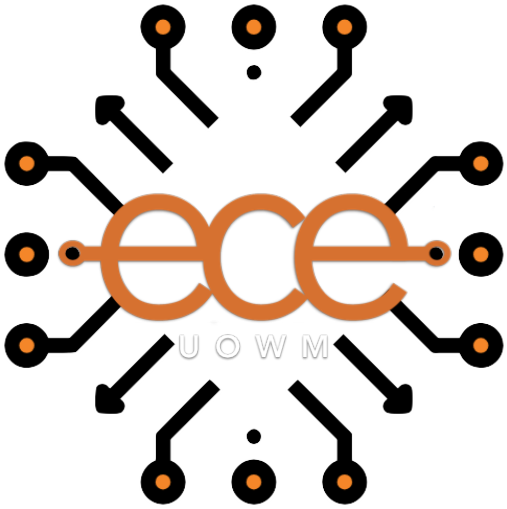

University of Western Macedonia
Department of Electrical & Computer Engineering

Acceleration and Optimization for Real-Time Multimodal Processing on the Edge

A thesis submitted for the degree of
Doctor of Philosophy

Dimitrios Tsiktsiris
Supervisor: Dr. Minas Dasygenis

Robotics, Embedded & Integrated Systems Laboratory
June 25, 2024

Abstract

Edge processing has emerged as a critical domain in computer vision, driven by the increasing amounts of data sources, over the traditional cloud computing architectures. Cloud computing has significant disadvantages in addressing latency, bandwidth, and security concerns, especially on real time applications. Edge computing addresses these issues by enabling local data processing, therefore reducing latency, bandwidth usage, and preserving privacy. This paradigm shift empowers advanced computer vision applications such as real-time video analytics, facial recognition, and augmented reality (AR), driving innovation and opening new possibilities.

Computer vision aims to bridge the gap between human perception and machine understanding, yet it faces significant challenges, particularly in handling large volumes of visual data. Deep learning (DL) approaches such as the Convolutional Neural Networks (CNNs), have been crucial in modeling pixel relations and recognizing complex patterns. However, the integration of multiple modalities by fusing data from multiple sources is essential for improving accuracy and robustness. This integration poses significant challenges and requires a sophisticated handling and alignment of the diverse data types, especially under the resource constraints of edge devices.

This dissertation aims to address these challenges by proposing a novel design methodology for edge processing, leading to an accelerated multimodal framework tailored for these environments. This framework enables the execution of complex and in-depth data processing directly at the source, leveraging novel artificial intelligence (AI) models and optimizations for various applications, such as abnormal event detection, object recognition, proximity assessment, and facial recognition. These approaches demonstrate significant improvements in decision-making capabilities, response times, and overall system performance, even with the limited resources of embedded systems.

To mitigate the computational overhead of data preprocessing, we have implemented various computational processing designs in hardware, taking ad-

vantage of the parallelism of Field-Programmable Gate Arrays (FPGAs). We introduce a portable VHSIC Hardware Description Language (VHDL) design for color transformation and Sobel edge detection, further improved using High-Level Synthesis (HLS) for increased efficiency. We also present an accelerated noise reduction technique based on image stacking, highlighting the potential of hardware acceleration for edge processing. Additionally, our research investigates the fusion of multimodal sensor data, combining information from various sources such as video, audio, and other sensors. This approach provides a more comprehensive understanding of the environment, enabling even more informed decision-making and improved system performance in complex, real-world scenarios.

The effectiveness of this methodology is demonstrated through an extensive evaluation on real-world datasets and deployment scenarios in public transportation, emphasizing passenger safety and real-time decision-making. The results highlight the framework's versatility and robustness, showcasing its potential to transform diverse applications through the power of edge computing and multimodal data processing.

Περίληψη

Η επεξεργασία στο άκρο (edge processing) έχει αναδειχθεί ως ένας κρίσιμος τομέας στην υπολογιστική όραση, οδηγούμενη από τις αυξανόμενες ποσότητες δεδομένων από συσκευές Internet of Things (IoT), έξυπνες κάμερες και αυτόνομα συστήματα. Οι παραδοσιακές αρχιτεκτονικές υπολογισμού παρουσιάζουν σημαντικά μειονεκτήματα στην αντιμετώπιση των ζητημάτων καθυστέρησης, εύρους ζώνης και ασφάλειας, ιδίως όταν απαιτείται επεξεργασία σε πραγματικό χρόνο. Η επεξεργασία στο άκρο αντιμετωπίζει αυτά τα ζητήματα, επιτρέποντας την τοπική επεξεργασία κοντά στην πηγή δεδομένων, μειώνοντας έτσι την απόκριση, το εύρος ζώνης και ενισχύοντας την ιδιωτικότητα. Αυτή η αλλαγή αναβαθμίζει τις εφαρμογές υπολογιστικής όρασης, όπως η ανάλυση βίντεο σε πραγματικό χρόνο, η αναγνώριση προσώπου και η επαυξημένη πραγματικότητα, οδηγώντας την καινοτομία και ανοίγοντας νέες δυνατότητες.

Η υπολογιστική όραση στοχεύει να γεφυρώσει το χάσμα μεταξύ της ανθρώπινης αντίληψης και της μηχανικής κατανόησης, ωστόσο αντιμετωπίζει σημαντικές προκλήσεις, ιδίως στην επεξεργασία μεγάλων όγκων οπτικών δεδομένων. Οι προσεγγίσεις της Βαθιάς Μάθησης (Deep Learning, DL), ιδιαίτερα τα Συνελικτικά Νευρωνικά Δίκτυα (Convolutional Neural Networks, CNNs), έχουν κρίσιμο ρόλο στη μοντελοποίηση των σχέσεων των εικονοστοιχείων και στην αναγνώριση σύνθετων προτύπων. Ωστόσο, η ενσωμάτωση πολυτροπικών δεδομένων, που συνδυάζουν οπτικές πληροφορίες με άλλους τύπους δεδομένων, είναι απαραίτητη για τη βελτίωση της ακρίβειας και της αποτελεσματικότητας. Η ενσωμάτωση πολυτροπικών δεδομένων προσθέτει σημαντικές προκλήσεις, όπως την κανονικοποίηση των διαφορετικών τύπων δεδομένων, την χρονική συσχέτισή τους αλλά και την επεξεργασία τους υπό την περιορισμένη υπολογιστική ισχύ των συσκευών στο άκρο.

Η παρούσα διατριβή στοχεύει στην αντιμετώπιση αυτών των προκλήσεων προτείνοντας μια νέα μεθοδολογία για την επεξεργασία στο άκρο, η οποία μας οδήγησε σε ένα επιταχυνόμενο πολυτροπικό πλαίσιο προσαρμοσμένο για

αυτά τα περιβάλλοντα. Αυτό το πλαίσιο επιτρέπει την εκτέλεση σύνθετης επεξεργασίας δεδομένων κοντά στην πηγή, αξιοποιώντας νέα μοντέλα τεχνητής νοημοσύνης και βελτιστοποιήσεις για διάφορες εφαρμογές, όπως η συμπεριφορική ανάλυση, ο εντοπισμός συμβάντων, η αναγνώριση αντικειμένων, η εκτίμηση εγγύτητας και η αναγνώριση προσώπου. Αυτές οι προσεγγίσεις δείχνουν σημαντικές βελτιώσεις στις ικανότητες λήψης αποφάσεων, στους χρόνους απόκρισης και στη συνολική απόδοση του συστήματος, ακόμη και υπό τους περιορισμένους πόρους των ενσωματωμένων συστημάτων.

Για να μειώσουμε την υπολογιστική επιβάρυνση της προεπεξεργασίας δεδομένων, έχουμε εφαρμόσει διάφορους σχεδιασμούς επεξεργασίας σε υλικό, εκμεταλλευόμενοι την παράλληλη επεξεργασία των FPGAs. Προτείνουμε έναν φορητό σχεδιασμό με τη γλώσσα περιγραφής υλικού VHSIC (VHSIC Hardware Description Language, VHDL) για την μετατροπή χρώματος και την ανίχνευση ακμών Sobel. Στη συνέχεια, παρουσιάζουμε μια βελτιωμένη παραλλαγή χρησιμοποιώντας Σύνθεση Υψηλού Επιπέδου (High-Level Synthesis, HLS) για αυξημένη αποδοτικότητα. Εξερευνούμε επίσης μια επιταχυμένη τεχνική μείωσης θορύβου βασισμένη στην στοίβαξη εικόνων, αναδεικνύοντας τις δυνατότητες επιτάχυνσης του υλικού για την επεξεργασία στο άκρο. Επιπλέον, η έρευνά μας εξετάζει τον συγκερασμό πολυτροπικών δεδομένων από διάφορους αισθητήρες, συνδυάζοντας πληροφορίες από πολλαπλές πηγές, όπως βίντεο, ήχο και άλλους αισθητήρες. Αυτή η προσέγγιση παρέχει μια πιο ολοκληρωμένη κατανόηση του περιβάλλοντος, αυξάνοντας την ακρίβεια του συστήματος σε σύνθετα, πραγματικά περιβάλλοντα.

