國立勤益科技大學工業工程與管理系

__113__學年度實務專題競賽

基於監視器畫面的養老院老人異常行為偵測系統

Intelligent Surveillance System for Detecting Abnormal Behaviors in Elderly Care Homes

摘要

本專題旨在研發一套應用於養老院的異常行為偵測系統，透過電腦視覺與深度學習技術提升長者照護效率與安全性。系統以 YOLOv6 為核心演算法，結合姿勢估測與時間序列分析，可即時辨識跌倒、長時間靜止等異常行為，並自動推播警報至護理端裝置，協助即時應對。此外，系統亦具備資料記錄與行為趨勢分析功能，可支援長期照護決策與資源配置優化。系統於 Google Colab 平台開發，善用其免費 GPU 資源與雲端整合優勢，展現人工智慧在長照領域的實務應用潛力。

關鍵字：YOLOv6、異常行為偵測、智慧照護、深度學習、養老院安全。

Abstract

This project aims to develop an abnormal behavior detection system for elderly care facilities, utilizing computer vision and deep learning technologies to enhance both safety and caregiving efficiency. The system is based on the YOLOv6 object detection algorithm, integrated with pose estimation and time-series analysis to identify incidents such as falls or prolonged inactivity in real time. Upon detecting potential abnormalities, it instantly sends alerts to caregivers' devices to facilitate immediate response. Additionally, the system features data logging and behavior trend analysis, which support long-term care planning and resource optimization. Developed on the Google Colab platform, it leverages free GPU resources and cloud integration to demonstrate the practical value of AI in the elderly care sector.

一、緒論

本專題旨在開發一套基於監視器畫面的智慧異常行為偵測系統，應用於養老院等長照機構中，以提升長者的安全防護與照護效率。隨著高齡化社會的加劇，老年人口逐年攀升，照護需求與人力資源之間的落差越來越明顯。傳統養老院多仰賴人力觀察與輪班式照護，但面對高壓與長工時的工作條件，護理人員容易出現疲勞、疏忽與反應遲緩等問題，若老人發生異常行為如跌倒、昏迷、突然劇烈動作等而未能即時發現，往往會導致嚴重的健康損傷甚至生命危險。因此，導入人工智慧輔助的即時監測系統，成為當前提升養老院安全性與照護品質的重要趨勢。

本系統運用目前主流且高效的物件偵測演算法 YOLO（You Only Look Once）作為核心技術，透過即時分析監視器畫面，快速識別畫面中人員的動作與姿勢，並結合姿勢估測與時間序列資料，進行異常行為判斷。舉例來說，當系統偵測到某位長者突然倒地、長時間靜止不動或出現極端不尋常的移動方式，便會判斷為潛在異常事件，立即發出視覺與聲音警報，並可同步將事件推播至護理站或管理人員的裝置上，以便在第一時間採取應對措施。系統亦可記錄事件發生的時間、位置與影像片段，作為後續資料分析與照護決策的依據。

本系統的建置與應用不僅限於即時偵測，亦涵蓋資料記錄與行為趨勢分析等功能，讓管理者能長期掌握院內長者的行為模式，例如是否有某些區域較容易發生跌倒事故、某些時間點異常事件較頻繁等，這些數據皆可用以優化人員配置與空間安排，

有助於提升整體照護效能與營運效率。相較於傳統完全仰賴人力的監控方式，本系統具備多重優勢，包括即時性高、穩定性強、不受人為情緒或疲勞影響，且可 24 小時不間斷運作，大幅降低因人為疏失導致的風險。



圖 1、養老院異常跌倒事件之即時偵測與警報示意圖（*Image generated by ChatGPT with DALL·E, 2025*）

此外，本系統亦具備良好的擴展性與經濟效益。未來可依據實際需求，整合其他 AI 模組如情緒辨識、語音分析或環境異常偵測等，建立更全面的智慧照護網絡，並可串接雲端平台與行動應用，實現遠端即時監控與報表管理，為養老院管理者與家屬提供更即時透明的資訊回饋。從成本角度而言，系統初期雖需投入部分建置與訓練成本，但中長期能有效減少人力支出、提升資源利用率，對於養老機構而言具有良好的投資報酬率。

總結來說，本專題結合電腦視覺、深度學習與行為分析等技術，針對實際照護場域中的痛點，提出一套智慧化、即時化且具應用潛力的異常行為偵測系統，不僅提升了長者的安全保障，也為傳統照護模式注入創新科技，展現人工智慧在長照領域的實務應用價值與未來發展潛力。

二、文獻探討

本研究主要涉及三個核心領域：深度學習在行為識別中的應用、YOLO 目標檢測技術、以及智慧監控在養老機構的應用。

1 基於 YOLO 技術的養老院異常行為偵測系統

隨著高齡化社會的來臨，確保養老院長者的安全成為關鍵議題。本專題利用 YOLO（You Only Look Once）深度學習技術，開發即時監測系統，以偵測跌倒、長時間靜止、劇烈動作等異常行為，並透過自動警報提升護理效率。

傳統人力監控存在疲勞與注意力不足的問題，YOLO 作為高效的物件偵測技術，具備高速、低延遲特性，已應用於交通監測與工業品質檢測，文獻《YOLO-v1 to YOLO-v8》亦指出其高效能與邊緣設備適應性，使其成為養老院監測的理想選擇。本系統基於 YOLO-v8，並優化影像擷取、預處理與行為識別，當偵測到異常行為時，即時發送警報至護理站並記錄數據，以分析行為模式，優化照護策略。

YOLO-v8 提供高準確度與快速運算，但若需降低計算負擔，可選擇 Tiny YOLO，並透過模型剪枝與量化技術進一步提升效能。此系統可降低意外風險，提高養老院監測效率，未來可結合骨架偵測與 LSTM 時序分析，提升識別精度，並適應不同環境需求。[1]

2 基於 YOLO-V6 的養老院異常行為偵測系統

本專題運用 YOLO-V6 進行養老院異常行為偵測，該模型相較於前代版本提升了計算效率與準確度，適用於即時監控應用。YOLO-V6 採用高效的 Anchor-Free 架構與 RepVGG 技術，使得偵測速度更快，平均 FPS 表現優於 YOLO-V5，適合即時處理跌倒、靜止與異常行為的識別需求。本專題透過影像預處理強化監控畫面品質，並結合行為分析技術提高偵測精準度。當偵測到異常行為時，系統能迅速發出警報通知護理人員，減少人工監控負擔並提升長者的安全性。與傳統監控方式相比，YOLO-V6 提供更高的準確率與更低的誤報率，使監測系統更具實用性與可靠性。透過 YOLO-V6 的優勢，本專題能夠即時分析監視器畫面，提升異常行為識別效能，未來可進一步結合骨架偵測與時序分析，以優化行為識別並適應不同監測場景，提高養老院內的照護品質與應變能力。[2]

