhìn $\boldsymbol{W_l^v}$ và $\boldsymbol{b_l^v}$ sao cho $\boldsymbol{corr}\left(\boldsymbol{f_1(X_1)}, \boldsymbol{f_2(X_2)}\right)$ càng cao càng tốt. Nếu θ_1 là vectơ của tất cả các tham số $\boldsymbol{W_l^2}$ và $\boldsymbol{b_l^2}$ của hình chiếu thứ nhất đối với $\boldsymbol{l} = \boldsymbol{1}, \ldots, \boldsymbol{d}$, và tương tự với $\boldsymbol{\theta_2}$, thì

$$(\boldsymbol{\theta_1^*}, \boldsymbol{\theta_2^*}) = \boldsymbol{argmax}_{(\boldsymbol{\theta_1}, \boldsymbol{\theta_2})}\, \boldsymbol{corr}\left(\boldsymbol{f_1}(\boldsymbol{X_1};\, \boldsymbol{\theta_1}), \boldsymbol{f_2}\,(\boldsymbol{X_2};\, \boldsymbol{\theta_2})\right) \quad \boldsymbol{(2.4(1))}$$



Giản đồ của DCCA, bao gồm hai mạng sâu để các lớp đầu ra (lớp trên cùng) có tương quan tối đa. Các nút màu xanh lam tương ứng với các đặc điểm đầu vào ($n_1 = n_2 = 3$), các nút xám là các nút ẩn ($c_1 = c_2 = 4$), và lớp đầu ra là nút đỏ ($o = 2$). Cả hai mạng đều có $d = 4$ lớp

Để tìm $(\boldsymbol{\theta_1^*}, \boldsymbol{\theta_2^*})$, chúng tôi theo dõi gradient của mục tiêu tương quan như được ước tính trên dữ liệu huấn luyện. Gọi $\boldsymbol{H_1} \in \boldsymbol{R^{o \times m}}, \boldsymbol{H_2} \in \boldsymbol{R^{o \times m}}$ là các ma trận có cột là biểu diễn cấp cao nhất được tạo ra bởi các mô hình sâu trên hai khung nhìn, đối với tập huấn luyện có kích thước m. Gọi $\overline{\boldsymbol{H_1}} = \boldsymbol{H_1} - \frac{1}{\boldsymbol{m}} \boldsymbol{H_1}$ là ma trận dữ liệu ở tâm (tương ứng $\overline{H_2}$), và xác định $\widehat{\boldsymbol{\Sigma_{12}}} = \frac{1}{\boldsymbol{m-1}} \overline{\boldsymbol{H_1}}\overline{\boldsymbol{H_2'}}$ và $\widehat{\boldsymbol{\Sigma_{11}}} = \frac{1}{\boldsymbol{m-1}} \overline{\boldsymbol{H_1}}\overline{\boldsymbol{H_1'}} + \boldsymbol{r_1 I}$ cho hằng số chính quy $\boldsymbol{r_1}$ (tương ứng $\widehat{\boldsymbol{\Sigma_{22}}}$). Giả sử $\boldsymbol{r1} > \boldsymbol{0}$, để $\widehat{\boldsymbol{\Sigma_{11}}}$ là xác định dương.

Tổng tương quan của k thành phần trên cùng của H_1 và H_2 là tổng của k giá trị đơn nhất trên cùng của ma trận $\boldsymbol{T} = \widehat{\boldsymbol{\Sigma_{11}}}^{-\frac{1}{2}} \widehat{\boldsymbol{\Sigma_{12}}} \widehat{\boldsymbol{\Sigma_{22}}}^{-\frac{1}{2}}$. Nếu chúng ta lấy $\boldsymbol{k} = \boldsymbol{o}$, thì đây chính xác là chuẩn vết ma trận của $\boldsymbol{T}$, hoặc

$$\boldsymbol{corr}\,(\boldsymbol{H_1}, \boldsymbol{H_2}) = \|\boldsymbol{T}\|_{\boldsymbol{tr}} = \boldsymbol{tr}(\boldsymbol{T'T})^{\frac{1}{2}}. \quad (\mathbf{2.4(2)})$$

Các tham số $\boldsymbol{W_l^v}$ và V_k của DCCA được huấn luyện để tối ưu hóa đại lượng này bằng cách sử dụng tối ưu hóa dựa trên gradient. Để tính toán gradient của $\boldsymbol{corr}(\boldsymbol{H_1}, \boldsymbol{H_2})$ đối với tất cả các tham số $\boldsymbol{W_l^v}$ và $\boldsymbol{b_l^v}$, chúng ta có thể tính toán gradient của nó đối với $\boldsymbol{H_1}$ và $\boldsymbol{H_2}$ và sau đó sử dụng backpropagation. Nếu phân rã giá trị đơn nhất của T là $\boldsymbol{T} = \boldsymbol{UDV'}$, thì

$$\frac{\boldsymbol{\partial corr}(\boldsymbol{H_1}, \boldsymbol{H_2})}{\boldsymbol{\partial H_1}} = \frac{\mathbf{1}}{\boldsymbol{m-1}} (\mathbf{2}\boldsymbol{\nabla_{11}}\overline{\boldsymbol{H_1}} + \boldsymbol{\nabla_{12}}\overline{\boldsymbol{H_2}}) \; (\mathbf{2.4(3)})$$

với $\boldsymbol{\nabla_{12}} = \widehat{\boldsymbol{\Sigma}}_{\mathbf{11}}^{-\frac{1}{2}} \boldsymbol{UV'} \widehat{\boldsymbol{\Sigma}}_{\mathbf{22}}^{-\frac{1}{2}} (\mathbf{2.4(4)})$ và $\boldsymbol{\nabla_{11}} = -\frac{1}{2} \widehat{\boldsymbol{\Sigma}}_{\mathbf{11}}^{-\frac{1}{2}} \boldsymbol{UDU'} \widehat{\boldsymbol{\Sigma}}_{\mathbf{11}}^{-\frac{1}{2}} \; (\mathbf{2.4(5)})$

Đạo hàm của gradient không hoàn toàn đơn giản (ví dụ, liên quan đến gradient của dấu vết của căn bậc hai ma trận, mà chúng tôi không thể tìm thấy trong các tài liệu tham khảo tiêu chuẩn như (Petersen & Pedersen, 2012)). Ta cũng chính thức hóa (10) bằng cách thêm vào nó một trọng số phạt bậc hai (penalty weight) với trọng số $\lambda_b > 0$ cho tất cả các tham số.

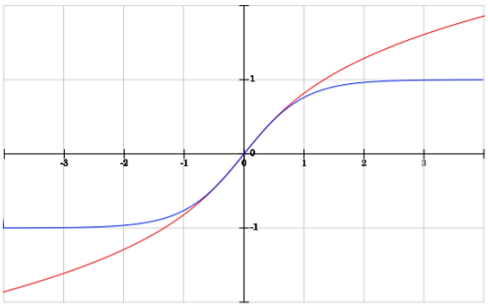
Vì mục tiêu tương quan là một hàm của toàn bộ tập huấn luyện không phân tách thành tổng trên các điểm dữ liệu, nên không rõ cách sử dụng quy trình tối ưu ngẫu nhiên hoạt động trên các điểm dữ liệu tại một thời điểm. Chúng tôi đã thử nghiệm với phương pháp ngẫu nhiên dựa trên các lô nhỏ, nhưng đã thu được kết quả tốt hơn nhiều với tối ưu hóa toàn bộ bằng cách sử dụng phương pháp tối ưu hóa bậc hai L-BFGS (Nocedal & Wright, 2006) đã được phát hiện là hữu ích cho việc học sâu trong các bối cảnh khác (Le và cộng sự, 2011).

Nói chung, sẽ không đạt được kết quả tốt nhất nếu tối ưu hóa tham số được bắt đầu từ khởi tạo ngẫu nhiên — một số hình thức đào tạo trước là cần thiết. Trong các thử nghiệm, ta khởi tạo các tham số của mỗi lớp bằng một bộ mã tự động làm giảm sự cố (Vincent và cộng sự, 2008). Dữ liệu đào tạo đầu vào được đặt ở trung tâm được tập hợp thành ma trận $\boldsymbol{X} \in \boldsymbol{R^{n \times m}}$, một ma trận méo (distorted) $\tilde{X}$ được tạo bằng cách thêm nhiễu Gauss trung bình bằng không (zero-mean) với phương sai $\boldsymbol{\sigma_a^2}$. Đối với các tham số $\boldsymbol{W} \in \boldsymbol{R^{c \times n}}$ và $\boldsymbol{b} \in \boldsymbol{R^c}$, dữ liệu tái tạo $\widehat{\boldsymbol{X}} = \boldsymbol{W's} \left(\boldsymbol{W\tilde{X}} + \boldsymbol{b\bar{1}'}\right)$ được hình thành. Sau đó, sử dụng L-BFGS để tìm mức tối thiểu cục bộ của tổng sai số bình phương từ việc tái tạo lại dữ liệu ban đầu, cộng với hình phạt bậc hai: $\boldsymbol{l_a}(\boldsymbol{W}, \boldsymbol{b}) = \left|\left|\boldsymbol{X} - \boldsymbol{X}\right|\right|_{\boldsymbol{F}}^{\mathbf{2}} + \boldsymbol{\lambda_a} \left(\left|\left|\boldsymbol{W}\right|\right|_{\boldsymbol{F}}^{\mathbf{2}} + \left|\left|\boldsymbol{b}\right|\right|_{\mathbf{2}}^{\mathbf{2}}\right), (\mathbf{2.4(6)})$

trong đó $||\cdot||_F$ là ma trận Frobenius norm. Các giá trị tối thiểu W^* và b^* được sử dụng để khởi tạo tối ưu hóa mục tiêu DCCA và để tạo ra biểu diễn cho việc đào tạo trước lớp tiếp theo. σ_a^2 và λ_a được coi là siêu tham số (hyperparameter) và được tối ưu hóa trên tập phát triển.