Η αποτελεσματικότητα αυτής της μεθοδολογίας αποδεικνύεται μέσω μιας εκτενούς αξιολόγησης σε πραγματικά σύνολα δεδομένων και σενάρια ανάπτυξης στην δημόσια συγκοινωνία, με έμφαση στην ασφάλεια των επιβατών και τη λήψη αποφάσεων σε πραγματικό χρόνο. Τα αποτελέσματα πιστοποιούν την ευελιξία και την αποτελεσματικότητα της προτεινόμενης μεθοδολογίας και αναδεικνύουν την συνεισφορά της στον τομέα της επιταχυνόμενης επεξεργασίας στο άκρο χρησιμοποιώντας πολυτροπικά δεδομένα.

Εκτεταμένη Περίληψη στα Ελληνικά

Η επεξεργασία στο άκρο (edge processing) έχει αναδειχθεί ως ένας κρίσιμος τομέας στην υπολογιστική όραση (computer vision), οδηγούμενη από τις αυξανόμενες ποσότητες δεδομένων από συσκευές Διαδικτύου των Πραγμάτων (Internet of Things, IoT), τις έξυπνες κάμερες και τα αυτόνομα συστήματα. Οι παραδοσιακές αρχιτεκτονικές υπολογιστικής νέφους (cloud computing) παρουσιάζουν σημαντικά μειονεκτήματα στην αντιμετώπιση των ζητημάτων καθυστέρησης, εύρους ζώνης και ιδιωτικότητας, ιδίως όταν απαιτείται επεξεργασία σε πραγματικό χρόνο.

Η επεξεργασία στο άκρο αντιμετωπίζει αυτά τα ζητήματα, επιτρέποντας την τοπική επεξεργασία κοντά στην πηγή των δεδομένων και την αποστολή μόνο της απαραίτητης πληροφορίας στο νέφος (cloud). Ως αποτέλεσμα, περιορίζεται η χρήση του εύρους ζώνης (bandwidth) και ενισχύεται η ιδιωτικότητα των χρηστών. Η συγκεκριμένη μέθοδος κρίνεται ιδιαίτερα χρήσιμη για εφαρμογές υπολογιστικής όρασης που απαιτούν άμεση απόκριση, όπως η ανάλυση βίντεο σε πραγματικό χρόνο, και προσφέρει βελτιωμένες επιδόσεις, χαμηλότερη καθυστέρηση, μειωμένη χρήση εύρους ζώνης και ιδιωτικότητα.

Εστιάζοντας στον τομέα της υπολογιστικής όρασης, η επεξεργασία του μεγάλου όγκου δεδομένων από τους οπτικούς αισθητήρες αποτελεί σημαντική πρόκληση. Οι προσεγγίσεις της βαθιάς μάθησης (Deep Learning, DL) έχουν ιδιαίτερα σημαντικό ρόλο στη μοντελοποίηση των σχέσεων μεταξύ των εικονοστοιχείων (pixels) και στην αναγνώριση σύνθετων προτύπων, φέρνοντας επανάσταση στις εφαρμογές υπολογιστικής όρασης μέσω της αυτοματοποιημένης εξαγωγής χαρακτηριστικών με ιδιαίτερα υψηλή ακρίβεια.

Ξεκινώντας από τα θεμελιώδη συστατικά των σύγχρονων μεθόδων βαθιάς μάθησης, παρουσιάζονται τα Συνελικτικά Νευρωνικά Δίκτυα (Convolutional Neural Networks, CNNs) που αποτελούν θεμέλιο για σύνθετες προσεγγίσεις στο πεδίο της υπολογιστικής όρασης, χάρη στην ικανότητά εκμάθησης ιεραρχικών αναπαραστάσεων από τα παρεχόμενα δεδομένα. Η χρήση των συνελικτικών (convolutional) φίλτρων τα καθιστά ικανά για εξαγωγή χαρακτηριστικών

(feature extraction) από τις εικόνες σε πολλαπλά επίπεδα, επιτρέποντας την ανίχνευση τόσο απλών, όσο και σύνθετων μοτίβων. Ως αποτέλεσμα, τα CNNs αποτελούν τον ακρογωνιαίο λίθο στα μοντέλα βαθιάς μάθησης που ειδικεύονται στην αναγνώριση μοτίβων, στην κατηγοριοποίηση εικόνων, στην ανίχνευση αντικειμένων και στην δημιουργία νέου οπτικού περιεχομένου.

Πέρα από τα CNNs, παρουσιάζονται επίσης και τα Επαναληπτικά Νευρωνικά Δίκτυα (Recurrent Neural Networks, RNNs), τα οποία έχουν σχεδιαστεί για την επεξεργασία ακολουθιών. Η αρχιτεκτονική τους διατηρεί μια κρυφή κατάσταση που αποτυπώνει πληροφορίες από προηγούμενες εισόδους, καθιστώντας τα ιδανικά για εργασίες που απαιτούν την κατανόηση χρονοσειρών (timeseries) και ιστορικών πληροφοριών, όπως η μοντελοποίηση γλώσσας, η αναγνώριση ομιλίας και η πρόβλεψη χρονοσειρών. Παράλληλα, γίνεται αναφορά και σε μια ειδική κατηγορία των RNNs, τα Επαναληπτικά Δίκτυα Ευρείας Προσωρινής Μνήμης (Long Short-Term Memory, LSTM), τα οποία έχουν σχεδιαστεί για να αντιμετωπίζουν το πρόβλημα των εξαφανιζόμενων παραμέτρων (vanishing gradients) που παρατηρείται στα RNN δίκτυα, βελτιώνοντας έτσι την απόδοση σε μακροπρόθεσμες συσχετίσεις. Τέλος, παρουσιάζονται και τα Τρισδιάστατα Συνελικτικά Νευρωνικά Δίκτυα (3D Convolutional Neural Networks, 3D CNNs), τα οποία επεκτείνουν την έννοια της δισδιάστατης συνέλιξης στον χρονικό τομέα, εξάγοντας χωροχρονικά χαρακτηριστικά σε ακολουθίες εικόνων που περιλαμβάνουν τόσο την χωρική δομή των αντικειμένων, όσο και τα μοτίβα κίνησης τους. Ως εκ τούτου, είναι ιδανικά για την ανάλυση βίντεο, και ιδιαίτερα σε εφαρμογές όπως η ανίχνευση δραστηριότητας και η αναγνώριση χειρονομιών.

Με βάση τις προηγούμενες βασικές αρχιτεκτονικές νευρωνικών δικτύων παρουσιάζουμε την εκτίμηση στάσης (pose estimation) ως πρώτη τεχνική για την αναγνώριση της ανθρώπινης δραστηριότητας (activity recognition). Η εκτίμηση στάσης αναφέρεται στην διαδικασία προσδιορισμού της θέσης και του προσανατολισμού αντικειμένων ή ανθρώπινων μορφών σε εικόνες ή βίντεο. Ξεκινώντας με την εκτίμηση στάσης σε δύο διαστάσεις, αναλύονται οι μέθοδοι που βασίζονται σε απλές τεχνικές επεξεργασίας εικόνας, όπως η ανάλυση περιγραμμάτων και η τμηματοποίηση χρώματος. Στη συνέχεια, εξετάζονται τα μοντέλα που βασίζονται σε τμήματα, όπως το μοντέλο Εικονιστικών Δομών (Pictorial Structures), τα οποία αναπαριστούν το ανθρώπινο σώμα ως μια συλλογή από συνδεδεμένα μέρη. Στη συνέχεια, εξετάζεται η 3D εκτίμηση στάσης, η οποία προσδιορίζει την τρίτη διάσταση, το βάθος, προσφέροντας πληροφορίες απαραίτητες για εφαρμογές όπως η εικονική πραγματικότητα (Virtual

Reality, VR) και η βιομηχανική ανάλυση. Μερικά από τα πιο γνωστά μοντέλα για δισδιάστατη εκτίμηση στάσης περιλαμβάνουν τα OpenPose, AlphaPose και SimpleBaseline. Πιο συγκεκριμένα, το OpenPose χρησιμοποιεί CNNs και την έννοια των Συγγενικών Πεδίων Μερών (Part Affinity Fields, PAFs) για την ανίχνευση και την παρακολούθηση πολλών ατόμων σε πραγματικό χρόνο. Τέλος, το AlphaPose βασίζεται σε ένα δομημένο μοντέλο που αρχικά εντοπίζει άτομα σε μια σκηνή και στη συνέχεια εκτιμά τις στάσεις τους ξεχωριστά, ενώ το SimpleBaseline αποτελεί μια απλούστερη προσέγγιση, χρησιμοποιώντας βαθιά νευρωνικά δίκτυα για να αναβαθμίσει τα 2D σημεία-κλειδιά που εντοπίζονται σε εικόνες σε 3D χώρο.