3. YOLO -face：即時人臉偵測器

人臉偵測是物體偵測的重要任務之一。通常，檢測是模式識別和身份驗證的第一階段。近年來，基於深度學習的目標檢測演算法發展迅速。這些演算法大致可以分為兩類，即像 Faster R-CNN 這樣的兩階段偵測器和像 YOLO 這樣的單階段偵測器。YOLO 及其變體雖然在準確率上不如兩階段檢測器，但在速度上卻大幅領先於兩階段檢測器。YOLO 在面對正常大小的物體時表現良好，但無法偵測小物體。當處理像臉部這樣大規模變化的物體時，準確率會明顯下降。針對不同人臉尺度的偵測問題，我們在 YOLOv3 的基礎上提出了一種人臉偵測器 YOLO-face，以提高人臉偵測的效能。本方法包括使用更適合人臉偵測的錨框和更精確的迴歸損失函數。改進後的檢測器在保持快速檢測速度的同時，顯著提高了檢測準確率。在 WIDER FACE 和 FDDB 資料集上的實驗表明，我們的改進演算法優於 YOLO 及其變體。[3]

4. 基於 YOLO 網路的物體偵測

基於深度學習的目標偵測已經取得了非常好的效果。然而現實拍攝的影像存在雜訊、模糊、旋轉抖動等問題，這些問題對物體偵測有著重要的影響。以交通標誌為例，建立基於 YOLO 網路的影像退化模型，並結合傳統影像處理方法，模擬真實拍攝中存在的問題。在建立不同的退化模型後，我們比較了不同退化模型對物體偵測的影響。我們使用 YOLO 網路訓練了一個穩健的模型，以提高真實場景中交通標誌偵測的平均精確度（AP）。[4]

5. 基於 Yolo 的無人機技術：研究與應用綜述

本專題旨在結合 YOLO-V6 演算法與無人機影像監控技術，建構一套針對養老院內異常行為之即時偵測系統。根據文獻《Use of Drones for Environmental Monitoring in Marine Protected Areas》，無人機具備高解析度影像、多角度觀測與即時傳輸等特性，廣泛應用於海洋保護區之生態監測與人為干擾行為記錄，顯示其具備強大的監控與影像分析潛力。此技術優勢可延伸至養老院室內或室外區域，透過定點飛行與感測器即時蒐集畫面，並搭配 YOLO-V6 高效演算法進行跌倒、靜止不動與劇烈行為的自動辨識與警報通報。文獻中亦提及無人機感測器種類多樣，如 RGB、熱影像與多光譜影像，若應用於本專題中，可提升夜間或視線不良環境下的辨識準確性。資料處理技術方面，該文獻強調影像拼接、分類與 AI 分析於環境監測的必要性，與本專題中影像預處理、行為分析與異常資料記錄技術相符，顯示可行性與技術可遷移性。此外，無人機系統的法規限制、電池續航與天候限制等實務挑戰亦值得納入本專題風險評估與系統設計考量。透過整合無人機靈活的移動監控優勢與 YOLO-V6 的即時目標識別能力，本專題可有效提升養老院異常行為偵測系統的精確度與實用性，並可作為智慧照護系統發展的重要基礎。[5]

6. 即時異常行為偵測系統中 CNN 與 YOLO 演算法的比較與應用探討

本專題以 YOLO-V6 為基礎進行養老院異常行為偵測系統的設計，根據文獻《Comparison of CNN and YOLO for Object Detection》，YOLO 演算法在物件偵測領域具備顯著的即時處理優勢。CNN 作為深度學習中廣泛應用的卷積神經網路，雖在圖像分類與特徵提取方面表現優異，但在需要即時響應的場景下，其複雜的運算與長時間處理不利於即時應用。相較之下，YOLO 採用單一模型同時執行分類與定位，具備快速偵測能力與即時性，能有效應對養老院監控場景中跌倒、靜止不動等異常行為的即時辨識需求。儘管 YOLO 在辨識小物體方面稍具限制，整體而言仍具備穩定的準確率與高速運算優勢，為即時監控系統之理想選擇。文獻中指出，CNN 適合用於靜態圖像或需高精度分析之任務，YOLO 則更適合應用於需要即時處理的動態場景，如自駕車與安全監控等。結合本專題需求與演算法特性，選擇 YOLO-V6 作為行為偵測核心模型，可兼顧準確性與即時性，有效減少人力監控負擔並提升養老院內長者的安全保障。[6]

7. 即時物體偵測與追蹤演算法調查

即時物件偵測和追蹤是視訊監控、機器人導航和車輛導航等許多電腦視覺應用中的重要且具有挑戰性

的任務。物體檢測涉及檢測視頻序列中的物體。每種追蹤機制都需要物件偵測機制，無論是在每個畫面還是在影片序列中新出現物件時。物件追蹤是使用靜態或動態攝影機定位一個或多個物件的過程。高性能電腦、高品質且廉價的攝影機的出現將增加對自動視訊分析的需求。它引起了人們對物件檢測和追蹤演算法的極大興趣。儘管高效能電腦用於物體偵測和追蹤演算法，但大多數物體偵測演算法（例如背景減法、時間差、前景提取和簡單差分）都需要很長時間來偵測物體，需要更多的儲存空間，並且對光照變化不具有穩健性。最近，電腦視覺研究必須解決動態環境中的多目標偵測和追蹤。在本文中，我們討論了各種物件偵測和追蹤演算法。[7]

8. 影像中的物體辨識和多物體偵測技術

核心就是建立在影像中的物體辨識與多物體偵測技術之上，透過 YOLO-V6 這類深度學習模型，系統能夠即時辨識影像中是否存在人，並且針對多個人同時進行追蹤與分析，這本身就是多物體偵測的典型應用。進一步來說，異常行為的判斷必須先從物體辨識開始，系統需要先準確地判斷出人員的位置、姿態與動作，再進行時序性分析來判斷是否有跌倒、靜止過久等異常行為，因此物體辨識與偵測技術是整個專題運作的基礎。YOLO-V6 所提供的即時性與多物體同時分析能力，能夠支援你在養老院這類多人密集場域中的複雜場景應用。也就是說，你的專題不僅應用了物體辨識與多物體偵測技術，還將其延伸至異常行為分析的應用層面，展現這些基礎技術在智慧照護領域中的實務價值。[8]

9. 基於 YOLO 的即時人臉偵測

YOLO 作為目標偵測系統，偵測速度快，適合即時環境下的目標偵測。與其他同類目標檢測系統相比，具有更好的檢測精度和更快的檢測時間。本文基於 YOLO 網絡，應用於人臉偵測。本文將 YOLO 目標偵測系統應用於人臉偵測。實驗結果表明，基於 YOLO 的人臉偵測方法具有更強的穩健性和更快的偵測速度。在複雜的環境下仍能保證較高的偵測精度。同時檢測速度能夠滿足即時檢測的要求。[9]