Non-saturating nonlinearity - Phi tuyến tính không bão hòa

Bất kỳ dạng phi tuyến tính sigmoid nào cũng có thể được sử dụng để xác định đầu ra của các nút trong mạng DCCA, nhưng trong các thử nghiệm, ta đã thu được kết quả tốt nhất bằng cách sử dụng một hàm sigmoid không bão hòa mới dựa trên căn bậc ba. Nếu $g: R \to R$ là hàm $g(y) = \frac{y^3}{3} + y$ thì hàm cần là $s(x) = g^{-1}(x)$. Giống như các phi tuyến tính logistic (σ) và tanh phổ biến hơn, s có hình dạng $sigmoid$ và có độ dốc đơn vị tại $x = 0$. Giống như tanh, nó là một hàm lẻ. Tuy nhiên, logistic và tanh tiếp cận giá trị tiệm cận của chúng rất nhanh, tại thời điểm đó đạo hàm giảm về cơ bản bằng 0 (tức là chúng bão hòa). Mặt khác, s không bị giới hạn và đạo hàm của nó giảm dần theo x. Chúng tôi giả thuyết rằng các thuộc tính này phù hợp hơn cho việc tối ưu hóa hàng loạt bằng các phương pháp bậc hai, nếu không thì có thể bị mắc kẹt sớm trong quá trình tối ưu hóa. Hình dưới vẽ hàm s cùng với tanh.



1 So sánh hàm sigmoid bậc 3 (đỏ) với hàm tanh chuẩn (xanh)

Một tính chất khác mà hàm sigmoid không bão hoà có cùng với logistic và tanh là đạo hàm của nó là một hàm đơn giản về giá trị của nó. Ví dụ: $s'(x) = \sigma(x)(1 - \sigma(x))$ và $tanh'(x) = 1 - tanh^2(x)$.

Thuộc tính này thuận tiện trong việc triển khai, vì nó có nghĩa là đầu vào của một đơn vị có thể bị ghi đè bởi đầu ra của nó. Ngoài ra, hóa ra, sẽ hiệu quả hơn nếu tính các đạo hàm dưới dạng một hàm của giá trị trong tất cả các trường hợp này (ví dụ: cho trước $y = tanh(x)$, $1 - y^2$ có thể được tính hiệu quả hơn $(1 - tanh^2(x))$.Trong trường hợp của s, ta có $s'(x) = (s2(x) + 1)^{-1}$ như được dễ dàng chỉ ra với sự phân biệt ngầm định. Nếu $y = s(x)$, thì $x = \frac{y^3}{3} + y, \frac{dx}{dy} = y^2 + 1$ và $\frac{dy}{dx} = \frac{1}{y^2+1}$.

Để tính $s(x)$, ta dùng phương pháp Newton. Để giải $g(x) - x = 0$,

$$y_{n+1} = y_n - \frac{g(y_n) - x}{g'(y_n)} = y_n - \frac{\frac{y_n^3}{3} + y_n - x}{y_n^2 + 1} = \frac{\frac{2y_n^3}{3} + x}{y_n^2 + 1}$$

Đối với x dương, khởi tạo $y_0 = x$, quá trình lặp giảm đơn điệu, do đó sự hội tụ được đảm bảo. Trong phạm vi giá trị trong các thử nghiệm, nó hội tụ với độ chính xác của máy móc chỉ trong một vài lần lặp lại. Khi x âm, chúng ta sử dụng thuộc tính s là số lẻ, vì vậy $s(x) = -s(-x)$.

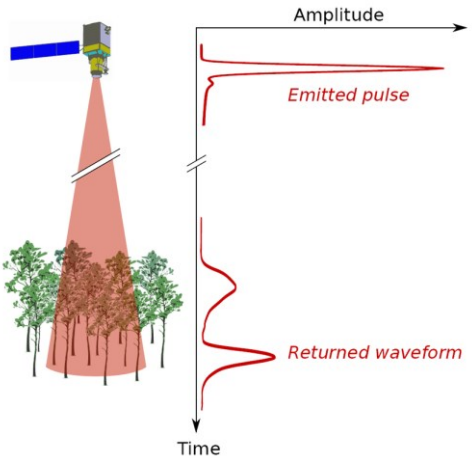
2.4.2. Sparse Depth Completion

Sparse Depth Completion là một nhiệm vụ của để depth completion từ sparse depth và hình ảnh RGB tương ứng. Sparse depth được tạo ra bởi phương pháp âm thanh nổi chứa nhiều thông tin hơn về đường viền đối tượng và ít thông tin hơn về các vùng không có kết cấu. Cảm biến LiDAR tạo ra sự thưa thớt có cấu trúc do hành vi quét. Hệ thống SLAM dựa trên đặc điểm (chẳng hạn như ORB SLAM) chỉ nắm bắt thông tin độ sâu tại vị trí của các điểm đặc trưng tương ứng.

Đối với độ thưa thớt đầu vào, xếp chồng các depth map thưa thớt và hình ảnh RGB tương ứng lại với nhau để xây dựng đầu vào bốn kênh (RGB-D) trước khi đưa vào mạng ước tính độ sâu dựa trên ResNet. Phương pháp xử lý này tạo ra kết quả tốt hơn so với ước tính độ sâu một mắt chỉ với hình ảnh RGB. Các nghiên cứu khác về encoder-decoder dựa trên các nghiên cứu hai nhánh tương tự như các khung được sử dụng trong các tác vụ phân đoạn RGB-D. Phương pháp tiếp cận này không áp dụng các biện pháp xử lý đặc biệt đối với nhánh sâu thưa. Trong hầu hết các khuôn khổ hai nhánh, các đặc điểm từ các nguồn khác nhau được trích xuất độc lập và hợp nhất thông qua các phép nối hoặc bổ

sung trực tiếp hoặc sử dụng các đặc điểm từ nhánh RGB để cung cấp thêm hướng dẫn để tinh chỉnh kết quả dự đoán độ sâu.

2.4.3. LiDAR



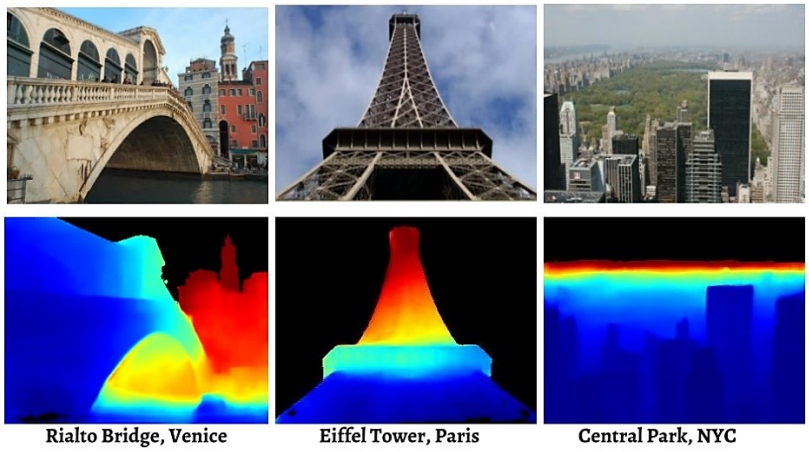
Trong những năm gần đây, LiDAR đã được triển khai trong vô số ứng dụng, thiết kế của công cụ này đã được cải thiện đáng kể, dẫn đến một thiết kế với các yêu cầu về chi phí, kích thước, trọng lượng và công suất thấp đáng kể. Vai trò của LiDAR trong các nền tảng di động và trên không đã tăng lên, phát triển từ một kỹ thuật đo lường hữu ích phù hợp cho các nghiên cứu về aerosol khí quyển để tạo điều kiện thuận lợi cho việc lập bản đồ và tránh những chướng ngại vật vốn được cho là đầy thách thức. Công dụng của LiDAR còn là ứng dụng quét tia hồng ngoại từ các không gian xung quanh, cùng với đó là đo lường thời gian phản xạ lại của các tia sáng. Tập hợp kết quả các tia sáng và cảm biến trong máy quét sẽ nhận diện được độ nông sâu của môi trường xung quanh nó. Một áp dụng phổ biến gần đây là Apple đưa máy quét LiDAR lần đầu lên smartphone là iPhone 12 Pro, iPhone 12 Pro Max và gần đây nhất là iphone 13 Pro, và iphone 13 Pro Max.

Tác dụng của cảm biến LiDAR, (1) đo khoảng cách với độ chính xác cao, (2) tái tạo hình ảnh có độ phân giải cao và (3) tái tạo được hình ảnh 3D động. Việc phân loại các công cụ LiDAR có thể rất rộng và chủ quan, tùy thuộc vào bối cảnh áp dụng. Tuy nhiên, công cụ này thường được phân loại bằng cách sử dụng ba loại chức năng thu thập thông tin mà nó cung cấp là không gian, quang phổ và thời gian.

LiDAR, với độ chính xác cao và phạm vi cảm biến xa, đã được tích hợp vào một số lượng lớn robot và phương tiện tự động. Depth completion đã được nghiên cứu kỹ lưỡng kể từ khi xuất hiện các cảm biến độ sâu hoạt động. Các phương pháp hiện tại chủ yếu nhằm mục đích xử lý các phép đo độ sâu chưa hoàn chỉnh từ hai loại cảm biến, máy quét ánh sáng có cấu trúc và LiDAR. Gần đây, một số phương pháp đã đạt được depth completion dựa trên các phép đo sparse depth. Uhrig và cộng sự, đề xuất một lớp tích chập bất biến thưa thớt để nâng cao các phép đo độ sâu từ LiDAR, ngoài ra, kết hợp sparse depth và hình ảnh màu sắc làm đầu vào của một mạng không có sẵn và tiếp tục khám phá tính khả thi của việc hoàn thành LiDAR tự giám sát.

Depth completion từ các phép đo LiDAR là một thách thức vì một số lý do. Thứ nhất, các phép đo LiDAR rất thưa thớt và cũng có khoảng cách không đều trong không gian hình ảnh. Thứ hai, đó là một nhiệm vụ không hề nhỏ là cải thiện độ chính xác của dự đoán bằng cách sử dụng hình ảnh màu tương ứng, nếu có, vì độ sâu và màu sắc là các phương thức cảm biến khác nhau. Thứ ba, độ sâu sự thật mặt đất dày đặc thường không có sẵn và việc thu thập các chú thích ở cấp độ pixel có thể tốn nhiều công sức và không thể mở rộng.