Στη συνέχεια, αναλύεται το πεδίο της αναγνώρισης δραστηριότητας (activity recognition). Η αναγνώριση δραστηριότητας περιλαμβάνει την αναγνώριση και την κατηγοριοποίηση ενεργειών ή συμπεριφορών που απεικονίζονται σε εικόνες και βίντεο. Το αντίστοιχο κεφάλαιο περιγράφει την εξέλιξη της αναγνώρισης δραστηριότητας, από τις πρώτες μεθόδους που βασίζονταν σε χειροποίητα χαρακτηριστικά, όπως ο αλγόριθμος Μετασχηματισμού Κλιμακωτά Αναλλοίωτων Χαρακτηριστικών (Scale-Invariant Feature Transform, SIFT) και ο αλγόριθμος Ιστογραμμάτων Κατευθυνόμενων Κλίσεων (Histogram of Oriented Gradients, HOG), έως τα σύγχρονα μοντέλα βαθιάς μάθησης, όπως τα CNNs, τα RNNs και τα 3D CNNs. Παρουσιάζονται συγκεκριμένα μοντέλα για την αναγνώριση δραστηριότητας, όπως τα Συνελικτικά Τρισδιάστατα Δίκτυα (Convolutional 3D Networks, C3D), τα δίκτυα Χρονικών Τμημάτων (Temporal Segment Networks, TSNs), τα δίκτυα Χρονικής Μετατόπισης Μονάδας (Temporal Shift Module, TSM), τα δίκτυα SlowFast, καθώς και τα μοντέλα "Expand, Excite, Extend, Depthwise Separable 3D Convolutions" (X3D) και "Mobile Video Networks" (MoViNets). Κάθε μοντέλο παρουσιάζει μοναδικές καινοτομίες και βελτιώσεις σε σχέση με τις παραδοσιακές προσεγγίσεις.

Εκτός από τα παραπάνω πεδία, δίνεται έμφαση και στο πεδίο της ανίχνευσης αντικειμένων (object detection), το οποίο επιτρέπει στους υπολογιστές να αναγνωρίζουν και να εντοπίζουν αντικείμενα σε οπτικά δεδομένα. Στο παραπάνω πλαίσιο, γίνεται αναφορά στην εξέλιξη της ανίχνευσης αντικειμένων, από τις πρώτες μεθόδους που βασίζονταν σε χειροποίητα χαρακτηριστικά έως τα σύγχρονα μοντέλα βαθιάς μάθησης, όπως τα Συνελικτικά Νευρωνικά Δίκτυα που βασίζονται στην Περιοχή (Region-Based Convolutional Neural Networks, R-CNNs) και τα μοντέλα "You Only Look Once" (YOLO) και Ανιχνευτές ενός Σταδίου (Single-Shot Detectors, SSDs).

Τέλος, το επόμενο πεδίο που αναλύεται είναι η αναγνώριση προσώπου (face

recognition), με έμφαση στην χρήση CNNs για την εκμάθηση και την εξαγωγή χαρακτηριστικών. Παρουσιάζονται οι βασικές μεθοδολογίες για την αναγνώριση προσώπου, όπως οι αλγόριθμοι "Viola-Jones", οι μέθοδοι που βασίζονται σε γεωμετρικά χαρακτηριστικά και οι προσεγγίσεις που βασίζονται στην εμφάνιση του προσώπου. Επιπλέον, εξετάζονται σημαντικά μοντέλα βαθιάς μάθησης για την αναγνώριση προσώπου, όπως τα Siamese νευρωνικά δίκτυα (Siamese Neural Networks, SNNs), το "DeepFace" και το "FaceNet". Τα SNNs αποτελούν μια κλάση αρχιτεκτονικών CNN που χαρακτηρίζονται από τα δίδυμα (siamese) υποδίκτυά τους, τα οποία μοιράζονται βάρη και παραμέτρους. Το DeepFace και το FaceNet, τα οποία αναπτύχθηκαν από τις εταιρίες Facebook και Google αντίστοιχα, σημείωσαν σημαντική πρόοδο στην ακρίβεια της αναγνώρισης προσώπου.

Η αποτελεσματικότητα των παραπάνω μεθόδων είναι άρρηκτα συνδεδεμένη τόσο με την ποσότητα, όσο και με την ποιότητα των δεδομένων εκπαίδευσης, αλλά και με τη διαθέσιμη υπολογιστική ισχύ για την επεξεργασία τους. Στην παρούσια διδακτορική διατριβή προτείνεται μια νέα μεθολογία για την βελτίωση της ακρίβειας και της αποτελεσματικότητας των προαναφερθέντων προσεγγίσεων με την ενσωμάτωση πολυτροπικών (multimodal) δεδομένων. Ο συγκερασμός (fusion) πολυτροπικών δεδομένων όπως οπτικά, δεδομένα βάθους, πανοραμικά και φασματογραφήματα ήχου βελτιώνουν την ακρίβεια και την αποτελεσματικότητα των μεμονωμένων μεθόδων. Για παράδειγμα, ένας συνδυασμός δεδομένων από αισθητήρες ήχου, βάθους και πανοραμικών καμερών μπορεί να προσφέρει μια πιο ολοκληρωμένη εικόνα του περιβάλλοντος χώρου σε αντίθεση με τη χρήση μόνο ενός τύπου δεδομένων, επιτρέποντας έτσι και πιο ακριβείς και αξιόπιστες αποφάσεις. Η επεξεργασία πολυτροπικών δεδομένων μπορεί να χρησιμοποιηθεί σε ένα ευρύ φάσμα εφαρμογών, από συστήματα αυτόματης οδήγησης, μέχρι και σε εφαρμογές παρακολούθησης υγείας και ασφαλείας, όπου η συνδυαστική ανάλυση δεδομένων από διάφορους τύπους πηγών είναι σημαντική για την εξαγωγή βέλτιστων συμπερασμάτων. Ωστόσο, η ενσωμάτωση πολυτροπικών δεδομένων επιφέρει επίσης και σημαντικές προκλήσεις, όπως η κανονικοποίηση των διαφορετικών τύπων δεδομένων, η χρονική συσχέτισή τους αλλά και η επεξεργασία τους υπό την περιορισμένη υπολογιστική ισχύ των συσκευών στα άκρα του δικτύου, οι οποίες διαθέτουν περιορισμένους πόρους σε σχέση με τα κεντρικά συστήματα επεξεργασίας, γεγονός που απαιτεί αποδοτικά μοντέλα και αλγορίθμους τεχνητής νοημοσύνης (Artificial Intelligence, AI).

Η διδακτορική διατριβή στοχεύει στην αντιμετώπιση των προαναφερθέντων

προκλήσεων, προτείνοντας μια νέα μεθοδολογία επεξεργασίας στο άκρο (edge processing), η οποία οδήγησε στη σχεδίαση και την υλοποίηση ενός επιταχυνόμενου (accelerated) πολυτροπικού πλαισίου (framework), ειδικά προσαρμοσμένο σε περιβάλλοντα χαμηλής υπολογιστικής ισχύος. Το προτεινόμενο πλαίσιο επιτρέπει την εκτέλεση σύνθετης επεξεργασίας δεδομένων κοντά στην πηγή, αξιοποιώντας νέα μοντέλα τεχνητής νοημοσύνης και βελτιστοποιήσεις για διάφορες εφαρμογές, όπως η συμπεριφορική ανάλυση, ο εντοπισμός συμβάντων, η αναγνώριση αντικειμένων, η εκτίμηση εγγύτητας και η αναγνώριση προσώπου, οι οποίες παρουσιάζουν σημαντικές βελτιώσεις στη λήψη αποφάσεων, στους χρόνους απόκρισης, καθώς και στη συνολική απόδοση του συστήματος, ακόμη και υπό τους περιορισμένους πόρους των ενσωματωμένων συστημάτων.

Για τον περιορισμό του υπολογιστικού κόστους της προεπεξεργασίας (preprocessing) των δεδομένων στις edge συσκευές, εστιάσαμε τόσο στην επιτάχυνση του υλικού (hardware), όσο και στην βελτιστοποίηση του λογισμικού (software). Αρχικά, εξετάσαμε την επιτάχυνση υλικού, χρησιμοποιώντας πλακέτες Συστοιχιών Επιτόπια Προγραμματιζόμενων Πυλών (Field-Programmable Gate Arrays, FPGAs) για την υλοποίηση αλγορίθμων προεπεξεργασίας εικόνας, όπως μετατροπή χρώματος και ανίχνευση ακμών.