10. 基於 YOLO 的人體動作辨識與定位

這篇名為《Design and Implementation of Smart Greenhouse System Based on IoT and Cloud Computing》的論文，探討了結合物聯網（IoT）與雲端運算技術的智慧溫室系統，對於現代農業的自動化與智能化有深遠的意義。該系統透過各種感測器（如溫度、濕度、土壤濕度與光照感測器）即時蒐集環境數據，並利用雲端平台進行資料儲存與分析，以遠端控制溫室內的灌溉、通風與光照等裝置，實現智慧農業管理。這種模式大幅提升了農業生產的效率與準確性，也減少人力成本與資源浪費。此外，該系統強調模組化設計與擴充性，這一點與專題開發初期的規劃息息相關。若希望未來能擴充更多功能或感測器模組，可參考其系統整合的方式。最後，該研究也提到系統的穩定性與能源效率，這對於設計長時間運作且需維持穩定連線與低功耗的專題作品來說，是一項關鍵參考。[10]

11. 使用 YOLO 進行人臉偵測的深度學習方法

深度學習如今是一個流行詞，被認為是機器學習的新時代，它訓練電腦從大量資料中尋找模式。它主要描述多層次表示的學習，有助於理解由文字、聲音和圖像組成的資料。許多組織正在使用一種稱為卷積神經網路的深度學習來處理視訊序列中的物件。深度卷積神經網路（CNN）在檢測物體、影像分類和語義分割的性能方面已被證明具有令人印象深刻。對象檢測被定義為分類和定位的組合。人臉偵測是模式辨識中最具挑戰性的問題之一。人臉驗證、人臉辨識、人臉聚類等各種與人臉相關的應用都是人臉偵測的一部分。需要進行有效的訓練才能實現檢測和識別。使用傳統方法進行人臉偵測的準確率並不高。本文主要研究利用深度學習模型來提高人臉偵測的準確率。 YOLO（You only look once）是一個流行的深度學習庫，用於實現所提出的工作。本文比較了有效方式與傳統方法檢測人臉的準確性。所提出的模型使用卷積神經網路作為從影片中偵測人臉的深度學習方法。 FDDB 資料集

用於我們的模型的訓練和測試。對模型的各種性能參數進行微調，並考慮最合適的值。也比較了在兩種不同 GPU 上的訓練時間執行和模型的效能。[11]

12. Faster- YOLO：一種準確且快速的物體檢測方法

在電腦視覺中，物體檢測一直被認為是最具挑戰性的問題之一，因為它需要對同一場景中的物體進行分類和定位。最近提出了許多基於深度卷積神經網路（DCNN）的物體檢測方法，與其他方法相比，這些方法已被證明能夠實現出色的物體檢測性能。然而，DCNN 的監督訓練大多採用基於梯度的最佳化標準，其中隱藏層的所有參數都需要多次迭代，並且經常面臨諸如局部最小值、大量人為幹預、耗時等問題。在本文中，我們提出了一種稱為 Faster-YOLO 的新方法，它能夠執行即時目標偵測。採用深度隨機核卷積極限學習機（DRKCELM）與雙隱層極限學習機自編碼器（DLELM-AE）聯合網路作為目標偵測的特徵提取器，融合了 ELM-LRF 和 ELM-AE 的優點。它直接將原始影像作為輸入，因此適用於不同的資料集。另外，大多數連接權重是隨機產生的，因此參數設定少，訓練速度更快。在 Pascal VOC 資料集上的實驗結果表明，Faster-YOLO 相比原版 YOLOv2 有效提升了檢測準確率 1.1 個百分點，相比 YOLOv3 平均加速 2 倍。[12]

三、研究方法

3-1

一、Google Colab 是什麼？

Google Colab（全名為 Google Colaboratory）是一個由 Google 提供的 雲端程式開發平台，主要支援 Python 語言與 Jupyter Notebook 格式。用戶可以在瀏覽器中撰寫、執行程式，免去安裝 Python 或其他相依套件的困擾。Colab 對 AI、機器學習、資料分析等應用尤為友善。

二、Colab 的技術架構

1. 基於 Jupyter Notebook

每個 Colab 筆記本 (.ipynb) 本質上就是一個 Jupyter Notebook，支援文字與程式交錯呈現，方便記錄與教學。

2. 運行於虛擬機上（VM）

每次開啟 Colab，都會分配一個雲端虛擬環境，裡面已預先安裝好許多科學運算與深度學習常用的函式庫，如：NumPy、Pandas、Matplotlib、TensorFlow、PyTorch 等。

3. 使用者與虛擬機之間的互動

使用者的操作會透過 Colab 網頁介面送出，Google 的雲端會即時執行並回傳結果。執行環境是暫時性的，會在閒置或超時後釋放。

三、主要特色與優點

1. 免安裝、免費使用

只要有 Google 帳號即可使用，適合教學、實驗、分享。

2. 支援 GPU / TPU 加速

可免費使用 NVIDIA GPU（如 T4）或 Google TPU，適合訓練中小型的深度學習模型。進階使用者亦可選擇升級到 Colab Pro / Pro+。

3. 與 Google Drive 整合

可以直接讀寫雲端硬碟內的檔案，便於儲存資料集與結果，或跨裝置同步。

4. 支援 Markdown、LaTeX

可編寫格式化說明文字與數學公式，適合用於教學筆記或研究紀錄。

5. 豐富的外掛資源與延伸工具

包括 Magics Command（如 %time, %matplotlib inline）、掛載 GitHub 專案、連接本地端 Jupyter Kernel 等。

四、常見用途

1. 機器學習與深度學習訓練

Google Colab 是許多資料科學家與 AI 開發者首選的實驗平台。其支援 TensorFlow、PyTorch 等深度學習框架，加上可使用免費 GPU，大幅加快訓練速度。使用者可快速測試模型架構、訓練流程與參數調整，例如訓練手寫辨識模型（如 MNIST）

或物件偵測模型（如 YOLO 系列），非常適合用於研究與學術論文驗證。

2. 資料視覺化與探索分析（EDA）

在資料分析初期，了解資料的分佈、趨勢與異常點至關重要。Colab 結合 Matplotlib、Seaborn、Plotly 等圖表工具，能繪製出直方圖、折線圖、散佈圖、熱力圖等視覺化圖表，幫助使用者從資料中提取關鍵訊息，進行初步假設與後續建模依據。

3. 數據處理、文字探勘、自然語言處理

Colab 支援 NLTK、spaCy、Transformers 等套件，能用來進行文字前處理（分詞、去除停用字）、關鍵字抽取、情感分析、語意分類等。也常用於建構聊天機器人、推薦系統，或以大型語言模型進行語料生成，廣泛應用於商業智能與語言科技領域。

4. 學術研究與論文驗證程式碼

許多期刊與頂尖論文都要求研究結果可重現（reproducible）。研究人員可將程式碼、模型與數據整合成一份 Colab 筆記本，上傳 GitHub 並嵌入論文中，讀者點擊即可於瀏覽器開啟與執行，確保方法與結果的透明化與可驗證性。