2.4.4. Depth prediction

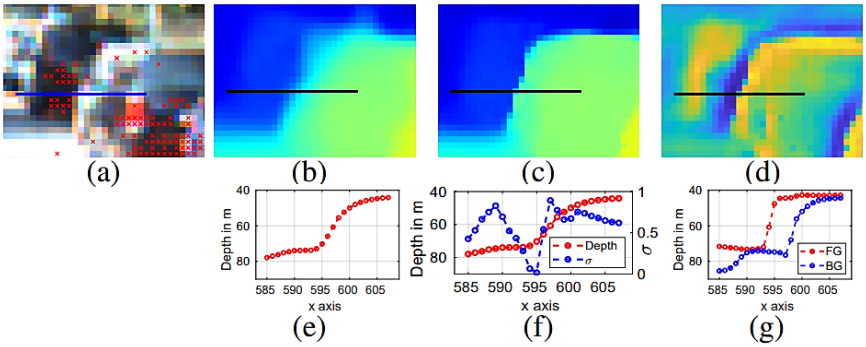


Dự đoán chiều sâu (Depth prediction) dựa trên RGB, một lĩnh vực nghiên cứu tích cực do nhu cầu cảm biến độ sâu trong nhiều ứng dụng chỉ có sẵn ở máy ảnh RGB. Một số nhà nghiên cứu ước tính độ sâu từ hình ảnh RGB bằng cách sử dụng các đặc điểm được làm thủ công, chắc chắn sẽ tạo ra nhiễu.

Với tiến bộ và phát triển đáng kể của học sâu, mạng nơ-ron đã đạt được nhiều thành công trong depth prediction. (Eigen, Puhrsch và Fergus 2014) điều chỉnh CNN hai tỷ lệ để ước tính độ sâu từ một hình ảnh RGB duy nhất với chức năng mất lỗi bất biến tỷ lệ để cải thiện cả độ chính xác toàn cục và cục bộ. Laina và cộng sự. (Laina và cộng sự 2016) đề xuất một mạng phần dư tích tụ đầy đủ để dự đoán độ sâu, kết quả này tốt hơn kết quả trong (Eigen, Puhrsch và Fergus 2014). Mousavian và cộng sự. (Mousavian, Pirsiavash, và Koseck 2016) cùng thực hiện phân đoạn ngữ nghĩa và ước lượng độ sâu với mạng tích hợp sâu, giúp giảm đáng kể lỗi dự đoán. Xu và cộng sự. (Xu và cộng sự 2017) đề xuất một mô hình sâu kết hợp thông tin bổ sung thu được từ nhiều đầu ra CNN để cải thiện hơn nữa kết quả.

Mặc dù độ chính xác được cải thiện nhiều với những nỗ lực này, nhưng depth map dự đoán chỉ từ hình ảnh RGB vẫn không đủ chính xác để sử dụng thực tế.

2.4.5. Depth Representations



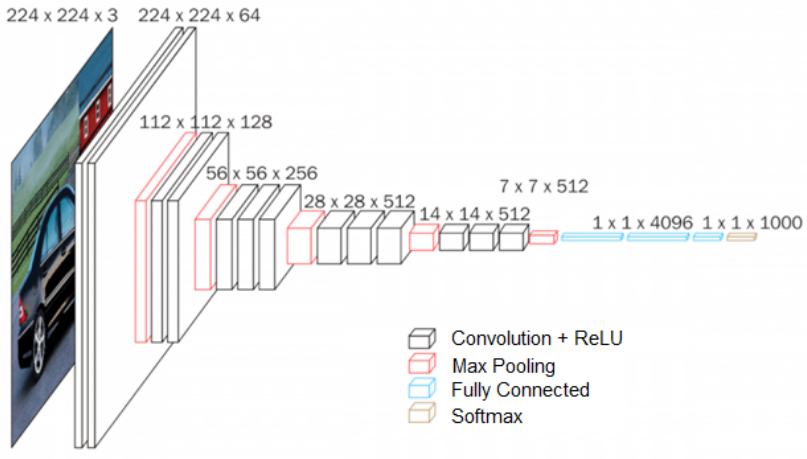
Hình 7 Chiều sâu bôi quanh ranh giới.

Hiển thị độ sâu chân lý mặt đất (màu đỏ) phủ lên một hình ảnh (a), độ sâu ước tính bằng phương pháp SoTA (b), độ sâu hợp nhất (c), trọng lượng ước tính σ (d), độ sâu một slice (e), độ sâu hợp nhất và σ slice (f), và slice tiền cảnh và nền (g). Khả năng ngoại suy trong (g) dẫn đến ranh giới độ sâu sắc nét trong (f), thay vì độ sâu bị làm nhiễu trong (e).

Depth map đã được sử dụng để hợp nhất RGBD và phân đoạn cá thể, chúng mã hóa các tia quan sát cảm biến và vùng kề giữa các điểm. Depth map là các đại diện nhỏ gọn và các lưới thông thường có thể được xử lý với CNN theo cách tương tự để hình ảnh siêu phân giải. Đây là sự lựa chọn đại diện cho các kỹ thuật chỉnh màu và hợp nhất cho depth completion.

Layered Depth Images (LDI) đại diện trung gian cho kết xuất dựa trên hình ảnh hiệu quả, được thu thập bằng cách tích lũy các giá trị độ sâu thông qua đệm z từ nhiều hình ảnh độ sâu của các điểm xem gần đó. Tulsiani và cộng sự, suy ra biểu diễn độ sâu 2 lớp (phục hồi độ sâu của cảnh có thể nhìn thấy và không nhìn thấy) từ một hình ảnh đầu vào duy nhất bằng cách học tổng hợp chế độ xem từ đặc điểm giám sát có hướng dẫn của camera nhiều chế độ. Hedman và cộng sự, đề xuất thuật toán tái tạo ảnh 3D xây dựng biểu diễn hình học nhiều lớp của cảnh bằng cách làm cong một số depth map và ghép ảnh toàn cảnh màu và độ sâu cho bề mặt cảnh trước và sau. Trong các trường hợp này, multi layered representation được xây dựng, từ các điểm nhìn/depth map đa lớp của cảnh.

2.4.6. VGG-16



Hình 8 Kiến trúc VGG-16

VGG-16 đã được chứng minh là một cột mốc quan trọng trong nhiệm vụ của nhân loại để làm cho máy tính "nhìn thấy" thế giới. Rất nhiều nỗ lực đã được thực hiện để cải thiện khả năng này theo kỷ luật Thị giác Máy tính (CV) trong một số thập kỷ. VGG16 là một trong những đổi mới quan trọng mở đường cho một số đổi mới tiếp theo trong lĩnh vực này. Karen Simonyan và Andrew Zisserman đã đề xuất ý tưởng về mạng VGG vào năm 2013 và gửi mô hình thực tế dựa trên ý tưởng này trong ImageNet Challenge 2014.

Thứ nhất, so với các trường tiếp nhận lớn trong lớp chập đầu tiên, mô hình này đề xuất việc sử dụng trường tiếp nhận 3 x 3 rất nhỏ (bộ lọc) trong toàn bộ mạng với bước sóng là 1 pixel. Hai bộ lọc 3 x 3 liên tiếp cung cấp trường tiếp nhận hiệu quả là 5 x 5. Tương tự, ba bộ lọc 3 x 3 tạo nên trường tiếp nhận là 7 x 7. Bằng cách này, sự kết hợp của nhiều bộ

lọc 3 x 3 có thể hỗ trợ một receptive area có quy mô lớn hơn. Thứ hai, VGG16 cũng làm giảm số lượng thông số trọng lượng trong mô hình một cách đáng kể.

