MAKE FACE ID GREAT AGAIN

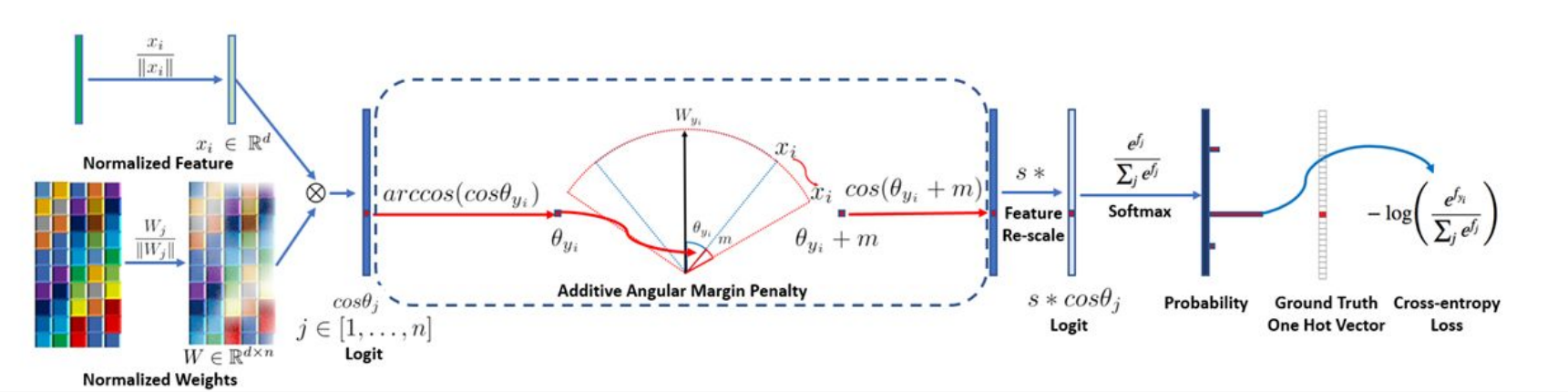
NGUYỄN CAO QUỐC - 240101022

Tóm tắt

- Lớp: CS2205.CH183
- Link Github của nhóm: https://github.com/imCaoQuoc/FaceID.git
- Link YouTube video:
- Ảnh + Họ và Tên: NGUYỄN CAO QUỐC

Giới thiệu

Face Identification là một bài toán quen thuộc trong lĩnh vực thị giác máy tính, có tác dụng nhận diện và phân loại khuôn mặt của một ai đó. Trong báo cáo này, em đề xuất một phương pháp mới giúp cải thiện khả năng trích xuất ra những vector đặc trưng hữu hiệu nhất để làm vector khuôn mặt đại diện cho một cá nhân cụ thể trong hệ thống face id. Đồng thời áp dụng thêm ArcFace Loss nhằm gia tăng khoảng cách giữa các lớp khác nhau và thu hẹp khoảng cách giữa các lớp giống nhau.



Hình 1: Minh họa quá trình áp dụng ArcFace Loss

Mục tiêu

- Cải thiện kết quả trích xuất đặc trưng của model thông qua kết hợp model CNN với ArcFace Loss.
- Tạo ra giải thuật có thể chọn được vector đặc trưng khuôn mặt tốt nhất của mỗi người dùng từ video đầu vào.
- Tối ưu hiệu suất của giải thuật trong thời gian thực, giúp tiết kiệm thời gian của người dùng.
- Đánh giá hiệu suất của giải thuật trên nhiều model khác nhau, bao gồm các model CNN cơ bản như MobileFace, ResNet tới các model cao cấp như ViT.

Nội dung và Phương pháp

Trong bài toán nhận diện khuôn mặt, việc tạo ra một hàm loss có thể phân biệt hiệu quả những đặc trưng khác nhau trên các khuôn mặt khác nhau là việc rất quan trọng. Một trong các hàm loss được sử dụng nhiều nhất là Softmax Loss.

$$L_{\text{softmax}} = -\log \frac{e^{W_{y_i}^{\top} x_i + b_{y_i}}}{\sum_{j=1}^{N} e^{W_j^{\top} x_i + b_j}}$$

Hình 2: Công thức tổng quát của Softmax Loss

Nội dung và Phương pháp

- ArcFace Loss là một cải tiến của Softmax Loss, được thiết kế để tăng cường khả năng phân biệt giữa các lớp trong bài toán Face Id.
- Với Softmax truyền thống, logits được tính dựa trên tích vô hướng giữa vector đặc trưng và vector trọng số, còn ArcFace áp dụng chuẩn hóa L2 cho cả hai vector trên, đưa chúng về một siêu mặt phẳng (hypersphere). Giúp rõ hơn sự phân chia giữa các lớp.
- Thay vì sử dụng cos(theta) để đo khoảng cách giữa 2 vector ở trên, ArcFace cộng thêm một biên góc m vào góc của lớp đúng. Cụ thể, nếu logit của Softmax là s.cos(theta) thì logit của ArcFace là s.cos(theta+m). Với s là hệ số khuếch đại các giá trị sau chuẩn hóa.