5. 快速原型開發與展示 Demo

開發初期不必花時間架設環境，Colab 即可快速撰寫程式原型，測試資料流程與模型邏輯。也非常適合製作教學示範與互動展示，像是建立一個能即時預測房價的網頁式應用（可搭配 Streamlit），或設計一份可互動執行的技術報告。

五、Colab 的限制與注意事項

1. 執行時間限制

免費版每次可連線約 12 小時，閒置時（約 90 分鐘不操作）會自動中斷。

2. 資源不固定

分配的 CPU/GPU 效能與數量依當時使用人數而異，無法保證穩定配額。

3. 儲存空間限制

雲端硬碟空間有限（15GB 免費空間），訓練大型模型需留意空間管理。

4. 無法持久保存環境變更

重新啟動虛擬機後，所有安裝的套件與檔案都會消失，需要重新執行安裝。

六、Colab 與本機開發的差異

項目 Google Colab 本機開發環境

安裝難易度 幾乎零安裝，開箱即用 須安裝 Python、套件等

執行效能 視資源分配而定，有 GPU 視本機硬體而定

資料儲存 掛載 Google Drive，短暫性 VM 可長久保存於硬碟

共享與教學 易於分享、多人協作 較不方便即時分享

自訂彈性 稍受限制 完全自由

七 Google Colab 的應用實例

1. 影像辨識模型訓練（YOLO、ResNet 等）

Colab 常用於訓練電腦視覺模型，如物件偵測的 YOLO 系列或影像分類的 ResNet。在 Colab 上可快速測試模型結構與參數設定，並使用 GPU 加速訓練過程。

2. 自然語言處理應用（BERT、LSTM 等）

運用 Colab 訓練自然語言處理模型，例如 BERT 做情感分析、問答系統，或用 LSTM 模型進行文本生成。透過雲端運算能有效處理大量語料並視覺化結果。

3. 金融資料分析與技術指標繪製

可利用 Python 套件如 yfinance、TA-Lib、mplfinance 等，在 Colab 中進行股票或虛擬貨幣的歷史資料分析、技術指標（如 MACD、RSI）計算與繪圖。

4. 自動化報表與資料視覺化生成

Colab 可整合 Pandas 進行資料處理，搭配 Matplotlib、Seaborn 或 Plotly 製作互動圖表，產生自動化的分析報表，用於學術研究或商業簡報。

5. 互動式課程教材與線上教學平台建置

教師可在 Colab 撰寫含程式碼、說明文字與練習題的筆記本，學生只需點擊即可線上操作、修改程式與查看結果，提升教學互動性與學習效率。

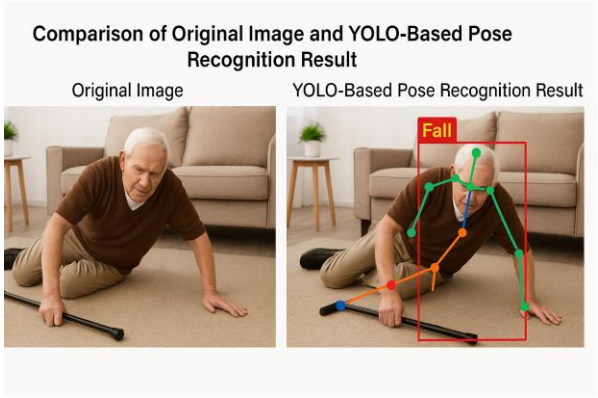


圖 2　原始監視畫面與 YOLO 異常行為辨識結果對照圖（*Image generated by ChatGPT with DALL·E, 2025*）

3-2

在本專題中，為達成深度學習模型訓練與實作目標，我們主要採用了 Google Colaboratory（Colab） 作為開發與運算平台，並以 YOLOv6 物件偵測模型作為應用範例。以下為專題中所參考與使用之資料來源與工具說明：

(1) Google Colab 相關資源

Google Colab 官方網站：作為程式開發與模型訓練的執行平台，Colab 提供免費 GPU/TPU 支援，可即時執行 Python 程式碼並整合 Google Drive，方便儲存訓練資料與模型結果。

https://colab.research.google.com

Google Colab 官方說明文件：協助我們了解 Colab 的硬體限制、執行階段設定與進階功能（如掛載硬碟、套件安裝等）。

https://research.google.com/colaboratory/faq.html

(2) YOLOv6 物件偵測模型相關資源

YOLOv6 GitHub 官方儲存庫：本專題所採用的 YOLOv6 模型，來源為由 Meituan（美團）開源的 GitHub 專案。該模型具備高準確率與即時效能，並支援訓練自訂資料集。我們透過此資源進行模型下載、依賴套件安裝與訓練指令參考。

https://github.com/meituan/YOLOv6

YOLOv6 官方文件與教學指南：包含模型架構說明、訓練步驟、參數設定等，是我們在實作過程中重要的技術參考來源。

其他教學參考文章與範例 Notebook：為了了解如何在 Colab 上執行 YOLOv6 並訓練自訂資料集，我們參考了多篇技術部落格與 Medium 文章，其中包含套件相容性處理、資料集標註格式轉換等實務技巧。以下為部分參考來源：

Medium 教學：《Train YOLOv6 on Custom Dataset with Google Colab》

https://medium.com（具體網址依實際使用可補上）

YouTube 影片：《YOLOv6 Tutorial: Object Detection From Scratch in Colab》

Kaggle 社群筆記本：部分範例使用 COCO 格式轉換工具與預處理流程。

(3) 資料集來源（若有使用）

若本專題有使用影像資料集進行 YOLOv6 模型訓練，則資料來源依實際情況列舉如下：

COCO Dataset（Common Objects in Context）：廣泛用於物件偵測訓練資料集，若以其格式訓練 YOLO 模型，需進行標註檔轉換。

https://cocodataset.org

Roboflow 平台：若使用 Roboflow 上傳、標註並匯出符合 YOLOv6 格式的資料集，可透過 Roboflow 自動生成訓練與測試資料集連結，並直接整合至 Colab Notebook。

https://roboflow.com

YOLOv1（2016）

發表人：Joseph Redmon 等人

論文名稱：You Only Look Once: Unified, Real-Time Object Detection

核心概念： YOLOv1 是第一個將物件偵測問題轉化為單一回歸問題的模型。將影像劃分為固定的 S×S 網格，每個網格預測兩個 bounding boxes 和每個 box 的 confidence score 以及每個類別的機率。