Η πρώτη υλοποίηση περιγράφεται σε γλώσσα περιγραφής υλικού VHSIC (VHSIC Hardware Description Language, VHDL) στην πλατφόρμα Altera DE2-115 FPGA. Συγκεκριμένα, σχεδιάσαμε ένα σύστημα επεξεργασίας εικόνας που εστιάζει στην μετατροπή χρώματος και στην ανίχνευση ακμών Sobel. Η χρωματική μετατροπή σε κλίμακα του γκρι ακολουθεί τις προδιαγραφές ITU-R BT.709, ενώ η ανίχνευση ακμών χρησιμοποιεί πυρήνες ανίχνευσης οριζόντιων και κατακόρυφων κλίσεων βασισμένους στον τελεστή Sobel. Το σύστημα ενσωματώνεται στο FPGA μέσω ενός soft CPU NIOS II, το οποίο επικοινωνεί αμφίδρομα κάνοντας χρήση του πρωτοκόλλου USB για την αλληλεπίδραση με τα περιφερειακά επεξεργασίας εικόνας. Συνεχίσαμε κάνοντας χρήση της Σύνθεσης σε Υψηλό Επίπεδο (High Level Synthesis, HLS) στην πλατφόρμα Xilinx Pynq-Z1, ώστε να αξιολογήσουμε τα πλεονεκτήματα και μειονεκτήματα των δύο μεθοδολογιών, του σχεδιασμού σε VHDL και της μεθοδολογία Υψηλού Επιπέδου Σύνθεσης (High-Level Synthesis, HLS). Πιο συγκεκριμένα, η υλοποίηση HLS προσέφερε ταχύτερη δημιουργία πρωτοτύπων και ανάπτυξη, αξιοποιώντας το πλαίσιο Pynq και τις ευκολίες της γλώσσας προγραμματισμού Python. Ο σχεδιασμός για ανίχνευση ακμών με σύνθεση υψηλού επιπέδου περιλαμβάνει τέσσερα στάδια: (1) Μετατροπή της εισόδου σε κλίμακας του γκρι (AXIS2GrayArray), (2) Εφαρμογή τελεστή Sobel με αλγόριθμο Non-Max sup-

pression για βελτίωση ακρίβειας, (3) Κωδικοποίηση χρώματος RGB σε 32-bit ακέραιο (GrayArray2AXIS) για εκφόρτωση υπολογισμών από την CPU στο PL, και (4) Διμερής DMA (AXI4-Stream) για βελτιστοποιημένη μεταφορά δεδομένων. Η σύγκριση των δύο υλοποιήσεων (VHDL και HLS) έδειξε ότι η μέθοδος HLS πέτυχε καλύτερη απόδοση και αποτελεσματικότητα, ενώ ταυτόχρονα απλοποίησε σημαντικά την διαδικασία ανάπτυξης, χάρη στην προ-υλοποιημένη υποστήριξη βασικών λειτουργιών υλικού (π.χ. πρωτόκολλο AXI4).

Πέρα από την προεπεξεργασία, παρουσιάζονται εξελιγμένοι αλγόριθμοι τεχνητής νοημοσύνης, ειδικά βελτιστοποιημένοι για πολυτροπική επεξεργασία στο άκρο. Οι προσεγγίσεις που παρουσιάζονται αξιοποιούν δεδομένα από πολλαπλές πηγές αισθητήρων, με εστίαση στην αναγνώριση δραστηριότητας, αντικειμένων, προσώπου και ηχητικών σημάτων.

Ξεκινώντας με την αναγνώριση δραστηριότητας με την ταξινόμηση των στάσεων σώματος, αναπτύσσουμε μια διαδικασία τεσσάρων σταδίων: (1) ανίχνευση σημείων - κλειδιών του ανθρώπινου σώματος σε κάθε καρέ, (2) παρακολούθηση σκελετού σε πολλαπλά καρέ, (3) εξαγωγή χαρακτηριστικών από τα σημεία - κλειδιά και (4) ταξινόμηση της στάσης σώματος με LSTM. Η εξαγωγή χαρακτηριστικών περιλαμβάνει τη χρήση ενός κυκλικού buffer για την αποθήκευση των τροχιών σκελετού και εφαρμόζεται η μέθοδος Ανάλυσης Κύριων Συνιστωσών (Principal Component Analysis, PCA) για τη μείωση της διάστασης των χαρακτηριστικών, από 314 σε 50, βελτιώνοντας την ακρίβεια και την ταχύτητα του μοντέλου.

Στη συνέχεια, παρουσιάζουμε την ανάπτυξη ενός Χωροχρονικού Αυτοκωδικοποιητή (Spatiotemporal Autoencoder, SAE), ο οποίος εκπαιδεύεται σε κανονικά δεδομένα και ανιχνεύει μη φυσιολογικά γεγονότα βασιζόμενος στην απόκλιση από τα πρότυπα των δεδομένων εκπαίδευσης. Η αρχιτεκτονική του μοντέλου περιλαμβάνει ένα χωρικό αυτοκωδικοποιητή για την εκμάθηση χωρικών δομών και ένα χρονικό κωδικοποιητή - αποκωδικοποιητή για την εκμάθηση χρονικών προτύπων. Για την εκπαίδευση του μοντέλου, χρησιμοποιούνται καρέ από βίντεο με διαφορετικά βήματα παραλείψεων.

Εξετάζεται επίσης η υλοποίηση ενός πανοραμικού συστήματος ανίχνευσης αντικειμένων, με κατάλληλα τοποθετημένες κάμερες fisheye. Η αρχιτεκτονική του συστήματος περιλαμβάνει τρία στάδια: (1) δίκτυο backbone για την εξαγωγή χαρακτηριστικών, (2) δίκτυο FPN για την ενίσχυση των χαρακτηριστικών και (3) δίκτυο ανίχνευσης για την πρόβλεψη των περιγραμμάτων και της περιστροφής των αντικειμένων. Στην μεθοδολογία παρουσιάζεται μια νέα τεχνική ανίχνευσης περιγραμμάτων με πρόβλεψη περιστροφής, για την αντι-

μετώπιση της παραμόρφωσης των fisheye φακών. Επιπλέον, χρησιμοποιείται μια συνάρτηση απώλειας (loss function) που λαμβάνει υπόψη την κλίμακα των αντικειμένων, βελτιώνοντας την ακρίβεια της ανίχνευσης σε εικόνες fisheye.

Στο πεδίο της αναγνώρισης προσώπου, παρουσιάζεται η μεθοδολογία ενός ισχυρού συστήματος χρησιμοποιώντας δίδυμα δίκτυα Siamese και την αρχιτεκτονική ArcFace. Το σύστημα αυτό υπολογίζει ομοιότητες μεταξύ ζευγών εικόνων, χρησιμοποιώντας ένα εκπαιδευμένο δίκτυο συνελικτικών νευρώνων (CNN) για την εξαγωγή χαρακτηριστικών και ένα πρόσθετο module (xCos) για τον υπολογισμό ομοιότητας μεταξύ των χαρτών χαρακτηριστικών. Η ενσωμάτωση μηχανισμού προσοχής (attention mechanism) βοηθά στην αντιμετώπιση των προκλήσεων που προκύπτουν από εμπόδια, όπως οι ιατρικές μάσκες προσώπου, επιτρέποντας την ακριβή αναγνώριση ατόμων ακόμα και όταν το πρόσωπό τους είναι μερικώς καλυμμένο.

Τέλος, περιγράφεται και η μεθοδολογία ενός συστήματος ταξινόμησης ήχου, με στόχο την ανίχνευση μη φυσιολογικών γεγονότων σε ηχητικά δεδομένα. Για την αντιμετώπιση των προκλήσεων που σχετίζονται με το θόρυβο, χρησιμοποιείται μοντέλο DenseNet-121, το οποίο εκπαιδεύεται σε δεδομένα με διαφορετικές αναλογίες σηματοθορυβικού λόγου (Signal-to-Noise Ratio, SNR). Η ανθεκτικότητα και η ικανότητα γενίκευσης του μοντέλου αξιολογούνται σε πραγματικά δεδομένα, αποδεικνύοντας την αποτελεσματικότητά του στην ανίχνευση μη φυσιολογικών γεγονότων σε ποικίλα ακουστικά περιβάλλοντα.