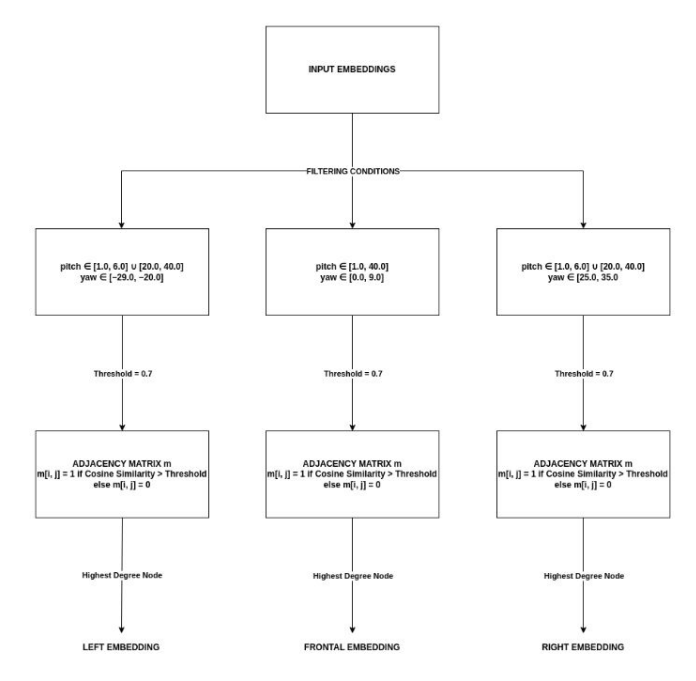
$$L_{\text{ArcFace}} = -\log \frac{e^{s\cdot\cos(\theta_{y_i}+m)}}{e^{s\cdot\cos(\theta_{y_i}+m)} + \sum_{j=1,j\neq y_i}^{N} e^{s\cdot\cos\theta_j}}$$

Hình 3: Công thức của ArcFace Loss

Nội dung và Phương pháp

- Đối với cách chọn vector đặc trưng, em đề xuất một giải thuật do em tạo ra có tên là Graph-Base Pose Filtering and Embedding Selection.
- Dựa vào các vector đặc trưng từ đầu ra của các model detection và các hướng khuôn mặt (pose), em sẽ tạo ra các điều kiện lọc theo ba pose chính: mặt có xu hướng quay trái, mặt có xu hướng quay phải, mặt không quay nhiều.
- Với mỗi nhóm pose trên, em tạo một ma trận liên kết m*m giữa các vector (m là số lượng vector thỏa điều kiện của từng nhóm).
- Sử dụng cosine similarity score để tính giá trị cho từng cặp vector tương ứng (loại bỏ hai vector trùng nhau do cosine similarity score của chúng luôn là 1). Với mỗi cặp vector có similarity score vượt ngưỡng threshold đặt trước, giá trị tại ô của chúng được gán bằng 1, ngược lại gán 0.
- Vector nào có nhiều liên kết với những vector còn lại nhất sẽ được sử dụng làm vector đặc trưng cho nhóm pose đó.

Nội dung và Phương pháp



Hình 4: Minh họa giải thuật chọn vector đặc trưng của khuôn mặt

Kết quả dự kiến

- Khả năng nhận dạng đối tượng hiệu quả: nhờ vào khả năng của ArcFace Loss, việc nhận diện được nhiều đối tượng khác nhau sẽ giúp tổng quan hệ thống ít sai sót hơn.
- Chọn đúng vector đặc trưng sẽ giúp cho mỗi đối tượng trong hệ thống ít bị nhận nhầm nhờ vào khả năng tổng quát hóa của vector đặc trưng được chọn
- Tối ưu hóa tài nguyên: giải thuật được tối ưu hóa để hoạt động hiệu quả với tài nguyên tính toán hạn chế, phù hợp trên nhiều thiết bị khác nhau.

Tài liệu tham khảo

[1] Jia Guo et al. Perspective Reconstruction of Human Faces by Joint Mesh and Landmark Regression. 2022. arXiv: 2208.07142 [cs.CV]. url: https://arxiv.org/abs/2208.07142.

[2] Jiankang Deng et al. “ArcFace: Additive Angular Margin Loss for Deep Face Recognition”. In: IEEE Transactions on Pattern Analysis and Machine Intelligence 44.10 (Oct. 2022), pp. 5962–5979. Issn: 1939-3539. doi: 10.1109/tpami.2021.3087709. Url: http://dx.doi.org/10.1109/TPAMI.2021.3087709.

[3] Sheng Chen et al. MobileFaceNets: Efficient CNNs for Accurate Real-Time Face Verification on Mobile Devices. 2018. arXiv: 1804 .07573 [cs.CV]. url: https ://arxiv.org/abs/1804.07573.

[4] Kaiming He et al. Deep Residual Learning for Image Recognition. 2015. arXiv:1512.03385 [cs.CV]. url: https://arxiv.org/abs/1512.03385.

[5] Alexey Dosovitskiy et al. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. 2021. arXiv: 2010.11929 [cs.CV]. url: https://arxiv.org/abs/2010.11929.

[6] Jiankang Deng et al. RetinaFace: Single-stage Dense Face Localisation in the Wild.2019. arXiv: 1905.00641 [cs.CV]. url: https://arxiv.org/abs/1905.00641.

[7] Zheng Zhu et al. WebFace260M: A Benchmark Unveiling the Power of Million-Scale Deep Face Recognition. 2021. arXiv: 2103.04098 [cs.CV]. url: https://arxiv.org/abs/2103.04098.

[8] Weiyang Liu et al. SphereFace: Deep Hypersphere Embedding for Face Recognition.2018. arXiv: 1704.08063 [cs.CV]. url: https://arxiv.org/abs/1704.08063.

[9] Hao Wang et al. CosFace: Large Margin Cosine Loss for Deep Face Recognition. 2018.arXiv: 1801.09414 [cs.CV]. url: https://arxiv.org/abs/1801.09414.

[10] Feng Wang et al. “NormFace: L 2 Hypersphere Embedding for Face Verification”. In: Proceedings of the 25th ACM international conference on Multimedia. MM ’17. ACM, Oct. 2017, pp. 1041–1049. doi: 10.1145/3123266.3123359. url: http://dx.doi.org/10.1145/3123266.3123359.