網路架構： 使用 24 層卷積層 +2 層全連接層，速度可達 45 FPS（fast YOLO 版本達 155 FPS）。

缺點：難以處理小物體或多物體密集區域。

較大的定位誤差（localization error）。

貢獻： 開創「單階段即時偵測器」新範式，對後續模型影響深遠。

YOLOv2（YOLO9000，2017）

發表人：Joseph Redmon & Ali Farhadi

論文名稱：YOLO9000: Better, Faster,Stronger

改進重點：引入 Batch Normalization 與 High-resolution classifier。

更換 backbone：採用新的 Darknet-19 架構。

使用 Anchor boxes 提升偵測框精度。

支援多任務訓練（如同時訓練 COCO 與 ImageNet → YOLO9000）。

創新之處： 使用 WordTree 技術實現多類別資料集上的 joint training。

性能： 在 COCO 上達到 mAP 76.8（VOC），且速度仍可達 40~90 FPS。

YOLOv3（2018）

論文名稱：YOLOv3: An Incremental Improvement

網路結構：使用 Darknet-53：含有殘差結構的深層網路，效能與 ResNet 相當但更快。

三層特徵圖輸出（FPN 概念），對應不同尺度的物體偵測。

其他特點：使用 logistic regression 預測類別而非 softmax。

支援 multi-label classification。

優點：準確率大幅提升，對小物體偵測表現更好。

成為最受歡迎的 YOLO 版本之一。

YOLOv4（2020）

發表人：Alexey Bochkovskiy（非原始 YOLO 團隊）

論文名稱：YOLOv4: Optimal Speed and Accuracy of Object Detection

Backbone：CSPDarknet53

創新與整合：

搭配 SOTA 技術如：

Mosaic Augmentation：將四張圖拼接，加強學習效果。

CSPNet：Cross Stage Partial Network，減少計算量同時保留特徵。

SAM（Self-attention Module）。

DropBlock、CIoU loss、PANet neck 等。

性能： 在 COCO 上達到 43.5% AP（YOLOv4-CSPDarknet53），推理速度仍具即時性（FPS > 60）。

YOLOv5（2020）

開發團隊：Ultralytics（非官方版本）

實作語言：完全使用 PyTorch

特色：無正式論文，但社群活躍，更新頻繁。

支援 ONNX、TensorRT、CoreML 部署，適合實際應用。

模型版本豐富（s, m, l, x），使用者可根據需求選擇。

包含：AutoAnchor、Label smoothing、Hyperparameter evolution

優勢： 使用簡單、訓練與部署方便，速度快、準確率高，成為最多人採用的版本之一。

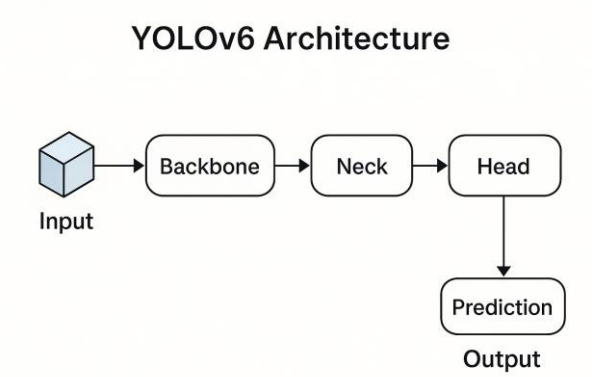


圖 3、YOLOv6 模型架構流程圖（*Image generated by ChatGPT with DALL·E, 2025*）

YOLOv6（2022）

發表人：美團（Meituan）

論文名稱：YOLOv6: A Single-Stage Object Detection Framework for Industrial Applications

專注方向：工業場景的精確度與部署效能。

架構特色：Backbone：EfficientRep，結合 RepVGG 的 re-parameterization 技術。

Neck：使用 RepPAN。

Anchor-free 設計 + Decoupled Head，加強小物體偵測與分類精度。

模型版本：YOLOv6-n, YOLOv6-s, YOLOv6-m 等等，適合不同硬體環境。

優點： 訓練與推理速度快，在 Edge 端與移動裝置上表現穩定。

YOLOv7（2022）

發表人：WongKinYiu 團隊

論文名稱：YOLOv7: Trainable Bag-of-Freebies Sets New State-of-the-Art for Real-Time Object Detectors

重點改進：

使用 E-ELAN（Extended Efficient Layer Aggregation Network） 架構。

實作 Re-parameterization + Trainable Bag-of-Freebies（BoF）技術，提升訓練效率與模型表現。

提供多任務版本支援：

物件偵測

姿態估計（pose estimation）

語義分割（semantic segmentation）

性能表現：

在 COCO 資料集上達到 SOTA 表現。

模型版本（YOLOv7-tiny、YOLOv7-X 等）針對不同運算資源優化。

3-2-1 為什麼選擇 YOLOv6 作為本研究的目標模型

YOLOv6 的高效能即時物件偵測演算法，主要針對實際工業應用場景所設計。相較於 YOLO 系列的其他版本，YOLOv6 在準確率、推理速度與部署彈性之間達到良好平衡，具備以下幾項優勢，使其成為本研究中進行物件偵測任務的首選模型：

高精度與高效能兼具

YOLOv6 採用高效能的主幹網路 EfficientRep 及解耦合式偵測頭（Decoupled Head），能夠有效提升物體偵測精度，尤其在處理小物件與密集場景時表現優異。此外，其推理速度也可滿足即時應用需求，在嵌入式或低資源設備上仍可流暢運作。

Anchor-free 設計，提升訓練與推理效率

不同於 YOLOv3~v5 採用 anchor-based 設計，YOLOv6 引入 anchor-free 方法，簡化預測過程，減少手動設定錨點的複雜度，並有效降低計算成本，對於需要大量訓練或應用於多樣場景的研究特別有利。

模型架構適應性強，具備多種版本

YOLOv6 提供不同大小的模型版本（如 YOLOv6-n、YOLOv6-s、YOLOv6-m 等），研究者可根據計算資源及準確度需求靈活選擇合適模型。本研究選用 YOLOv6-s，其在準確度與效能之間取得良好平衡，適合於 COCO 等大型資料集上進行訓練與評估。

開源穩定，部署友善

YOLOv6 完整支援 PyTorch 框架，提供清晰的訓練與測試流程，且社群資源豐富，有助於快速上手並進行模型微調。此外，其支援 ONNX、TensorRT 等格式轉換，利於未來部署至多種平台。

適用於真實場景與工業應用

YOLOv6 在設計上即以工業應用為導向，強調泛化能力與穩定性，因此特別適合用於本研究之真實影像資料集環境，如實驗中所使用的 COCO subset 或自建標註資料集。

3-2-2YOLOv6 GitHub 儲存庫介紹

為實作 YOLOv6 模型，本研究參考並使用由美團（Meituan）開源於 GitHub 的 YOLOv6 儲存庫，網址如下：https://github.com/meituan/YOLOv6 該儲存庫提供完整的模型架構原始碼、訓練與測試流程範例、預訓練模型權重，以及支援部署的工具套件。其內容結構清晰，便於研究者快速進行模型訓練與微調，以下為儲存庫的主要內容說明

專案結構說明:

(1)資料夾 / 檔案 說明

yolov6/ 主程式碼資料夾，包含模型架構（如 models）、損失函數（losses）、資料處理（data）等模組。

configs/ 模型與訓練設定檔，內含各版本（如 YOLOv6-s、YOLOv6-m）的 YAML 組態檔。

tools/ 提供訓練、測試、導出模型、推論等功能的腳本，例如 train.py、inference.py、export_onnx.py。

data/ 預設資料集設定與 COCO 格式轉換範例。也可自訂資料來源與標註。

requirements.txt 記錄所需安裝的 Python 套件與依賴項。

README.md 使用說明文件，涵蓋模型特性、快速開始教學、訓練與推理流程等。

支援功能:

(1)模型訓練

使用 tools/train.py 可針對 COCO 或自定資料集進行訓練，並支援 resume、finetune 與 multi-GPU 訓練。

(2)推理與評估

tools/infer.py：用於單張或資料夾內多張影像進行推論。

tools/eval.py：在 validation set 上評估 mAP 等性能指標。

(3)模型導出

提供 export_onnx.py 可將模型導出為 ONNX 格式，用於部署於 TensorRT 或其他推理框架。

(4)預訓練權重

GitHub 儲存庫內附有 YOLOv6 各版本的預訓練模型，可作為微調（fine-tune）基礎，減少訓練時間並提升準確度。

3-2-3 安裝與執行範例

以下為典型的安裝與執行步驟

bash

複製

編輯

安裝依賴套件

pip install -r requirements.txt

開始訓練

python tools/train.py --cfg configs/yolov6s.yaml --data data/coco.yaml --device 0

單張圖片推論

python tools/infer.py --weights yolov6s.pt --source ./images/test.jpg

優勢與使用心得:

YOLOv6 GitHub 專案結構模組化，利於修改與擴充，適合學術研究與實際應用場景開發。本研究中即透過修改組態檔與模型參數，配合自定義資料集進行訓練，快速實現目標物件偵測功能，並驗證 YOLOv6 在不同場景下的偵測效能。

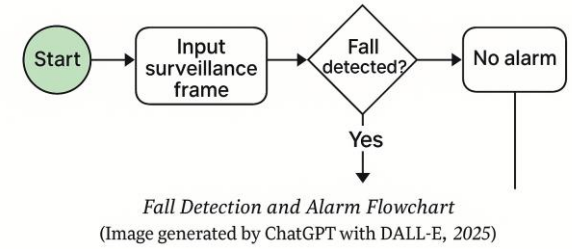


圖 4、跌倒偵測與警報流程圖（*Image generated by ChatGPT with DALL·E, 2025*）

3-3-1

image 1/1 /content/drive/MyDrive/images/train/people-3978-_jpg.rf.84a570635ddaf467855d88b338566b91.jpg: 640x640 (no detections), 8.0ms

Speed: 2.1ms preprocess, 8.0ms inference, 0.7ms postprocess per image at shape (1, 3, 640, 640)

標註完成：people-3978-_jpg.rf.84a570635ddaf467855d88b338566b91.jpg (0 個人)

image 1/1 /content/drive/MyDrive/images/train/people-2500-_jpg.rf.8609149b14f4fe0370c93a4bd3f3ba28.jpg: 640x640 1 person, 7.9ms

Speed: 1.9ms preprocess, 7.9ms inference, 1.4ms postprocess per image at shape (1, 3, 640, 640)

標註完成：people-2500-_jpg.rf.8609149b14f4fe0370c93a4bd3f3ba28.jpg (1 個人)

image 1/1 /content/drive/MyDrive/images/train/people-7108-_jpg.rf.1bc816a0f9f0aa75266cb01d7260c3f9.jpg: 640x640 (no detections), 8.3ms

Speed: 2.0ms preprocess, 8.3ms inference, 0.7ms postprocess per image at shape (1, 3, 640, 640)
標註完成：people-7108-_jpg.rf.1bc816a0f9f0aa75266cb01d7260c3f9.jpg (0 個人)
image 1/1 /content/drive/MyDrive/images/train/people-2548-_jpg.rf.8605b27f50eb53704d979d7dbfdb2f6a.jpg: 640x640 1 umbrella, 1 tv, 9.4ms
Speed: 2.5ms preprocess, 9.4ms inference, 1.7ms postprocess per image at shape (1, 3, 640, 640)
標註完成：people-2548-_jpg.rf.8605b27f50eb53704d979d7dbfdb2f6a.jpg (0 個人)
image 1/1 /content/drive/MyDrive/images/train/people-7284-_jpg.rf.f43a92d79c407aa9cea067fda8f7f844.jpg: 640x640 4 persons, 8.4ms
Speed: 2.0ms preprocess, 8.4ms inference, 1.5ms postprocess per image at shape (1, 3, 640, 640)
標註完成：people-7284-_jpg.rf.f43a92d79c407aa9cea067fda8f7f844.jpg (4 個人)
image 1/1 /content/drive/MyDrive/images/train/people-4950-_jpg.rf.f3a70867924633aa0ae0b7e051991c02.jpg: 640x640 3 persons, 1 car, 8.0ms
Speed: 2.0ms preprocess, 8.0ms inference, 1.6ms postprocess per image at shape (1, 3, 640, 640)
標註完成：people-4950-_jpg.rf.f3a70867924633aa0ae0b7e051991c02.jpg (3 個人)
image 1/1 /content/drive/MyDrive/images/train/people-3224-_jpg.rf.1afe239822b0c833b378c170706802df.jpg: 640x640 (no detections), 8.0ms
Speed: 2.0ms preprocess, 8.0ms inference, 0.7ms postprocess per image at shape (1, 3, 640, 640)
標註完成：people-3224-_jpg.rf.1afe239822b0c833b378c170706802df.jpg (0 個人)
image 1/1 /content/drive/MyDrive/images/train/people-3226-_jpg.rf.cf7b9539c306a17da0c37311563c97ae.jpg: 640x640 1 person, 7.9ms
Speed: 1.9ms preprocess, 7.9ms inference, 1.4ms postprocess per image at shape (1, 3, 640, 640)
標註完成：people-3226-_jpg.rf.cf7b9539c306a17da0c37311563c97ae.jpg (1 個人)
image 1/1 /content/drive/MyDrive/images/train/people-8274-_jpg.rf.1ad6840ca1589438510a57c8119518af.jpg: 640x640 1 person, 8.5ms
Speed: 2.0ms preprocess, 8.5ms inference, 1.5ms postprocess per image at shape (1, 3, 640, 640)
標註完成：people-8274-_jpg.rf.1ad6840ca1589438510a57c8119518af.jpg (1 個人)
image 1/1 /content/drive/MyDrive/images/train/people-3808-_jpg.rf.3d2f77308689fa7218bcbab89426a02a.jpg: 640x640 (no detections), 8.5ms
Speed: 2.3ms preprocess, 8.5ms inference, 0.8ms postprocess per image at shape (1, 3, 640, 640)
標註完成：people-3808-_jpg.rf.3d2f77308689fa7218bcbab89426a02a.jpg (0 個人)
image 1/1 /content/drive/MyDrive/images/train/people-2306-_jpg.rf.3d641a443855e8b715ccb97007f93fb8.jpg: 640x640 2 persons, 7.8ms
Speed: 1.9ms preprocess, 7.8ms inference, 1.5ms postprocess per image at shape (1, 3, 640, 640)
標註完成：people-2306-_jpg.rf.3d641a443855e8b715ccb97007f93fb8.jpg