Στην συνέχεια, δίνεται έμφαστη στην αρχιτεκτονική του πλαισίου για την επεξεργασία στο άκρο. Η αρχιτεκτονική περιλαμβάνει ένα επίπεδο αφαίρεσης υλικού (Hardware Abstraction Layer, HAL) για την διαχείριση πολλαπλών αισθητήρων, καθώς και μηχανισμούς διεργασιακής επικοινωνίας (Inter-Process Communication, IPC) για την ανταλλαγή δεδομένων. Η χρήση εικονικών συσκευών βίντεο μέσω v4l2loopback και η εφαρμογή μηχανισμών συγχρονισμού δεδομένων από διαφορετικές πηγές εξασφαλίζουν την αξιόπιστη λειτουργία του πλαισίου σε πραγματικό χρόνο.

Η προτεινόμενη μεθοδολογία που οδήγησε στην δημιουργία του παραπάνω πλαισίου αξιολογείται σε πρακτική εφαρμογή του σε μέσα μαζικής μεταφοράς και συγκεκριμένα στα αυτόνομα οχήματα (Autonomous Vehicles, AVs). Η επιλογή των αυτόνομων οχημάτων βασίζεται στις απαιτητικές συνθήκες λειτουργίας τους και τους ενεργειακούς περιορισμούς, οι οποίοι συνάδουν με τη μεθοδολογία της επεξεργασίας στο άκρο. Τα αυτόνομα οχήματα απαιτούν λήψη αποφάσεων και άμεση απόκριση σε πραγματικό χρόνο, κάτι που μπορεί να προσφέρει η αποκεντρωμένη αρχιτεκτονική και η επεξεργασία χαμηλής

καθυστέρησης στο άκρο. Η ενεργειακή απόδοση της επεξεργασίας στο άκρο συμπληρώνει ιδανικά τη λειτουργία των αυτόνομων οχημάτων με μπαταρία, μειώνοντας τις ανάγκες επικοινωνίας και επεκτείνοντας την αυτονομία τους. Επιπλέον, ενισχύεται η ιδιωτικότητα, καθώς δεν μεταδίδονται ευαίσθητα δεδομένα εκτός του οχήματος.

Οι βασικές ανησυχίες των επιβατών σε αυτόνομα οχήματα περιλαμβάνουν την ψυχολογική δυσφορία λόγω έλλειψης ανθρώπινης επίβλεψης που δημιουργεί φόβους που σχετίζονται με την ασφάλεια εντός της καμπίνας. Οι μέθοδοι και οι προσεγγίσεις μας εφαρμόστηκαν μέσω του πολυτροπικού πλαισίου για την ανίχνευση και την απόκριση σε διάφορα περιστατικά ασφάλειας, κάνοντας χρήση προηγμένων αισθητήρων παρακολούθησης. Επιπρόσθετα, παρουσιάζονται αποτελέσματα και μελέτες για την συνεισφορά της προτεινόμενης μεθοδολογίας σε πραγματικά περιβάλλοντα και σενάρια, όπως για την καταμέτρηση επιβατών, την ανίχνευση επιθέσεων και συμβάντων καθώς και την έγκαιρη ενημέρωση. Επιπλέον, εξετάζεται η αποδοτικότητα του προτεινόμενου συστήματος με βάση τους περιορισμένους υπολογιστικούς πόρους και την κατανάλωση ενέργειας. Το προτεινόμενο πολυτροπικό σύστημα ανίχνευσης σχεδιάστηκε για να λειτουργεί σε πραγματικό χρόνο εντός των περιορισμών ενός ενσωματωμένου συστήματος, διασφαλίζοντας την άμεση ανίχνευση κρίσιμων συμβάντων σε αυτόνομα οχήματα με χαμηλή κατανάλωση ενέργειας. Για τη βελτιστοποίηση της υπολογιστικής αποδοτικότητας, εφαρμόστηκαν διάφορες στρατηγικές. Πρώτον, η αρχιτεκτονική του Συνελικτικού Αυτόματου Κωδικοποιητή (Convolutional AutoEncoder, CAE) χρησιμοποιεί διαχωριστικές συνελίξεις βάθους, οι οποίες μειώνουν σημαντικά το μέγεθος των παραμέτρων σε σύγκριση με τις τυπικές συνελίξεις. Επιπλέον, οι βελτιστοποιήσεις δικτύου μέσω εργαλείων όπως το TensorRT επιτρέπουν στο μοντέλο να λειτουργεί αποδοτικά στην πλατφόρμα NVIDIA Jetson. Τα πειραματικά αποτελέσματα δείχνουν ότι η προσέγγισή μας επιτυγχάνει ταχύτητα επεξεργασίας της τάξεως των 37 καρέ ανά δευτερόλεπτο (Frames Per Second, FPS) στο ενσωματωμένο σύστημα NVIDIA Jetson AGX Xavier. Η εφαρμογή του πλαισίου σε πραγματικό περιβάλλον αυτοκινούμενων οχημάτων αποτελεί μια κρίσιμη δοκιμή για την αξιολόγηση της αποτελεσματικότητας και της απόδοσής του, παρέχοντας πολύτιμες πληροφορίες για τις δυνατότητες και τους περιορισμούς του.

Συνοψίζοντας, η παρούσα διατριβή αντιμετωπίζει τις προκλήσεις της πολυτροπικής επεξεργασίας στο άκρο προτείνοντας μια νέα μεθοδολογία σχεδίασης, που οδήγησε σε ένα επιταχυνόμενο πολυτροπικό πλαίσιο. Το πλαίσιο αυτό επιτρέπει την επεξεργασία δεδομένων απευθείας στην πηγή, αξιοποιώντας

νέα μοντέλα βαθιάς μάθησης και βελτιστοποιήσεις για διάφορες εφαρμογές, όπως η αναγνώριση δραστηριότητας, ο εντοπισμός αντικειμένων, ο υπολογισμός της εγγύτητας μεταξύ των αντικειμένων και η αναγνώριση προσώπων. Ένα από τα κύρια ευρήματα αυτής της έρευνας είναι ο συγκερασμός πολυτροπικών δεδομένων εντός του πλαισίου. Με τον συνδυασμό πολλαπλών αισθητήρων, επιτυγχάνεται μια πιο ολοκληρωμένη κατανόηση του περιβάλλοντος, οδηγώντας σε σημαντικές βελτιώσεις στην ακρίβεια και την αξιοπιστία των αλγορίθμων σε σύνθετα και πραγματικά περιβάλλοντα. Η έμφαση στην επιτάχυνση του υλικού και οι τεχνικές βελτιστοποίησης μετρίασαν αποτελεσματικά το υπολογιστικό κόστος, επιτρέποντας την επεξεργασία στο άκρο σε πραγματικό χρόνο. Τα ευρήματα αυτής της διατριβής έχουν σημαντικές επιπτώσεις για το μέλλον της τεχνητής νοημοσύνης και της επεξεργασίας στο άκρο. Η πολυτροπική επεξεργασία σε ενσωματωμένες συσκευές θέτει τις βάσεις για μια νέα γενιά έξυπνων συστημάτων που είναι ικανά να λειτουργούν αυτόνομα και να προσαρμόζονται σε δυναμικά περιβάλλοντα.

Acknowledgements

I would like to thank everyone who supported and guided me throughout the journey of completing this dissertation.

First and foremost, I want to express my deepest gratitude to my supervisor, Dr. Minas Dasygenis, for his constant support, essential counsel and constructive input throughout this research work. His expertise, encouragement and patience have been highly important in shaping this research.

I am also thankful to the members of my doctoral advisory committee, Dr. Konstantinos Siozios and Dr. Nikolaos Konofaos for their insightful comments, recommendations and expertise, which have improved the overall quality of this thesis.

I would also like to extend my appreciation to Dr. Antonios Lalas and Dr. Konstantinos Votis, for supporting this research, enriching its overall scope and impact.

Finally, I am deeply grateful to my family for their forever faith, encouragement and understanding; they have been a constant source of strength and inspiration.

Declarations

I hereby declare that this dissertation represents my own work that has been completed as part of my doctoral program at the University of Western Macedonia, faculty of Electrical and Computer Engineering. Parts of the presented work in this document have been previously published in journals, conference proceedings, as well as book chapters.

The following list showcases the papers that correspond to the methodology and results described in **Chapter 3** ("Accelerated Multimodal Framework for Edge Computing") and **Chapter 4** ("Enhancing Safety and Security in Public Transportation"), respectively:

[1] D. Tsiktsiris, D. Ziouzios and M. Dasygenis, "A portable image processing accelerator using FPGA," in 2018 7th International Conference on Modern Circuits and Systems Technologies (MOCAST), 2018

[2] D. Tsiktsiris, D. Ziouzios and M. Dasygenis, "A High-Level synthesis implementation and evaluation of an image processing accelerator," Technologies, vol. 7, p. 4, 2018.

[3] D. Tsiktsiris, D. Ziouzios and M. Dasygenis, "HLS Accelerated Noise Reduction Approach Using Image Stacking on Xilinx PYNQ," in 2019 8th International Conference on Modern Circuits and Systems Technologies (MOCAST), 2019.