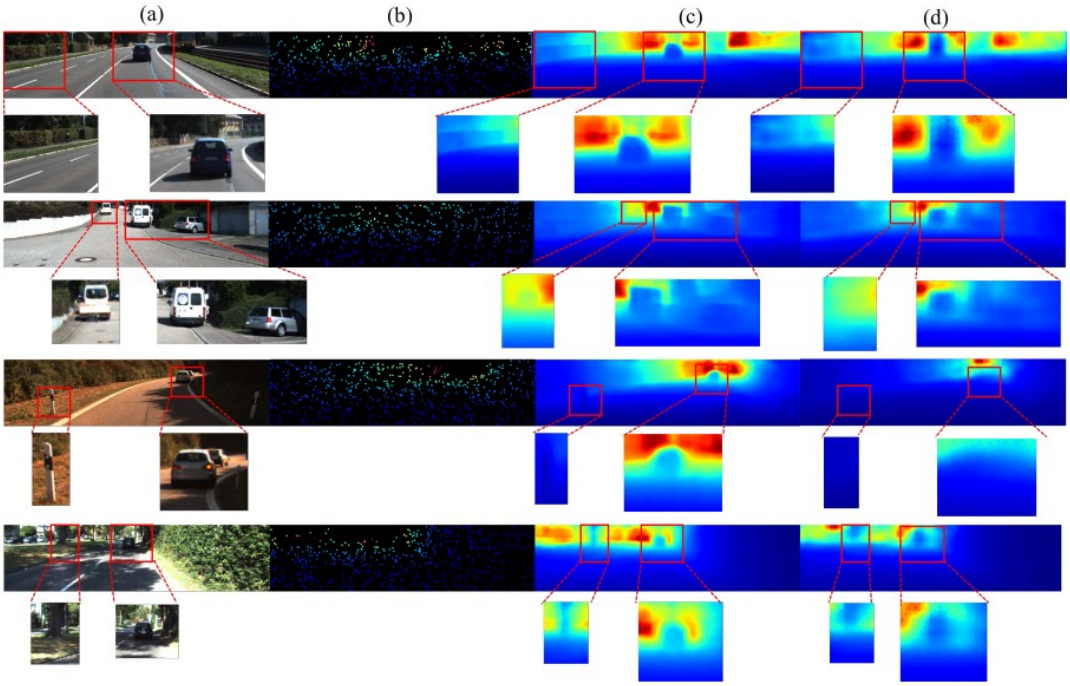
Đầu vào cho bất kỳ cấu hình mạng nào được coi là hình ảnh có kích thước cố định 224 x 224. Quá trình xử lý trước duy nhất được thực hiện là chuẩn hóa các giá trị RGB cho mỗi pixel. Hình ảnh được chuyển qua ngăn xếp đầu tiên gồm 2 lớp tích chập có kích thước tiếp nhận rất nhỏ là 3 x 3, tiếp theo là kích hoạt ReLU. Mỗi lớp trong số hai lớp này chứa 64 bộ lọc. Sải tích chập được cố định ở 1 pixel và phần đệm là 1 pixel. Cấu hình này bảo toàn độ phân giải không gian và kích thước của bản đồ kích hoạt đầu ra giống với kích thước hình ảnh đầu vào. Các bản đồ kích hoạt sau đó được chuyển qua tổng hợp tối đa theo không gian trên cửa sổ 2 x 2 pixel, với khoảng cách là 2 pixel. Điều này làm giảm một nửa kích thước của các lần kích hoạt. Do đó, kích thước của các kích hoạt ở cuối ngăn xếp đầu tiên là 112 x 112 x 64. Các kích hoạt sau đó chảy qua ngăn xếp thứ hai tương tự, nhưng với 128 bộ lọc so với 64 trong ngăn xếp thứ nhất. Do đó, kích thước sau ngăn xếp thứ hai trở thành 56 x 56 x 128. Tiếp theo là ngăn xếp thứ ba với ba lớp tích chập và một lớp tổng hợp tối đa. Không. số bộ lọc được áp dụng ở đây là 256, làm cho kích thước đầu ra của ngăn xếp là 28 x 28 x 256. Tiếp theo là hai ngăn xếp gồm ba lớp tích chập, với mỗi ngăn chứa 512 bộ lọc. Đầu ra ở cuối cả hai ngăn xếp này sẽ là 7 x 7 x 512. Các chồng lớp chập trùng được theo sau bởi ba lớp được kết nối hoàn chỉnh với một lớp làm phẳng ở giữa. Hai lớp đầu tiên có 4.096 nơ ron mỗi lớp và lớp được kết nối đầy đủ cuối cùng đóng vai trò là lớp đầu ra và có 1.000 nơ ron tương ứng với 1.000 lớp có thể có cho tập dữ liệu ImageNet. Tiếp theo là lớp đầu ra là lớp kích hoạt Softmax được sử dụng để phân loại.

Một số trường hợp sử dụng mà bạn có thể thấy VGG16 được sử dụng thực tế: (1) nhận dạng hoặc phân loại hình ảnh - VGG16; (2) Phát hiện và bản địa hóa hình ảnh và (3) Vectơ nhúng hình ảnh - được sử dụng cho một vấn đề như xác minh khuôn mặt.

2.5. Phát biểu Bài toán:

2.5.1. Input và Output

- Input: Tập dữ liệu hình ảnh RGB (hoặc Gray-scale)
- Output: Tập dữ liệu hình ảnh có depth map



Hình 9 Kết quả trực quan trên bộ dữ liệu KITTI.
(a) Hình ảnh RGB (b) Sparse depth 500 điểm làm đầu vào (c) Depth map đã hoàn thành (d) Kết quả.

2.5.2. Hướng tiếp cận

- Sử dụng mô hình máy học (CFCNet) và áp dụng kĩ thuật kết hợp mô hình.
- Sử dụng Deep Canonical Correlation Analysis (DCCA)

2.5.3. Thách thức:

- Phân tích tương quan chuẩn chỉ phản ánh phương sai được chia sẻ bởi tổng hợp tuyến tính, không phản ánh phương sai được trích xuất từ biến.

- Các số liệu thống kê chính xác chưa được phát triển để suy diễn các phân tích chính tắc.
- Trọng lượng chính tắc (canonical weight) có rất nhiều sự không ổn định. (Joshua, 2016).

2.5.4. Cài đặt

- Hệ điều hành: Linux
- Python 3
- PyTorch 1.0+
- NVIDIA GPU + CUDA CuDNN
- Thư viện: matplotlib, cv2, PIL

2.6. Phương pháp Nghiên cứu và Tiến hành:

2.6.1. Phương pháp:

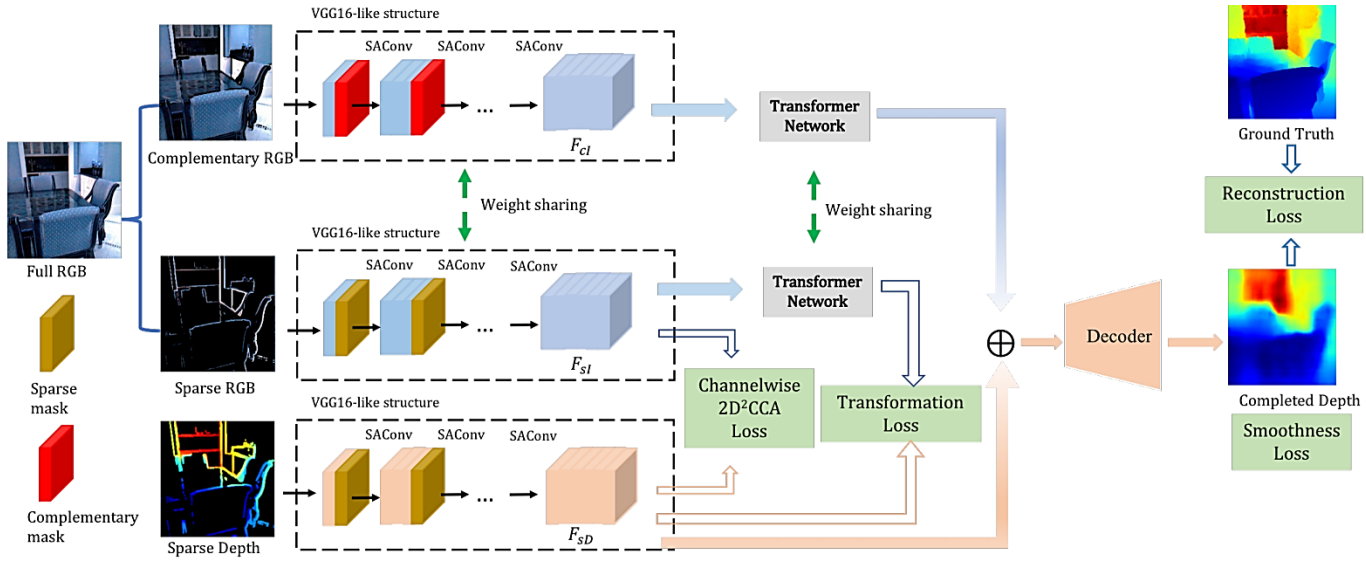
Cấu trúc Correlation For Completion Network (CFCNet, Hình 10) được đề xuất có sparse depth map, sparse RGB và complementary RGB. Sử dụng Sparsity-aware Attentional Convolutions (SAConv, như được hiển thị trong Hình 12) trong các bộ mã hóa giống như VGG16.

Hầu hết các phương pháp học sâu đa phương thức chỉ đơn giản là kết hợp (concatenate) hoặc thêm các đặc điểm bottleneck. Tuy nhiên, khi trích xuất ngữ nghĩa (semantics) và phạm vi giá trị của các đặc điểm khác nhau giữa các yếu tố, việc kết hợp và bổ sung trực tiếp trên nguồn dữ liệu đa phương thức sẽ không phải lúc nào cũng mang lại hiệu suất tốt hơn so với nguồn dữ liệu một phương thức. Để tránh vấn đề này, sử dụng bộ mã hóa để trích xuất ngữ nghĩa cấp cao hơn từ hai nhánh, ở đây đề xuất 2D^{2}CCA để đảm bảo các đặc điểm được trích xuất từ hai nhánh có mối tương quan tối đa. Mục đích là nắm bắt cùng một ngữ nghĩa từ các miền RGB và độ sâu. Tiếp theo, sử dụng một transformer network để chuyển đổi các đặc điểm được trích xuất từ miền RGB sang miền chiều sâu, làm cho các đặc điểm được trích xuất từ các nguồn khác nhau chia sẻ cùng một phạm vi số. Trong giai đoạn đào tạo, sử dụng các đặc điểm có sparse depth và hình ảnh sparse RGB tương ứng để tính toán 2D^{2}CCA Loss và Transformer Loss. Đối với đầu vào, kết hợp các đặc điểm sparse depth với các đặc điểm reconstructed missing depth. Các đặc điểm của reconstructed missing depth được trích xuất từ hình ảnh complementary RGB thông qua bộ mã hóa RGB và transformer. Để đảm bảo đào tạo một giai đoạn, chúng tôi áp dụng các thuật chia sẻ trọng lượng (weight-sharing strategies, Hình).

2.6.2. Framework:

2.6.2.1. Kiến trúc

Bộ mã hóa của tương tự như VGG-16 nhưng không có các lớp Fully Connected. ReLU được sử dụng trên các đặc điểm được trích xuất sau mỗi thao tác SAConv. Downsampling được áp dụng cho cả các đặc điểm và mặt nạ trong bộ mã hóa. Transformer network là mạng 2 lớp, kích thước 3×3, stride 1 và trong không gian 512 chiều, và SAConv. Bộ giải mã cũng là một mạng giống như VGG16 sử dụng giải mã để upsample. Kiến trúc này sử dụng trình tối ưu hóa SGD.



Hình 10 Kiến trúc CFCNet. Ở đây ⊕ là cho concatenation operation. Input 0 - 1 sparse mask đại diện cho mô hình đo chiều sâu thưa thớt. Complementary mask bổ sung cho sparse mask. Tách 1 hình ảnh RGB đầy đủ thành sparse RGB và complementary RGB bằng mask và đưa chúng bằng vào mạng bằng mask.