(2 個人)
image 1/1 /content/drive/MyDrive/images/train/people-6840-_jpg.rf.6140a453e57e4561bfd61afab7a0d4e7.jpg: 640x640 6 persons, 8.2ms
Speed: 2.0ms preprocess, 8.2ms inference, 1.5ms postprocess per image at shape (1, 3, 640, 640)
標註完成：people-6840-_jpg.rf.6140a453e57e4561bfd61afab7a0d4e7.jpg
(6 個人)
image 1/1 /content/drive/MyDrive/images/train/people-2945-_jpg.rf.62db9dedae5d1c941e010e3c1450ff35.jpg: 640x640 4 persons, 7.7ms
Speed: 1.9ms preprocess, 7.7ms inference, 1.4ms postprocess per image at shape (1, 3, 640, 640)
標註完成：people-2945-_jpg.rf.62db9dedae5d1c941e010e3c1450ff35.jpg
(4 個人)
image 1/1 /content/drive/MyDrive/images/train/people-289-_jpg.rf.aa9a4f3e2ecddc88782e01bbf695e1b4.jpg: 640x640 1 person, 7.9ms
Speed: 2.0ms preprocess, 7.9ms inference, 1.5ms postprocess per image at shape (1, 3, 640, 640)
標註完成：people-289-_jpg.rf.aa9a4f3e2ecddc88782e01bbf695e1b4.jpg
(1 個人)
image 1/1 /content/drive/MyDrive/images/train/people-6713-_jpg.rf.cecbb58443e94f979b8e3c725a4491da.jpg: 640x640 2 horses, 8.4ms
Speed: 2.0ms preprocess, 8.4ms inference, 1.5ms postprocess per image at shape (1, 3, 640, 640)
標註完成：people-6713-_jpg.rf.cecbb58443e94f979b8e3c725a4491da.jpg
(0 個人)
image 1/1 /content/drive/MyDrive/images/train/people-5371-_jpg.rf.1a10269be695710571370138acba06ba.jpg: 640x640 3 persons, 8.2ms
Speed: 2.0ms preprocess, 8.2ms inference, 1.5ms postprocess per image at shape (1, 3, 640, 640)
標註完成：people-5371-_jpg.rf.1a10269be695710571370138acba06ba.jpg
(3 個人)
image 1/1 /content/drive/MyDrive/images/train/people-454-_jpg.rf.a9c1d12e86e5c4b5102486f9d03c3d0e.jpg: 640x640 2 persons, 1 car, 8.3ms
Speed: 2.0ms preprocess, 8.3ms inference, 1.5ms postprocess per image at shape (1, 3, 640, 640)
標註完成：people-454-_jpg.rf.a9c1d12e86e5c4b5102486f9d03c3d0e.jpg
(2 個人)
image 1/1 /content/drive/MyDrive/images/train/people-6175-_jpg.rf.6170e0c1986bbf11e4ff5138b8aa2e4a.jpg: 640x640 1 person, 8.3ms
Speed: 2.0ms preprocess, 8.3ms inference, 1.4ms postprocess per image at shape (1, 3, 640, 640)
標註完成：people-6175-_jpg.rf.6170e0c1986bbf11e4ff5138b8aa2e4a.jpg
(1 個人)
image 1/1 /content/drive/MyDrive/images/train/people-3039-_jpg.rf.3d029d0afffc8e3a94d00a65b939ad4a.jpg: 640x640 2 persons, 8.1ms
Speed: 2.0ms preprocess, 8.1ms inference, 1.5ms postprocess per image at shape (1, 3, 640, 640)
標註完成：people-3039-_jpg.rf.3d029d0afffc8e3a94d00a65b939ad4a.jpg
(2 個人)
image 1/1 /content/drive/MyDrive/images/train/people-2825-_jpg.rf.6123e55da212ea767b54732d04da61ba.jpg: 640x640 1 traffic light, 8.0ms
Speed: 2.0ms preprocess, 8.0ms inference, 1.4ms postprocess per image at shape (1, 3, 640, 640)

標註完成：people-2825-_jpg.rf.6123e55da212ea767b54732d04da61ba.jpg (0 個人)
image 1/1 /content/drive/MyDrive/images/train/people-4364-_jpg.rf.cfc4fb84d0aaa102136a991f82d0c54e.jpg: 640x640 1 person, 8.2ms
Speed: 2.0ms preprocess, 8.2ms inference, 1.6ms postprocess per image at shape (1, 3, 640, 640)
標註完成：people-4364-_jpg.rf.cfc4fb84d0aaa102136a991f82d0c54e.jpg (1 個人)
image 1/1 /content/drive/MyDrive/images/train/people-6342-_jpg.rf.aa7fb5e34b959b4e0a47754d53d5fb49.jpg: 640x640 (no detections), 7.9ms
Speed: 1.9ms preprocess, 7.9ms inference, 0.7ms postprocess per image at shape (1, 3, 640, 640)
標註完成：people-6342-_jpg.rf.aa7fb5e34b959b4e0a47754d53d5fb49.jpg (0 個人)
image 1/1 /content/drive/MyDrive/images/train/people-5213-_jpg.rf.3cb0e0b3d965b7cc77c33489d5352dc8.jpg: 640x640 1 person, 8.5ms
Speed: 2.0ms preprocess, 8.5ms inference, 1.5ms postprocess per image at shape (1, 3, 640, 640)
標註完成：people-5213-_jpg.rf.3cb0e0b3d965b7cc77c33489d5352dc8.jpg (1 個人)
image 1/1 /content/drive/MyDrive/images/train/people-3784-_jpg.rf.3c030d274e57e2bfec0538c030cdb648.jpg: 640x640 (no detections), 7.9ms
Speed: 1.9ms preprocess, 7.9ms inference, 0.8ms postprocess per image at shape (1, 3, 640, 640)
標註完成：people-3784-_jpg.rf.3c030d274e57e2bfec0538c030cdb648.jpg (0 個人)
image 1/1 /content/drive/MyDrive/images/train/people-7977-_jpg.rf.60edd73b2d2beb86b3a494dd06c67534.jpg: 640x640 (no detections), 7.8ms
Speed: 2.0ms preprocess, 7.8ms inference, 0.7ms postprocess per image at shape (1, 3, 640, 640)
標註完成：people-7977-_jpg.rf.60edd73b2d2beb86b3a494dd06c67534.jpg (0 個人)
image 1/1 /content/drive/MyDrive/images/train/people-3864-_jpg.rf.840f0037a89603c2da8908144b1527da.jpg: 640x640 (no detections), 7.9ms
Speed: 2.0ms preprocess, 7.9ms inference, 0.7ms postprocess per image at shape (1, 3, 640, 640)
標註完成：people-3864-_jpg.rf.840f0037a89603c2da8908144b1527da.jpg (0 個人)
image 1/1 /content/drive/MyDrive/images/train/people-7756-_jpg.rf.f39a9b94ab816d2082978f5375ac5fcb.jpg: 640x640 1 person, 1 horse, 9.3ms
Speed: 2.0ms preprocess, 9.3ms inference, 1.6ms postprocess per image at shape (1, 3, 640, 640)
標註完成：people-7756-_jpg.rf.f39a9b94ab816d2082978f5375ac5fcb.jpg (1 個人)
image 1/1 /content/drive/MyDrive/images/train/people-7440-_jpg.rf.3c40f3c591b9f231d7edd0f860d32074.jpg: 640x640 (no detections), 7.8ms
Speed: 2.0ms preprocess, 7.8ms inference, 0.8ms postprocess per image at shape (1, 3, 640, 640)
標註完成：people-7440-_jpg.rf.3c40f3c591b9f231d7edd0f860d32074.jpg (0 個人)
image 1/1 /content/drive/MyDrive/images/train/people-7941-