[4] D. Tsiktsiris, K. Kechagias, M. Dasygenis, and P. Angelidis. "Accelerated seven segment optical character recognition algorithm". In 2019 Panhellenic Conference on Electronics & Telecommunications (PACET), 2019.

[5] D. Tsiktsiris, N. Dimitriou, A. Lalas, M. Dasygenis, K. Votis and D. Tzovaras, "Real-time abnormal event detection for enhanced security in autonomous shuttles mobility infrastructures," Sensors, vol. 20, p. 4943, 2020.

[6] D. Tsiktsiris, A. Lalas, M. Dasygenis, K. Votis and D. Tzovaras, "Enhanced Security Framework for Enabling Facial Recognition in Autonomous Shuttles Public Transportation During COVID-19," in IFIP International Conference on Artificial Intelligence Applications and Innovations, 2021.

[7] D. Tsiktsiris, A. Vafeiadis, A. Lalas, M. Dasygenis, K. Votis and D. Tzovaras, "A novel image and audio-based artificial intelligence service for security applications in autonomous vehicles," Transportation Research Procedia, vol. 62, p. 294–301, 2022.

[8] D. Tsiktsiris, T. Sanida, A. Sideris, and M. Dasygenis. "Accelerated Defective Product Inspection on the Edge Using Deep Learning". In Recent Advances in Manufacturing Modelling and Optimization: Select Proceedings of RAM 2021, Springer, 2022, p. 185–191.

[9] D. Tsiktsiris, A. Lalas, M. Dasygenis, K. Votis and D. Tzovaras, "An Efficient Method for Addressing COVID-19 Proximity Related Issues in Autonomous Shuttles Public Transportation," in Artificial Intelligence Applications and Innovations: 18th IFIP WG 12.5 International Conference, AIAI 2022, Hersonissos, Crete, Greece, June 17–20, 2022, Proceedings, Part I, 2022.

[10] D. Tsiktsiris, A. Lalas, M. Dasygenis and K. Votis, "Improving Passenger Detection with Overhead Fisheye Imaging", IEEE Access, 2024.

[11] D. Tsiktsiris, A. Lalas, M. Dasygenis and K. Votis, "Enhancing the Safety of Autonomous Vehicles: Semi-Supervised Anomaly Detection with Overhead Fisheye Perspective", IEEE Access, 2024.

Contents

List of Figures

List of Tables

List of Abbreviations

Abbreviation	Definition
ADAS	Advanced Driver Assistance System
AI	Artificial Intelligence
APC	Automated Passenger Counting
API	Application Programming Interface
AR	Augmented Reality
ASIC	Application-Specific Integrated Circuit
AUC	Area Under the Curve
AV	Autonomous Vehicle
AVOF	Autonomous Vehicle Overhead Fisheye
BRAM	Block Random-Access Memory
C3D	Convolutional 3D
CAE	Convolutional Autoencoder
CEPDOF	Challenging Events for Person Detection from Overhead Fisheye
CERTH	Centre for Research and Technology Hellas
CNN	Convolutional Neural Network
ConvLSTM	Convolutional Long Short-Term Memory
dB	Decibel
DL	Deep Learning
DMA	Direct Memory Access
Faster R-CNN	Faster Region-Based Convolutional Neural Network
FNN	Feed-Forward Neural Network

FPGA	Field-Programmable Gate Array
FPN	Feature Pyramid Network
GAN	Generative Adversarial Network
GB	Gigabyte
GPU	Graphics Processing Unit
GRU	Gated Recurrent Unit
HABBOF	Human-Aligned Bounding Boxes from Overhead Fisheye
HAL	Hardware Abstraction Layer
HFR	High Frame Rate
HLS	High-Level Synthesis
HMI	Human-Computer Interaction
HOG	Histogram of Oriented Gradients
HTTP	HyperText Transfer Protocol
ID	Identification
IoT	Internet of Things
IOU	Intersection Over Union
IP	Internet Protocol
IPC	Inter-Process Communication
ISO	International Organization for Standardization
k-NN	k-Nearest Neighbor
LBP	Local Binary Pattern
LDA	Linear Discriminant Analysis
LFR	Low Frame Rate
LFW	Labeled Faces in the Wild
LSTM	Long Short-Term Memory
ML	Machine Learning
MoViNet	Mobile Video Network
MQTT	Message Queuing Telemetry Transport
MSE	Mean Squared Error
NMS	Non-Maximum Suppression
NTU	Nanyang Technological University

ONVIF	Open Network Video Interface Forum
OS	Operating System
PAF	Part Affinity Field
PCA	Principal Component Analysis
PSNR	Peak Signal-to-Noise Ratio
QoRs	Quality of Results
RAM	Random-Access Memory
ReLU	Rectified Linear Unit
ResNet	Residual Network
RGB	Red-Green-Blue
RMPE	Regional Multi-Person Pose Estimation
RNN	Recurrent Neural Network
ROC	Receiver Operating Characteristic
RTL	Register-Transfer Level
RTSP	Real Time Streaming Protocol
SDG	Stochastic Gradient Descent
SIFT	Scale-Invariant Feature Transform
SNN	Siamese Neural Network
SNR	Signal-to-Noise Ratio
SPPE	Single Person Pose Estimator
SSD	Single Shot Detector
STN	Spatial Transformer Network
SSTN	Symmetric Spatial Transformer Network
SVM	Support Vector Machine
TCP	Transmission Control Protocol
TPG	Transports Publics Genevois
TSM	Temporal Shift Module
TSN	Temporal Segment Networks
t-SNE	t-distributed Stochastic Neighbor Embedding
UC	Use Case
UCF	University of Central Florida
UDP	User Datagram Protocol
USB	Universal Serial Bus

UTC	Coordinated Universal Time
VGG	Visual Geometry Group
VHDL	VHSIC Hardware Description Language
VHSIC	Very High Speed Integrated Circuit
VPN	Virtual Private Network
VR	Virtual Reality
X3D	Expand, Excite, Extend, Depthwise Separable 3D Convolutions
YOLO	You Only Look Once

1

Introduction

Edge processing has become a critical domain in the field of computer vision, due to the exponential growth of data generated by Internet of Things (IoT) devices [1], smart cameras [2], and autonomous systems [3]. Traditional cloud computing architectures are often unsuitable for real time applications due to latency, bandwidth, and security concerns, particularly when real-time processing is essential [4]. Edge computing mitigates these issues by enabling data to be processed locally or near the source, therefore reducing latency and bandwidth usage [5], and enhancing the speed and efficiency of computer vision applications [6]. Furthermore, edge processing improves data security and privacy by eliminating the transmission of sensitive information, which is crucial for compliance with regulations [7]. This shift not only addresses the limitations of cloud-centric models but also empowers advanced computer vision applications like real-time video analytics, facial recognition, and augmented reality (AR), driving innovation and opening new possibilities in various computer vision domains.

The field of computer vision bridges the gap between human perception and machine understanding and aims to empower computers with the ability to "see" and interpret visual information. This progress comes with significant challenges, especially in handling large amounts of data that are required to represent visual inputs. Those inputs, such as images and videos, are comprised

of millions of pixels which represent unique details across color channels. The processing of such pixels requires complex and resource-demanding approaches, especially when targeting real-time applications. To mitigate this processing complexity, deep learning (DL) approaches have been introduced to model those pixel relations by training on large datasets. Among the most popular DL approaches, Convolutional Neural Networks (CNNs) are able learn hierarchical representations of visual data. This enables them to recognize complex patterns, classify images, detect objects, and even generate new visual content [8].

However, the advancements in computer vision, extend in different types of visual data. All these types of data (depth maps, thermal, audio spectograms), which refer to the same environment, form the concept of multimodal data [9]. The integration of multimodal data processing by employing the fusion of multiple sources of data has become essential for improving accuracy and robustness in computer vision tasks [10]. Algorithms can achieve a more comprehensive understanding of the environment by leveraging multiple data sources, leading to more accurate and contextually aware applications [11]. A multimodal approach not only increases the performance of individual models but also enables new applications and innovations that were previously unattainable with single-modal data. As edge computing continues to evolve, its ability to process and fuse multiple modalities locally will be crucial for advancing the state-of-the-art in computer vision and beyond.

Despite its advantages, multimodal approaches pose a significant practical challenge due to the need to handle and align diverse data types [12]. Each modality may have different characteristics, resolution and temporal alignments, making the integration complex. For instance, the synchronization of depth maps and audio spectrograms require precise temporal alignment and robust data fusion techniques to ensure coherent and meaningful outputs. Moreover, the resource constraints of embedded devices, which are often used in edge computing environments, add another layer of complexity. These devices must process large volumes of diverse data in real-time while operating within limited power, memory, and computational resources. As edge computing continues to evolve, developing efficient algorithms and architectures capable of handling and integrating multimodal data on resource-constrained devices will be crucial for advancing the state-of-the-art in computer vision and beyond.