Xây dựng mạng từ các lớp sẽ được xây dựng bên dưới

```
def define_DCCASparseNet(rgb_enc=True, depth_enc=True, depth_dec=True, norm='batch',
init_type='xavier', init_gain=0.02, gpu_ids=[]):
    net = None
    norm_layer = get_norm_layer(norm_type=norm)
    net = DCCASparsenetGenerator(
        rgb_enc=rgb_enc, depth_enc=depth_enc, depth_dec=depth_dec)
    return init_net(net, init_type, init_gain, gpu_ids)
```

2.6.2.2. Backbone / Deep Network

2.6.2.2.1. Transformer Network model

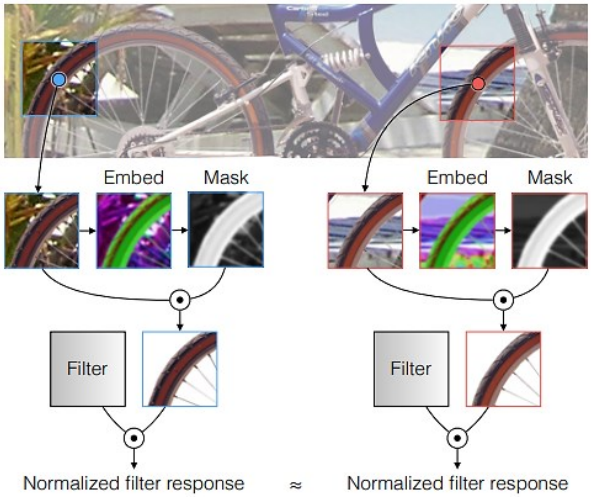
Giống như các mô hình seq2seq trước đó, mô hình Transformer ban đầu sử dụng kiến trúc **bộ mã hóa-giải mã**. Bộ mã hóa bao gồm các lớp mã hóa xử lý đầu vào lặp đi lặp lại hết lớp này sang lớp khác, trong khi bộ giải mã bao gồm các lớp giải mã làm điều tương tự với đầu ra của bộ mã hóa.

Chức năng của mỗi lớp bộ mã hóa là tạo ra các mã hóa có chứa thông tin về phần đầu vào nào có liên quan đến nhau. Nó chuyển mã hóa của nó sang lớp bộ mã hóa tiếp theo làm đầu vào. Mỗi lớp giải mã làm ngược lại, lấy tất cả các mã hóa và sử dụng thông tin theo ngữ cảnh kết hợp của chúng để tạo ra một chuỗi đầu ra. Để đạt được điều này, mỗi bộ mã hóa và lớp giải mã sử dụng một cơ chế chú ý (attention).

Đối với mỗi đầu vào, attention cân nhắc (weigh) sự liên quan của mọi đầu vào khác và rút ra từ chúng để tạo ra đầu ra. Mỗi lớp giải mã có một cơ chế chú ý bổ sung thu hút thông tin từ đầu ra của các bộ giải mã trước đó, trước khi lớp giải mã lấy thông tin từ mã hóa.

Cả hai lớp bộ mã hóa và bộ giải mã đều có mạng lưới thần kinh chuyển tiếp nguồn cấp dữ liệu để xử lý thêm đầu ra và chứa các kết nối còn lại và các bước chuẩn hóa lớp.

2.6.2.2.2. SAConv



Hình 11 Các bộ lọc Segmentation-aware convolution là bất biến đối với nền. (i) tính toán tín hiệu phân đoạn cho mỗi pixel (tức là "nhúng"), (ii) tạo foreground mask cho mỗi patch và (iii) kết hợp mask với convolution, để các bộ lọc chỉ xử lý local foreground trong mỗi patch hình ảnh.

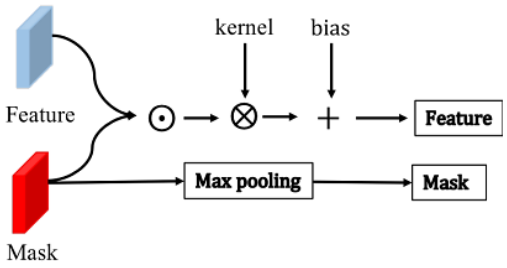
Segmentation-aware mask: Để giải quyết vấn đề làm mờ, segmentation-aware convolutional networks được đề xuất. Mạng này điều chỉnh hành vi của chúng trên cơ sở mỗi pixel theo tín hiệu phân đoạn (segmentation cues), để các bộ lọc có thể "tham dự" thông tin đến từ khu vực có neuron một cách chọn lọc và xử lý nó khác với tín hiệu nền (background signals). Để đạt được điều này, bổ sung cho mỗi patch hình ảnh với mặt nạ phân đoạn tiền cảnh cục bộ (local foregroundbackground segmentation mask) hoạt động như một cơ chế gating cho thông tin đưa vào neuron. Điều này tránh làm mờ đặc điểm, bằng cách giảm mức độ hòa lẫn thông tin tiền cảnh và bối cảnh và cho phép mức độ kích hoạt tế bào thần kinh thay đổi nhanh chóng, bằng cách tự động điều chỉnh hành vi của neuron với nội dung hình ảnh. Điều này còn hơn cả việc làm sắc nét các đầu ra mạng học sâu, như thực tế phổ biến hiện nay; nó khắc phục vấn đề làm mờ "trước khi thiệt hại được thực hiện", vì nó có thể được tích hợp ở cả giai đoạn đầu và sau của CNN.

Hadamard product là một hàm nhị phân lấy hai ma trận có cùng kích thước và tạo ra một ma trận khác có cùng kích thước với các toán hạng, trong đó mỗi phần tử i, j là sản phẩm của các yếu tố i, j của hai ma trận ban đầu. Cho hai ma trận $\boldsymbol{A}$ và $\boldsymbol{B}$ có cùng kích thước $\boldsymbol{m} \times \boldsymbol{n}$, kết quả Hadamard $\boldsymbol{A} \odot \boldsymbol{B}$ là một ma trận có cùng kích thước với các toán hạng:

$$(\boldsymbol{A} \odot \boldsymbol{B})_{\boldsymbol{ij}} = (\boldsymbol{A})_{\boldsymbol{ij}} (\boldsymbol{B})_{\boldsymbol{ij}}$$

SAConv được lấy cảm hứng từ local attention mask. Harley và cộng sự giới thiệu segmentation-aware mask để cho phép các hàm khai thác convolutionals "tập trung" vào các tín hiệu phù hợp với segmentation mask. Để truyền thông tin từ các nguồn đáng tin cậy, sparsity masks được sử dụng để làm cho các sparsity mask tham gia vào các tín hiệu từ các địa điểm đáng tin cậy. Sự khác biệt của SAConv và local attention mask là SAConv không áp dụng normalized

mask vì normalized mask ảnh hưởng đến sự ổn định của các tính toán 2DDCCA sau do các đặc điểm chiết xuất nhỏ về số lượng mà nó tạo ra sau nhiều lần chuẩn hóa. Ngoài ra, maxpooling operations được sử dụng trên mask sau mỗi SAConv để theo dõi khả năng hiển thị, nếu có ít nhất một giá trị nonzero hiển thị với một convolutional kernel, maxpooling sẽ đánh giá giá trị tại vị trí đó đến 1.



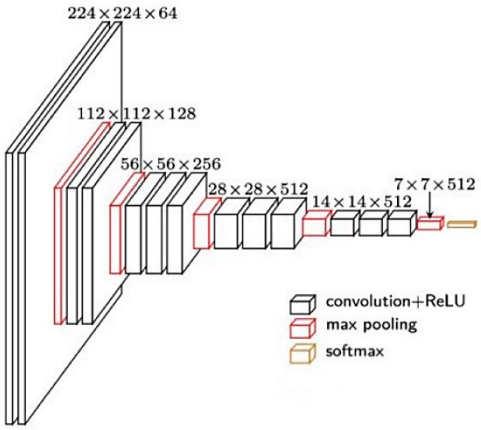
Hình 12 Kiến trúc SAConv
⊙ là cho sản phẩm Hadamard. ⊗ là cho convolution. + là để bổ sung nguyên tố.
Kích thước kernel là 3×3 và stride là 1 cho cả convolution và maxpooling.

Xây dựng SAConv Block với lớp Convolution cho sparse data

```
class SAConvBlock(nn.Module):
    def __init__(self, in_channel, out_channel, kernel_size, stride=1, padding=1,
dilation=1, bias=True):
        super(SAConvBlock, self).__init__()
        self.sparse_conv = SAConv(in_channel, out_channel, kernel_size, stride=stride,
                                  padding=padding, dilation=dilation, bias=True)
        self.relu = nn.ReLU(inplace=True)
    def forward(self, input):
        x, m = input
        x, m = self.sparse_conv((x, m))
        assert (m.size(1) == 1)
        x = self.relu(x)
        return x, m
```

2.6.2.3. Encoder – Decoder

Encoder – Decoder của kiến trúc này có cấu trúc gần giống với VGG-16 như đã nêu trên, tuy nhiên, CFCNet không có lớp Fully Connected như VGG-16.



Hình 13 Kiến trúc Encoder - Decoder của CFCNet rút ra được từ VGG-16

Encoder: Mỗi bộ mã hóa bao gồm hai thành phần chính: cơ chế tự chú ý (attention) và feed-forward neural network. Cơ chế tự chú ý chấp nhận mã hóa đầu vào từ bộ mã hóa trước đó và cân nhắc sự liên quan của chúng với nhau để tạo mã hóa đầu ra. Mạng lưới thần kinh chuyển tiếp nguồn cấp dữ liệu tiếp tục xử lý từng mã hóa đầu ra riêng lẻ. Các mã hóa đầu ra này sau đó được chuyển đến bộ mã hóa tiếp theo làm đầu vào của nó, cũng như các bộ giải mã. Bộ mã hóa

đầu tiên lấy thông tin vị trí và nhúng của chuỗi đầu vào làm đầu vào của nó, thay vì mã hóa. Thông tin vị trí là cần thiết cho máy biến áp để sử dụng thứ tự của chuỗi, bởi vì không có phần nào khác của máy biến áp sử dụng điều này.