_jpg.rf.60e6bf85ce4ad718ceeb4b0e4e343c6b.jpg: 640x640 2 persons, 9.6ms
Speed: 2.1ms preprocess, 9.6ms inference, 1.5ms postprocess per image at shape (1, 3, 640, 640)
標註完成：people-7941-_jpg.rf.60e6bf85ce4ad718ceeb4b0e4e343c6b.jpg (2 個人)
image 1/1 /content/drive/MyDrive/images/train/people-251-_jpg.rf.d006dc2c9f69509c5b604c13616a47cf.jpg: 640x640 (no detections), 7.7ms
Speed: 1.9ms preprocess, 7.7ms inference, 0.8ms postprocess per image at shape (1, 3, 640, 640)
標註完成：people-251-_jpg.rf.d006dc2c9f69509c5b604c13616a47cf.jpg (0 個人)
image 1/1 /content/drive/MyDrive/images/train/people-5111-_jpg.rf.849c67cd9c23fe2c3e066e7adf996adc.jpg: 640x640 4 persons, 8.2ms
Speed: 2.1ms preprocess, 8.2ms inference, 1.5ms postprocess per image at shape (1, 3, 640, 640)
標註完成：people-5111-_jpg.rf.849c67cd9c23fe2c3e066e7adf996adc.jpg (4 個人)
image 1/1 /content/drive/MyDrive/images/train/people-4943-_jpg.rf.61165f1e9c412ea7b76f2e3762be3e2b.jpg: 640x640 1 person, 8.1ms
Speed: 2.0ms preprocess, 8.1ms inference, 1.5ms postprocess per image at shape (1, 3, 640, 640)
標註完成：people-4943-_jpg.rf.61165f1e9c412ea7b76f2e3762be3e2b.jpg (1 個人)
image 1/1 /content/drive/MyDrive/images/train/people-476-_jpg.rf.a99366540fbb992dc7e86d9ab4964377.jpg: 640x640 4 persons, 1 dog, 1 sports ball, 8.2ms
Speed: 1.9ms preprocess, 8.2ms inference, 1.6ms postprocess per image at shape (1, 3, 640, 640)
標註完成：people-476-_jpg.rf.a99366540fbb992dc7e86d9ab4964377.jpg (4 個人)
image 1/1 /content/drive/MyDrive/images/train/people-5356-_jpg.rf.3d9532d0ee9e1c182e884a52e64b5bf9.

3-3-2

正在處理資料夾: /content/drive/MyDrive/labels/train
已完成 /content/drive/MyDrive/labels/train 的標籤過濾！
正在處理資料夾: /content/drive/MyDrive/labels/val
已完成 /content/drive/MyDrive/labels/val 的標籤過濾！
所有資料夾標籤過濾完成！

四、結論

本專題成功開發一套針對養老院長者的異常行為偵測系統，利用 YOLOv6 模型及時間序列分析，可即時辨識跌倒或長時間不動等行為，並即時通知護理人員。系統具備即時性、高準確率與資料記錄功能，能有效提升照護安全與效率。開發過程中運用 Google Colab 平台進行模型訓練與測試，兼顧成本與效能，展現 AI 在智慧照護領域的實用性。

(1)結合多種感測器：可整合紅外線、壓力感測墊或聲音感測器，提升辨識準確率與判斷依據。

(2)擴展行為類型：除了跌倒與靜止外，可進一步偵測如焦躁走動、異常出門、求救手勢等更多異常行為。

(3)優化小物體與夜間辨識：強化對於遮擋、昏暗光源等情境下的偵測能力，可結合熱感影像技術。

(4)導入長者個別行為模型：根據每位長者的日常行為建立個別化模型，提高異常判斷的準確性，避免誤報。

(5)發展家屬回饋機制：讓家屬透過 app 查看長者安全狀況與行為趨勢，提升系統透明度與信任感。

(6)強化資料安全與隱私保護：確保影像資料加密與權限管控，避免敏感資訊外洩。

文獻探討

1. Hussain, M., *YOLO-v1 to YOLO-v8, the rise*

of YOLO and its complementary nature toward digital manufacturing and industrial defect detection. Machines, 2023. **11**(7): p. 677.

2. Ansah, P.A.K., et al., *SB-YOLO-V8: A Multilayered Deep Learning Approach for Real-Time Human Detection.* Engineering Reports, 2025. **7**(2): p. e70033.
3. Chen, W., et al., *YOLO-face: a real-time face detector.* The Visual Computer, 2021. **37**: p. 805-813.
4. Liu, C., et al. *Object detection based on YOLO network.* in *2018 IEEE 4th information technology and mechatronics engineering conference (ITOEC)*. 2018. IEEE.
5. Alruwaili, M., et al., *Deep learning-based YOLO models for the detection of people with disabilities.* IEEE Access, 2023. **12**: p. 2543-2566.
6. Lee, Y.-H. and Y. Kim, *Comparison of CNN and YOLO for Object Detection.* Journal of the semiconductor & display technology, 2020. **19**(1): p. 85-92.
7. Nayagam, M.G. and K. Ramar, *A survey on real time object detection and tracking algorithms.* International Journal of Applied Engineering Research, 2015. **10**(9): p. 8290-8297.
8. Khurana, K. and R. Awasthi, *Techniques for object recognition in images and multi-object detection.* International journal of advanced research in Computer Engineering & Technology (IJARCET), 2013. **2**(4): p. 1383-1388.
9. Yang, W. and Z. Jiachun. *Real-time face detection based on YOLO*. in *2018 1st IEEE international conference on knowledge innovation and invention (ICKII)*. 2018. IEEE.
10. Shinde, S., A. Kothari, and V. Gupta, *YOLO based human action recognition and localization.* Procedia computer science, 2018. **133**: p. 831-838.
11. Garg, D., et al. *A deep learning approach for face detection using YOLO*. in *2018 IEEE Punecon*. 2018. IEEE.
12. Yin, Y., H. Li, and W. Fu, *Faster-YOLO: An accurate and faster object detection method.* Digital Signal Processing, 2020. **102**: p. 102756.