In this dissertation, we present the results of our research that resulted in the development of a multimodal edge computing framework. This framework combines the strengths of hardware acceleration [13] for preprocessing and han-

dling the diverse data types of a multimodal environment with state-of-the-art accelerated models for various computer vision tasks, including action recognition, object detection, facial identification, and sound event detection. Our approach utilizes specialized hardware components, such as Graphic Processing Units (GPUs) and Field-Programmable Gate Array (FPGAs), to perform efficient data preprocessing and integration, ensuring that the computational cost is minimized on resource-constrained edge devices. Additionally, we integrate advanced DL models optimized for real-time performance, capable of delivering high accuracy and robustness across different modalities. By addressing both the preprocessing and computational challenges, our framework demonstrates significant improvements in speed, efficiency, and accuracy, shaping the way for enhanced edge computing applications in computer vision. The outcome of this research highlight the potential for deploying sophisticated multimodal processing techniques in edge environments, contributing to the ongoing advancement of intelligent and responsive computer vision systems.

The effectiveness and applicability of our methodology are demonstrated through detailed evaluation on real-world datasets and deployment scenarios in public transportation. Our method provides a solid foundation for ensuring passenger safety and real-time decision-making using sophisticated AI algorithms. The results underscore the versatility and robustness of our framework, showcasing its potential to transform diverse applications through the power of edge computing and multimodal data processing.

1.1 Research Gaps and Challenges

Addressing technical challenges requires a holistic methodology that combines advancements in algorithmic design, hardware optimization, and system architecture. While significant progress has been made in each of these areas, a critical need for continued innovation remains in order to develop solutions that can seamlessly integrate and process multimodal data on resource-constrained edge devices. This innovation must not only focus on enhancing the raw computational power and efficiency of these systems, but also on creating more sophisticated models and algorithms capable of handling the inherent complexities of multimodal data. Additionally, practical considerations such as security, privacy, and scalability must be thoroughly examined to ensure that these advanced systems can be effectively deployed and maintained in real-world en-

vironments. In the following sections, we will delve deeper into the specific research gaps and challenges within these domains, highlighting the key areas that require further exploration and development to fully harness the potential of edge computing in computer vision applications.

1.1.1 Data Integration and Alignment

One of the primary technical challenges is the efficient integration and synchronization of multimodal data [14]. Given the diversity of data types, such as RGB images, depth maps, and audio spectrograms, achieving precise temporal alignment and robust data fusion remains a complex task [15]. Each modality presents unique characteristics and varying resolutions, complicating the process of standardizing them to produce coherent and meaningful outputs. This challenge is significant in real-time applications where any latency or misalignment can significantly impact performance and accuracy.

1.1.2 Resource Constraints

In addition to data integration issues, the resource constraints in edge devices pose a significant barrier [16]. Edge devices often operate within strict limitations on power, memory, and computational resources. Developing algorithms and architectures that can efficiently process large volumes of diverse data under these constraints is crucial. This requires innovative approaches to optimize both software and hardware components, ensuring that high-performance computing can be achieved without exceeding the limited capacities of edge devices. Moreover, the rapid evolution of DL models introduces further challenges. While these models significantly enhance the capabilities of computer vision systems, they also demand substantial computational power and memory [17]. Training and deploying these models on resource-constrained edge devices require efficient model optimization techniques, such as model pruning, quantization, and the use of specialized hardware accelerators like GPUs and FPGAs. Ensuring that these optimized models maintain high accuracy and robustness across different modalities is essential for their practical deployment.

1.1.3 Real-World Applications

Finally, real-world deployment scenarios add another challenging factor to these challenges. Implementing sophisticated edge computing frameworks in diverse environments, such as public transportation systems, involves overcoming the practical obstacles related to scalability, maintenance, and real-time decision-making. Ensuring the reliability and adaptability of these systems in dynamic and often unpredictable conditions is critical for their success.

1.2 Objectives and Contributions

The research gaps and challenges in edge processing for computer vision are broad and complex. Addressing these issues requires a continuous effort to develop innovative solutions that optimize data integration, computational efficiency, security, and real-world applicability. By tackling these challenges, the field can move closer to realizing the full potential of edge computing in enhancing computer vision applications.

This dissertation aims to address the aforementioned challenges by proposing a novel methodology that resulted in an accelerated multimodal framework tailored for edge computing environments. The key contributions of this research are:

- **Multimodal AI Algorithms**: The core of the framework employs sophisticated AI algorithms, designed to effectively process and interpret multimodal data. To enhance the integration and synchronization of diverse data types, advanced fusion techniques were examined by utilizing lateral connections and other innovative methods. This dissertation designs and optimizes a spectrum of cutting-edge AI techniques implemented for edge deployment and computational offloading, including:
  - **Video Abnormal Event Detection**: Several novel approaches for video abnormal event detection are presented, including pose classification, spatiotemporal autoencoders for anomaly detection and a hybrid convolutional autoencoder with center-weighted loss function suitable for overhead panoramic event detection.

  - **Panoramic Object Detection**: An overhead object detection system tailored for passenger counting in autonomous vehicles (AVs) is developed, leveraging a novel rotation-aware bounding box regression technique to handle the challenges of barrel distortion.
  - **Obstructed Facial Recognition**: A robust facial identification system, based on Siamese neural networks (SNNs) and the ArcFace [18] architecture, is implemented. This system incorporates an attention mechanism and specialized preprocessing techniques to effectively handle occlusions caused by face accessories.

- **Computational Offloading**: Recognizing the computational demands of multimodal data processing, the research exploits specialized hardware accelerators based on FPGAs. Three distinct implementations are presented: (1) a portable VHDL design for color transformation and Sobel edge detection on the Altera DE2-115, (2) an optimized HLS-based design on the Xilinx Pynq-Z1, allowing for direct comparison of VHDL and HLS methodologies. This offered valuable insights into the trade-offs between low-level control and high-level productivity in FPGA design. Furthermore, (3) an accelerated noise reduction design based on an image stacking technique, implemented using HLS in Xilinx Pynq-Z1 that further demonstrating the potential of hardware acceleration for edge preprocessing. Finally, we also designed Some parts of the algorithms to be mapped efficiently and accelerated in CUDA cores.

- **Real-World Validation**: Demonstration of the framework's effectiveness and applicability through rigorous evaluation on real-world datasets and deployment scenarios in the public transportation sector. Both multimodal AI algorithms and computational offloading were extensively utilized at edge devices, showing the success and the practical aspects of our research methodology.

1.3 Document Structure

This dissertation is structured to provide a comprehensive exploration of the proposed framework and its applications. Following this introduction, the remainder of this thesis is organized as follows:

- Chapter 2 provides a holistic review of the underlying algorithms of the proposed framework. It focuses into the theoretical foundations and practical considerations of these algorithms, emphasizing their relevance and applicability to the problem domain.
- Chapter 3 discusses the implementation and evaluation of our research and its framework implementation, highlighting its effectiveness through various use-case scenarios.
- In Chapter 4, the implementation of the proposed accelerated multimodal framework is presented in the field of security, safety and trust in Autonomous Vehicles (AVs).
- In Chapter 5, a comprehensive discussion is offered along with future directions following this thesis. This chapter also presents the thesis's conclusion, offering a summary of its findings and contributions to the field.
- Finally, in Appendix A – Publications, all the published research during this dissertation are presented, classified into three main categories, Journal, Conferences and Book Chapters.

2

Background

The field of Computer Vision plays a crucial role in the advancement of various technologies, particularly in pose estimation, activity recognition, facial recognition and object detection. Pose estimation involves the identification and tracking of human body positions [19], while activity recognition extends this by interpreting complex human actions, contributing significantly to areas, such as surveillance and healthcare monitoring [20]. Facial recognition on the other hand, is crucial in security and personal identification systems and relies heavily on computer vision to accurately identify and verify individuals based on their facial features [21]. Finally, object detection, a fundamental technique for countless artificial intelligence (AI)-driven applications, such as AVs and robotic vision, utilizes computer vision techniques to identify and locate various objects within an image [22]. The integration of computer vision in these areas not only enhances the efficiency and accuracy of systems, but also opens up new possibilities for technological innovation and human-machine synergy.

2.1 Backbone Neural Network Architectures

In this section, we will explore some of the most basic models and algorithms for multiple DL tasks, each with its unique approach in handling the complexities

of spatial and temporal data. Some of the most commonly used models include CNNs, Recurrent Neural Networks (RNNs), Long Short-Term Memory networks (LSTM) and 3D CNNs.