Decoder: Mỗi bộ giải mã bao gồm ba thành phần chính: một cơ chế tự chú ý, một cơ chế chú ý trên mã hóa, và một feed-forward neural network. Bộ giải mã hoạt động theo cách tương tự như bộ mã hóa, nhưng một cơ chế chú ý bổ sung được chèn vào thay vào đó rút ra thông tin liên quan từ các mã hóa được tạo ra bởi các bộ mã hóa. Giống như bộ mã hóa đầu tiên, bộ giải mã đầu tiên lấy thông tin vị trí và nhúng chuỗi đầu ra làm đầu vào của nó, thay vì mã hóa. Transformer không được sử dụng đầu ra hiện tại hoặc tương lai để dự đoán đầu ra, vì vậy trình tự đầu ra phải được che giấu một phần để ngăn chặn luồng thông tin ngược này. Bộ giải mã cuối cùng được theo sau bởi một sự biến đổi tuyến tính cuối cùng và lớp softmax, để tạo ra xác suất đầu ra so với từ vựng (vocabulary).

Xây dựng lớp Deconvolution với lớp decoder nền

```
class DeConv(Decoder):
    def __init__(self, in_channels, kernel_size):
        assert kernel_size >= 2, "kernel_size out of range: {}".format(
            kernel_size)
        super(DeConv, self).__init__()
        def convt(in_channels):
            stride = 2
            padding = (kernel_size - 1) // 2
            output_padding = kernel_size % 2
            assert -2 - 2*padding + kernel_size + \
                output_padding == 0, "deconv parameters incorrect"
            module_name = "deconv{}".format(kernel_size)
            return nn.Sequential(collections.OrderedDict([
                (module_name, nn.ConvTranspose2d(in_channels, in_channels//2,
kernel_size,
                                                 stride, padding, output_padding,
bias=False)),
                ('batchnorm', nn.BatchNorm2d(in_channels//2)),
                ('relu'    ,      nn.ReLU(inplace=True)),   ]))
        self.layer1 = convt(in_channels)
        self.layer2 = convt(in_channels // 2)
        self.layer3 = convt(in_channels // (2 ** 2))
        self.layer4 = convt(in_channels // (2 ** 3))
```

2.6.2.4. Metrics và Loss Function

2.6.2.4.1. 2D Deep Canonical Correlation Analysis (2D^{2}CCA)

Vì các nghiên cứu thị giác máy tính hiện đại thường sử dụng mạng sâu để trích xuất thông tin từ hình ảnh có độ phân giải tương đối lớn, kích thước batch bị giới hạn bởi việc sử dụng bộ nhớ GPU. Do đó, khi biểu diễn các đặc điểm tiềm ẩn trong high-dimesional networks, sử dụng DCCA với đại diện vector một chiều sẽ dẫn đến vấn đề SSS. Nhưng đối với phân tích tương quan chính tắc sâu 2 chiều (2DDCCA), một số hạn chế sẽ được khắc phục.

Biểu diễn bản đồ độ sâu đã hoàn thành (completed depth map) là $\boldsymbol{D}$ với hình ảnh RGB tương ứng là $\boldsymbol{I}$. Sparse depth map trong đầu vào và hình ảnh sparse RGB tương ứng are được ký hiệu là $\boldsymbol{sD}$ and $\boldsymbol{sI}$. RGB/Depth encoders được kí hiệu là $\boldsymbol{f_D}$ and $\boldsymbol{f_I}$ trong đó các tham số của bộ mã hóa được kí hiệu là $\boldsymbol{\theta_D}$ and $\boldsymbol{\theta_I}$ tương ứng. Như đã mô tả ở trên, $\boldsymbol{f_D}$ và $\boldsymbol{f_I}$ sử dụng SAConv để truyền thông tin từ các điểm đáng tin cậy để trích xuất các đặc điểm từ spares input. We tạo lưới đặc điểm 3D nhúng cặp ($\boldsymbol{F_{sD}} \in \mathbb{R}^{\boldsymbol{m*n*C}}$, $\boldsymbol{F_{sI}} \in \mathbb{R}^{\boldsymbol{m*n*C}}$) cho mỗi cặp bản đồ/hình ảnh spares depth ($\boldsymbol{sD}, \boldsymbol{sI}$) bằng cách định nghĩa $\boldsymbol{F_{sD}} = \boldsymbol{f_D}(\boldsymbol{sD};\ \boldsymbol{\theta_D})$ and $\boldsymbol{F_{sI}} = \boldsymbol{f_I}(\boldsymbol{sI};\ \boldsymbol{\theta_I})$. Bên trong mỗi cặp lưới tính năng, có $\boldsymbol{C}$ các cặp bản đồ

tính nặng ($F_{sD}^{i} \in \mathbb{R}^{m*n},\ F_{sI}^{i} \in \mathbb{R}^{m*n}$), $\forall i < C$ và $C = 521$. Thay vì phân tích mối tương quan toàn cầu giữa bất kỳ cặp nào có thể có (F_{sD}^{i}, F_{sI}^{j}), $\forall i \neq j$ chúng tôi phân tích mối tương quan chuẩn kênh giữa cùng một số kênh (F_{sD}^{i}, F_{sI}^{i}). Phân tích tương quan kênh này sẽ dẫn đến việc nhận được các tính năng có ý nghĩa ngữ nghĩa tương tự cho từng phương thức, hướng dẫn f_I để nhúng thêm thông tin có giá trị liên quan đến việc hoàn thành độ sâu.

Sử dụng biểu diễn tính năng 1 chiều sẽ dẫn đến vấn đề SSS trong nhiệm vụ thị giác máy tính dựa trên học sâu hiện đại, đó là lý do tại sao phương pháp tiếp cận 2 chiều được sử dụng ở đây để tạo ma trận hiệp phương sai xếp hạng đầy đủ $\widehat{\sum}_{sD,sI} \in \mathbb{R}^{m*n}$ với $E[F] = \frac{1}{C}\sum_{i=0}^{C-1} F^{i}$:

$$\widehat{\sum}_{sD,sI} = \frac{1}{C}\sum_{i=0}^{C-1}\left[F_{sD}^{i} - E[F_{sD}]\right]\left[F_{sI}^{i} - E[F_{sI}]\right]^{T}$$

Bên cạnh đó, tạo ma trận hiệp phương sai $\widehat{\sum}_{sD}$ and $\widehat{\sum}_{sI}$ với hằng số chính quy hóa r_1 hoặc r_2 và ma trận nhận dạng I:

$$\widehat{\sum}_{sD} = \frac{1}{C}\sum_{i=0}^{C-1}\left[F_{sD}^{i} - E[F_{sD}]\right]\left[F_{sD}^{i} - E[F_{sD}]\right]^{T} + r_1 I$$

$$\widehat{\sum}_{sI} = \frac{1}{C}\sum_{i=0}^{C-1}\left[F_{sI}^{i} - E[F_{sI}]\right]\left[F_{sI}^{i} - E[F_{sI}]\right]^{T} + r_2 I$$

Mối tương quan giữa F_{sD} và F_{sI}:

$$corr(F_{sD}, F_{sI}) = \left\|\left(\widehat{\sum_{sD}}^{-1/2}\right)\left(\widehat{\sum}_{sD,sI}\right)\left(\widehat{\sum_{sI}}^{-1/2}\right)\right\|_{tr}$$

Giá trị cao hơn của mối tương quan giữa F_{sD} và F_{sI} represents mối tương quan cao hơn giữa hai khối tính năng. Vì $corr(F_{sD}, F_{sI})$ là một vô hướng không âm, $-corr(F_{sD}, F_{sI})$) được sử dụng làm mục tiêu tối ưu hóa để hướng dẫn đào tạo hai bộ mã hóa tính năng.

Để tính toán gradient của $corr(F_{sD}, F_{sI})$ với θ_D and θ_I chúng ta có thể tính toán gradient của nó với F_{sD} và F_{sI}, then làm tuyên truyền trở lại. Về tính toán gradient, chúng tôi xác định $M = \left(\widehat{\sum_{sD}}^{-1/2}\right)\left(\widehat{\sum}_{sD,sI}\right)\left(\widehat{\sum_{sI}}^{-1/2}\right)$ và phân rã M as $M = USV^{T}$ sử dụng phân rã SVD. Với $\nabla_{sDsI} = \widehat{\sum_{sD}}^{-1/2} UV^{T} \widehat{\sum_{sRGB}}^{-1/2}$ và $\nabla_{sDsD} = -\frac{1}{2}\widehat{\sum_{sD}}^{-1/2} UDU^{T} \widehat{\sum_{sD}}^{-1/2}$:

$$\frac{\partial corr(F_{sD}, F_{sI})}{\partial F_{sI}} = \frac{1}{C}(2\nabla_{sDsD} F_{sD} + \nabla_{sDsI} F_{sI}$$

Tương tự với $\frac{\partial corr(F_{sD}, F_{sI})}{\partial F_{sD}}$

2.6.2.4.2. Error Metrics

Lỗi bình phương trung bình căn (RMSE) là một biện pháp thường được sử dụng trong những khác biệt giữa các giá trị (mẫu hoặc các giá trị dân) được dự đoán bởi một mô hình hay một ước lượng và các giá trị quan sát được. RMSE đại diện cho căn bậc hai của thời điểm mẫu thứ hai về sự khác biệt giữa các giá trị dự đoán và giá trị quan sát hoặc giá trị trung bình bậc hai của những khác biệt này. Các độ lệch này được gọi là phần dư khi các phép tính được thực hiện trên mẫu dữ liệu được sử dụng để ước tính và được gọi là lỗi (hoặc lỗi dự đoán) khi tính toán ngoài mẫu. RMSE phục vụ để tổng hợp cường độ của các lỗi trong các dự đoán trong nhiều thời điểm khác nhau thành một thước đo duy nhất về sức mạnh dự đoán. RMSE là thước đo độ chính xác, để so sánh các lỗi dự báo của các mô hình khác nhau cho một tập dữ liệu cụ thể chứ không phải giữa các bộ dữ liệu, vì nó phụ thuộc vào quy mô.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N}\|y(i) - \hat{y}(i)\|^{2}}{N}}$$

Sai số tuyệt đối trung bình (MAE) là một phương pháp đo lường sự khác biệt giữa hai biến liên tục. Giả sử rằng $\boldsymbol{X}$ và $\boldsymbol{Y}$ là hai biến liên tục thể hiện kết quả dự đoán của mô hình và kết quả thực tế. MAE được tính theo công thức sau:

$$\boldsymbol{MAE} = \frac{\sum_{i=1}^{N}|\boldsymbol{y_i} - \boldsymbol{x_i}|}{\boldsymbol{N}} = \frac{\sum_{i=1}^{N}|\boldsymbol{e_i}|}{\boldsymbol{N}}$$

Độ đo này thường được sử dụng để đánh giá sự sai khác giữa mô hình dự đoán và tập dữ liệu testing trong các bài toán hồi quy. Chỉ số này càng nhỏ thì mô hình học máy càng chính xác.

Standard metric (δ_i): phần trăm pixel dự đoán trong đó lỗi tương đối nằm trong khoảng 1.25^i, với i = 1, 2, 3.

2.6.2.4.3. Loss Function

```
# specify the training losses you want to print out
# The program will call base_model.get_current_losses
        self.loss_names = ['mse','dcca','total','transform','smooth']
# define loss functions
        self.criterionDCCA = DCCA_sparse_networks.DCCA_2D_Loss(outdim_size =
60,use_all_singular_values = True, device=self.device).to(self.device)
        self.MSE = DCCA_sparse_networks.MaskedMSELoss()
        self.SMOOTH = DCCA_sparse_networks.SmoothLoss()
        self.TransformLoss = DCCA_sparse_networks.TransformLoss()
def get_loss(self):
        self.loss_dcca = self.criterionDCCA(self.x_dataview,self.y_dataview)
        self.loss_mse = self.MSE(self.depth_est,self.depth_image)
        self.loss_smooth = self.SMOOTH(self.depth_est)
        self.loss_transform = self.TransformLoss(self.x_trans, self.x_dataview)
        self.loss_total = self.loss_mse + self.loss_dcca + self.loss_transform +
0.1*self.loss_smooth
```

Thành phần được chuyển đổi từ sparse RGB sang miền độ sâu là $\widehat{\boldsymbol{F_{sD}}}$

DCCA loss như đã phân tích ở mục DDCA và 2DDCCA:

$$\boldsymbol{L_{2DDCCA}} = -\boldsymbol{corr}(\boldsymbol{F_{sD}}, \boldsymbol{F_{sI}})$$

Transformer loss mô tả sự tương đồng về số giữa RGB và miền độ sâu, định mức L_2 được sử dụng để đo lường sự tương đồng về số:

$$\boldsymbol{L_{trans}} = \left\|\boldsymbol{F_{sD}} - \widehat{\boldsymbol{F_{sD}}}\right\|_2^2$$

Reconstruction loss: Một bộ mã hóa khác và một transformer network khác chia sẻ trọng lượng với bộ mã hóa và transformer network cho RGB dự phòng cũng được xây dựng. Đầu vào của bộ mã hóa là một hình ảnh RGB bổ sung. Chúng tôi sử dụng các đặc điểm được trích xuất từ hình ảnh RGB bổ sung để dự đoán các đặc điểm có độ sâu không thể quan sát được bằng cách sử dụng transformer network. Đối với hình ảnh RGB bổ sung, đặc điểm được trích xuất và thành phần được chuyển đổi được kí hiệu là $\boldsymbol{F_{cI}}$ và $\widehat{\boldsymbol{F_{cD}}}$. Sau đó liên kết $\boldsymbol{F_{sD}}$ và $\widehat{\boldsymbol{F_{cD}}}$, cả hai đều có 512 kênh. Từ đó có được một đặc điểm bottleneck 1024 kênh trên miền độ sâu và truyền đặc điểm này vào bộ giải mã như đã nêu trên. Đầu ra từ bộ giải mã là một completed depth map $\widehat{\boldsymbol{D}}$. Để so sánh sự không nhất quán giữa các groundtruth $\boldsymbol{D_{gt}}$ và completed depth map, sử dụng pixelwise định mức $\boldsymbol{L_2}$:

$$\boldsymbol{L_{recon}} = \left\|\boldsymbol{D_{gt}} - \widehat{\boldsymbol{D}}\right\|_2^2$$

MSE loss: Heatmap được hồi quy bằng cách sử dụng MSE loss, tương tự như các đường cơ sở đơn giản. Với $\boldsymbol{C_k^p} = \{\boldsymbol{x_k^p}, \boldsymbol{y_k^p}\}$ là tọa độ của đặc điểm thứ $\boldsymbol{k}$ của đối tượng thứ $\boldsymbol{p}$ cần rút trích. Các đặc điểm của các đối tượng trong cùng khung hình nằm ở các miền phân biệt ngữ nghĩa khác nhau, nên CFCNet cần xác định chúng bằng việc che phủ các miền với những Gaussian Kernel có độ lệch chuẩn khác nhau (CFCNet tự học và cập nhật độ lệch chuẩn thông qua dữ liệu dùng để training).

Xác định các tỉ lệ s theo hình dạng của heatmap (bản đồ nhiệt). Với mỗi đặc điểm cần rút trích $\boldsymbol{C_k^p} = \{\boldsymbol{x_k^p}, \boldsymbol{y_k^p}\}$, điều chỉnh độ lệch chuẩn thành $\boldsymbol{\sigma_0 . s_{k,x_k^p,y_k^p}}$. Khi đó, vùng được phủ bởi Gaussian Kernel cho đặc điểm cần rút trích trở thành:

$$\boldsymbol{h_{k,i,j}^p = e^{-\frac{(i-x_k^p)^2+(j-y_k^p)^2}{2\left(\sigma_0 . s_{k,x_k^p,y_k^p}\right)^2}}}$$

với $\boldsymbol{\|i - x_k^p\|_1 \leq 3\sigma,\ \|j - y_k^p\|_1 \leq 3\sigma.}$ Tuy nhiên, vùng phủ này tương đối nhỏ ($\boldsymbol{s_{k,x_k^p,y_k^p} \sim s_{k,i,j}}$) với điều kiện như trên, nên ta có thể thay $\boldsymbol{s_{k,x_k^p,y_k^p}}$ bằng $s_{k,i,j}$. Khi đó ta có bản đồ nhiệt (heatmap) mới, kí hiệu là $\boldsymbol{H^{\sigma_0 . s}}$(heatmap thích nghi tỉ lệ). Mối quan hệ giữa heatmap mới sau khi phủ Gaussian Kernel và heatmap cũ là:

$$\boldsymbol{H_{k,i,j}^{\sigma_0 . s} = \begin{cases} (H_{k,i,j}^{\sigma_0})^{\frac{1}{s_{k,i,j}}} \text{ với } H_{k,i,j}^{\sigma_0} > 0 \\ H_{k,i,j}^{\sigma_0} \text{ với } H_{k,i,j}^{\sigma_0} = 0 \end{cases}}$$

Đối với các đặc điểm cần rút trích có thành phần tỉ lệ lớn hơn 1, độ lệch chuẩn của chúng sẽ vượt $\boldsymbol{\sigma_0}$ và phần gaussian kernel phủ sẽ rộng hơn. Để mô hình ổn định, cần biến đổi một chút để tránh overfitting nhưng vẫn không mất tính tổng quát của mô hình. Hàm mất mát (Loss function) cho các tỉ lệ đã được xác định:

$$\boldsymbol{L_{reg} = \left\| \left(\frac{1}{s} - 1 \right) 1_{H^{\frac{\sigma_0}{s}} > 0} \right\|_2^2}$$

Với $\boldsymbol{1_{H^{\frac{\sigma_0}{s}} > 0}}$ chỉ các miền được phủ bởi gaussian kernel. Đồng thời, biến đổi dạng lũy thừa của $\boldsymbol{H^{\sigma_0 . s}}$ thành chuỗi đa thức bằng khai triển Taylor với $\boldsymbol{s = 1}$ được:

$$\boldsymbol{H_{k,i,j}^{\sigma_0 . s} = \begin{cases} \frac{1}{2} H_{k,i,j}^{\sigma_0} (1 + (1 + \propto_{k,i,j} ln\left(H_{k,i,j}^{\sigma_0}\right))^2) \text{ với } H_{k,i,j}^{\sigma_0} > 0 \\ 0 \text{ với } H_{k,i,j}^{\sigma_0} = 0 \end{cases}}$$

Với $\boldsymbol{\propto = \frac{1}{s} - 1}$. Khi đó, hàm mất mát là với tham số regularize λ được chọn là 1 để cân bằng hai đại lượng vế phải:

$$\boldsymbol{L_{MSE} = L_{regression} + \lambda L_{reg} = \|P - H^{\sigma_0 . s}\|_2^2 + \lambda \left\| \propto \left(1_{H^{\frac{\sigma_0}{s}} > 0} \right) \right\|_2^2}$$

Smooth loss để khắc phục tình trạng hiệu ứng lưới khi sparsity của đầu vào nặng, vì các đặc điểm của bottleneck có hạn chế, ví dụ: chỉ có 0,1% điểm lấy mẫu của toàn bộ độ phân giải, các depth completed map thường có hiệu ứng lưới. Với $\boldsymbol{\nabla^2}$ là đạo hàm thứ 2 của trong gradient.

$$\boldsymbol{L_{smooth} = \left\| \nabla^2 \hat{D} \right\|_1}$$

Total loss:

$$\boldsymbol{L_{total} = L_{2DDCCA} + w_t L_{trans} + w_r L_{recon} + w_s L_{smooth}}$$

2.6.2.5. Tập dữ liệu

2.6.2.5.1. Cityscape (Outdoor scene)

Tập dữ liệu của Cityscape chứa RGB và các depth map được tính toán từ kết hợp âm thanh nổi của các cảnh ngoài trời. Bài nghiên cứu sử dụng phần tách tập dữ liệu đào tạo / xác thực chính thức. Tập huấn luyện chứa 23K hình ảnh từ 41 trình tự và tập kiểm tra chứa 3 trình tự, giữa các hình ảnh được cắt thành kích thước 900 × 335 để tránh bầu trời trên và biểu tượng ô tô phía dưới.

2.6.2.5.2. KITTI (Outdoor scene)

Tập dữ liệu của KITTI chứa cả phép đo RGB và LiDAR, tổng cộng 22 chuỗi để sử dụng lái xe tự động. Bài nghiên cứu sử dụng phần tách chính thức, trong đó 46K hình ảnh dành cho đào tạo và 46K để thử nghiệm. Hình ảnh được giảm phần trên và kích thước được thay đổi thành 912 x 228

2.6.2.5.3. NYUv2 (Indoor scene)

Tập dữ liệu của NYUv2 chứa 464 chuỗi dữ liệu độ sâu và RGB trong nhà bằng cách sử dụng Kinect. Bài nghiên cứu sử dụng phần tách tập dữ liệu chính thức và lấy mẫu 50K hình ảnh để làm dữ liệu đào tạo. Dữ liệu thử nghiệm chứa 654 hình ảnh.

2.6.2.5.4. SLAM RGBD (Indoor scene)

Sử dụng trình tự bộ dữ liệu ICL-NUIM và TUM RGBD SLAM từ máy ảnh âm thanh nổi. Đầu tiên là tổng hợp, sau đó thu được với Kinect.

2.6.2.6. Phân tích và đánh giá tập dữ liệu

2.6.2.6.1. Outdoor scene - KITTI and Cityscapes

Đối với KITTI và Cityscapes hai bộ dữ liệu ngoài trời này, sử dụng bộ lọc thưa đồng nhất. Đối với tập dữ liệu KITTI, lấy mẫu 500 điểm dưới dạng độ sâu thưa giống như một số công trình trước đó. Sau đó, thực hiện so sánh với một số công trình hiện đại, chọn ngẫu nhiên 3000 hình ảnh để tính toán kết quả.

Tiếp theo, bài nghiên cứu tiến hành thực nghiệm bằng cách sử dụng cả hai bộ dữ liệu KITTI và Cityscape. Một số công trình dự đoán độ sâu một mắt sử dụng tập dữ liệu Cityscape để đào tạo và tập dữ liệu KITTI để thử nghiệm - 100 mẫu độ sâu thưa thớt được lấy mẫu đồng nhất làm đầu vào.

	MAE	RMSE	δ1	δ2	δ3
Ma et al.Ma and Karaman [2018]	-	3.378	93.5	97.6	98.9
SPN Liu et al. [2017]	-	3.243	94.3	97.8	99.1
CSPNCheng et al. [2018]	-	3.029	95.5	98	99
CSPN+UNetCheng et al. [2018]	-	2.977	95.7	98	99.1
PnPWang et al. [2019a]	1.024	2.975	94.9	98	99
CFCNet w/o smoothness	1.233	2.967	94.1	98.1	99.3
CFCNet w/ smoothness	1.197	2.964	94	98	99.3

Bảng 1 Kết quả đánh giá độ sâu của KITTI dataset

Methods	Input	Dataset	RMSE	δ1	δ2	δ3
Zhou et al. Zhou et al. [2017]	RGB	CS→K	7.58	57.7	84	93.7
Godard et al. Godard et al. [2017]	RGB	CS→K	14.445	5.3	32.6	86.2
Aleotti et al. Aleotti et al. [2018]	RGB	CS→K	14.051	6.3	39.4	87.6
CFCNet(50 pts)	RGB+sD	CS →K	7.841	78.3	92.7	97
CFCNet(100 pts)	RGB+sD	CS→K	5.827	82.6	94.7	97.9
Zhou et al. Zhou et al. [2017](cap 50m)	RGB	CS→K	6.148	59	85.2	94.5
CFCNet(50 pts, cap 50m)	RGB+sD	CS →K	6.334	79.2	93.2	97.3
CFCNet(100 pts, cap 50m)	RGB+sD	CS →K	4.524	83.7	95.2	98.1
CFCNet(50 pts, cap 50m)	RGB+sD	CS→CS	9.019	82.8	94.1	97.2
CFCNet(100 pts, cap 50m)	RGB+sD	CS→CS	6.887	88.9	96.1	98.1
CFCNet(100 pts, cap 50m)	RGB+sD	K→K	3.157	91	97.1	98.9

Bảng 2 Kết quả đánh giá độ sâu của Cityscape để đào tạo và KITTI để thử nghiệm

2.6.2.6.2. Indoor scene - NYUv2 and SLAM RGBD datasets

Đối với tập dữ liệu NYUv2, bộ lộc âm thanh nổi được sử dụng để lấy mẫu các điểm. Tập dữ liệu này so sánh các loại hiện đại nhất và độ thưa thớt khác nhau.

Đối với tập dữ liệu Slam RGBD, bài nghiên cứu tuân theo cài đặt trong CNN-SLAM hiện đại nhất và thực hiện đánh giá tập dữ liệu chéo. Số liệu trong CNN-SLAM được sử dụng để tính tỉ lệ phần trăm ước tính chính xác. Ước tính chính xác có nghĩa là sai số nằm trong khoảng ± 10% so với mặt đất.

Sample#	Methods	MAE	RMSE	δ1	δ2	δ3
100	Ma and Karaman [2018]	0.473	0.629	71.5	92.4	98
100	CFCNet	0.426	0.58	77.5	94.1	98.4
200	Ma and Karaman [2018]	0.451	0.603	73	93.5	98.4
200	CFCNet	0.385	0.531	80.9	95.1	98.7
500	Ma and Karaman [2018]	0.384	0.529	79.2	94.9	98.6
500	CFCNet	0.342	0.476	83	96.1	99

Bảng 3 So sánh tập dữ liệu của NYUv2 khi sử dụng bộ khuếch đại âm thanh

Sequence#	CFCNet	CNN-SLAMTateno et al. [2017]	LainaLaina et al. [2016]	RemodePizzoli et al. [2014]
ICL/office0	41.97	19.41	17.19	4.47
ICL/office1	43.86	29.15	20.83	3.13
ICL/office2	63.64	37.22	30.63	16.7
ICL/living0	51.76	12.84	15	4.47
ICL/living1	64.34	13.03	11.44	2.42
ICL/living2	59.07	26.56	33.01	8.68
TUM/seq1	54.7	12.47	12.98	9.54
TUM/seq2	66.3	24.07	15.41	12.65
TUM/seq3	74.61	27.39	9.45	6.73
Average	57.81	22.46	18.44	7.64

Bảng 4 So sánh về tỷ lệ phần trăm độ sâu ước tính chính xác trên hai tập dữ liệu SLAM RGBD, ICL-NUIM và TUM.

2.7. Kết luận

Với Depth completion, bài nghiên trực tiếp phân tích mối quan hệ giữa thông tin sparse depth và các pixel tương ứng của chúng trong ảnh RGB. Để kết hợp thông tin tốt hơn, bài nghiên cứu sử dụng 2DDCCA để đảm bảo các ngữ nghĩa giống nhau nhất được thu thập từ hai nhánh và sử dụng thông tin RGB bổ sung để bổ sung độ sâu còn thiếu.

Do thời gian làm nghiên cứu còn hạn chế nên có thể đã bỏ qua một số yếu tố có thể ảnh hưởng đến kết quả phân tích của nghiên cứu.

3. TÀI LIỆU THAM KHẢO

[1] Applied Multivariate Statistical Analysis by Richard A Johnson

[2] C. Jordan, Essai sur la géométrie à n dimensions, Bull. Soc. Math. Fr., 2 (1875), pp. 103-174

[3] H. Hotelling, Relations between two sets of variates, Biometrika, 28 (1936), pp. 321-377

[4] T. Hastie, R. Tibshirani, M. Wainwright, Statistical Learning with Sparsity: the Lasso and Generalizations, CRC Press (2015)

[5] Yan Xu, Xinge Zhu, Jianping Shi, Guofeng Zhang, Hujun Bao, Hongsheng, Depth Completion from Sparse LiDAR Data with Depth-Normal Constraints, SenseTime Research, The Chinese University of Hong Kong, State Key Lab of CAD&CG, Zhejiang University (2019)

[6] Jonas Uhrig, Nick Schneider, Lukas Schneider, Uwe Franke, Thomas Brox, and Andreas Geiger. Sparsity invariant cnns. In 2017 International Conference on 3D Vision (3DV), pages 11–20. IEEE, 2017

[7] Bing Zhou, Matias Aiskovich, Sinem Guven, Sparse Depth Completion with Semantic Mesh Deformation Optimization, 2021

[8] Xu, Y.; Zhu, X.; Shi, J.; Zhang, G.; Bao, H.; and Li, H. 2019. Depth Completion from Sparse LiDAR Data with Depth-Normal Constraints. In Proceedings of the IEEE International Conference on Computer Vision, 2811–2820.

[9] Mal, F.; and Karaman, S. 2018. Sparse-to-dense: Depth prediction from sparse depth samples and a single image. In 2018 IEEE International Conference on Robotics and Automation (ICRA), 1–8. IEEE.

[10] Laina, I.; Rupprecht, C.; Belagiannis, V.; Tombari, F.; and Navab, N. 2016. Deeper depth prediction with fully convolutional residual networks. In 2016 Fourth international conference on 3D vision (3DV), 239–248. IEEE.

[11] Eigen, D.; Puhrsch, C.; and Fergus, R. 2014. Depth map prediction from a single image using a multi-scale deep network. In Advances in neural information processing systems, 2366–2374.

[12] Mousavian, A.; Pirsiavash, H.; and Koseck ˇ a, J. 2016. Joint se- ´ mantic segmentation and depth estimation with deep convolutional networks. In 2016 Fourth International Conference on 3D Vision (3DV), 611–619. IEEE.

[13] Shubham Tulsiani, Richard Tucker, and Noah Snavely. Layerstructured 3d scene inference via view synthesis. In Proc. European Conf. Computer Vision (ECCV), pages 302–317, 2018.

[14] Peter Hedman, Suhib Alsisan, Richard Szeliski, and Johannes Kopf. Casual 3D Photography. 36(6):234:1–234:15, 2017