2.1.1 Convolutional Neural Networks

A convolutional neural network, often called CNN or ConvNet, is a specialized type of artificial neural network designed to process data with grid-like structures like 2D images. It is the foundation of the algorithms inspired by the structure and function of the brain. The core building block of a CNN is the convolutional layer which applies filters to the input data, a process called convolution. These filters are learned during the training process, adjusting themselves to best capture the essential characteristics of the data. After the convolutional layers, CNNs often include pooling layers. These layers reduce the spatial size of the data, keeping only the most salient information while discarding unnecessary details. This not only makes the computation more efficient but also helps prevent overfitting, where the model learns to memorize the training data too closely. The final layers of a CNN are typically fully connected layers, similar to those in regular neural networks. These layers combine the extracted features from the previous layers and make the final decisions, such as classifying an image into different categories. A well-known example of a CNN is the Visual Geometry Group (VGG) networks [23] (Figure 2.1).

Thanks to their ability to automatically learn hierarchical features from raw data, CNNs have revolutionized the field of computer vision, achieving state-of-the-art performance in tasks like image recognition, object detection, and image segmentation. They have also found applications in other areas, including natural language processing and time series analysis.

2.1.2 Recurrent Neural Networks

Recurrent Neural Networks (RNNs) are a class of artificial neural networks specifically designed for processing sequences of data. Unlike traditional feed-forward neural networks, which assume that inputs and outputs are independent of each other, RNNs leverage the sequential nature of data by maintaining a hidden state that captures information about previous inputs. This unique

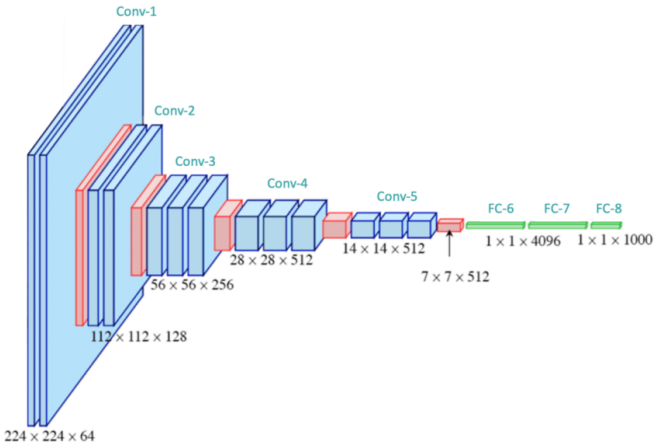

Figure 2.1: This diagram illustrates the VGG-16 CNN architecture [24]. The network consists of 13 convolutional layers (Conv-1 to Conv-5) followed by 3 fully connected layers (FC-6 to FC-8). The convolutional layers use varying kernel sizes and depth to extract features from input images of size $224 \times 224 \times 64$, reducing their spatial dimensions while increasing the number of channels. The fully connected layers transform the extracted features into a 1000-dimensional vector for classification. The dimensions of the feature maps at each stage are indicated, highlighting the progression from the input layer to the output layer.

ability allows RNNs to exhibit temporal dynamic behavior, making them particularly well-suited for tasks where context or historical information is crucial, such as language modeling, speech recognition, and time-series forecasting. At their basic architecture, RNNs are comprised of three layers, namely an input layer, a hidden layer and an output layer. The input layer simply receivers the input sequence at each time step t. The hidden layer maintains the hidden state, which is updated based on the current input and the previous hidden state and can be expressed as follows [25]:

$$h_t^l = f(T_{n,n}, h_t^{(l-1)} + T_{n,n}, h_{(t-1)}^l) \tag{2.1}$$

where $h_t^l \in \mathbb{R}^n$ is a hidden state in layer l at timestep t, $f \in \{\sigma, \tanh\}$ is a non-linear activation function (like tanh or ReLU), and $T_{n,n} : \mathbb{R}^n \to \mathbb{R}^m$ is an affine transform.

Finally, the output layer generates the output, which could be a classification or a regression depending on the task [26], and can be accordingly represented as follows:

$$Y_t = g(W_{hy} H_t + b_y) \tag{2.2}$$

where g is typically a softmax function for classification tasks, W_{hy} is the weight matrix for hidden-to-output connections, and b_y is the bias.

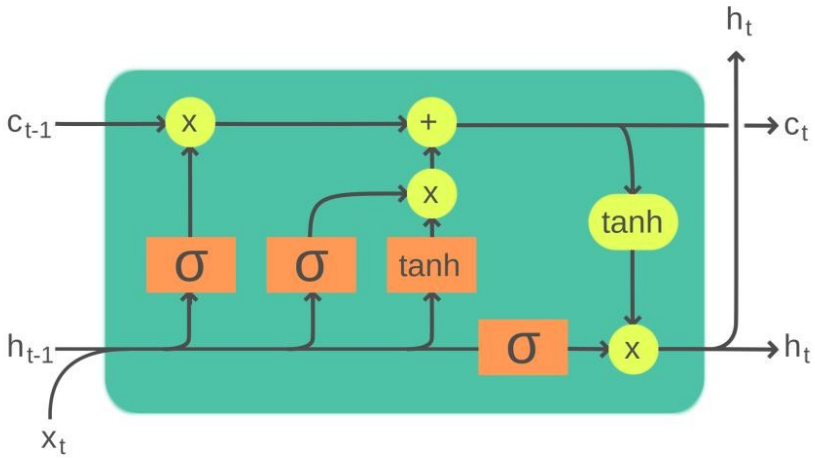

Figure 2.2: This diagram depicts the architecture of a LSTM cell, which is a type of RNN used to capture long-term dependencies in sequential data. The LSTM cell consists of several components that regulate the flow of information through the cell: input gate, forget gate, and output gate, which are represented by sigmoid functions. Additionally, there is a tanh activation function that creates new candidate values.

Despite their advantages, RNNs face significant challenges, primarily due to issues such as vanishing and exploding gradients during training. These problems hinder the network's ability to learn long-term dependencies effectively. To mitigate these issues, advanced variants of RNNs, such as Long Short-Term Memory (LSTM) networks and Gated Recurrent Units (GRUs) [27], have been developed. These architectures introduce gating mechanisms that control the flow of information, allowing the network to better manage long-range dependencies and improve performance on complex sequence-based tasks.

2.1.3 Long Short-Term Memory Networks

LSTM networks, a special kind of RNNs have the ability to process and make predictions based on time-series data. This was the basis upon which LSTMs were designed, as traditional RNNs struggle with the vanishing gradient problem, where the contribution of information decays geometrically over time, making it challenging to connect long-range temporal dependencies within the data. The core innovation of LSTMs is the cell state and its associated gating mechanisms, which regulate the flow of information. Specifically, LSTMs employ three gates—input, forget, and output gates—to control the addition and removal of information from the cell state (Figure 2.2). This allows LSTMs to preserve

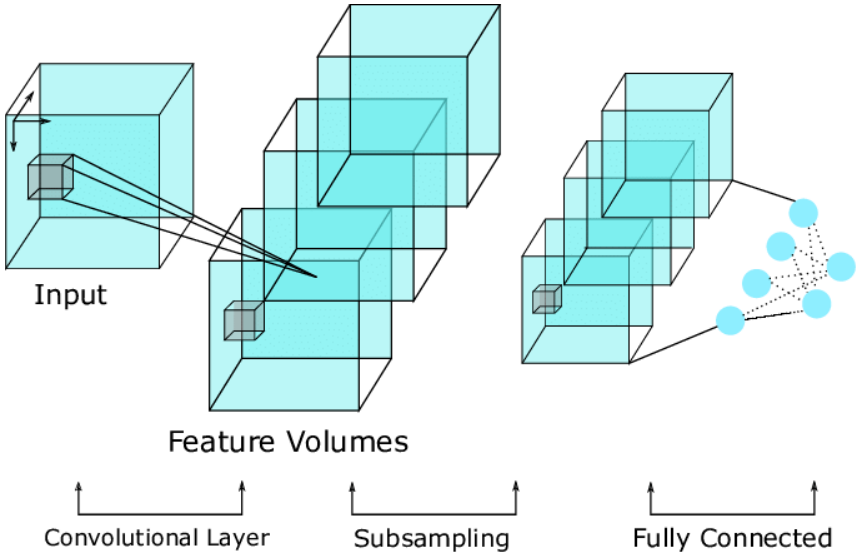

Figure 2.3: This diagram illustrates the architecture of a CNN. It begins with the input layer, where raw data are fed into the network. The data then pass through convolutional layers that extract feature volumes, followed by subsampling (pooling) layers that reduce the spatial dimensions of the feature maps. Finally, the feature volumes are passed through fully connected layers to perform classification or regression tasks. The diagram highlights the transition from feature extraction to decision making within the network.

relevant information while discarding irrelevant data, thereby maintaining a balance between short-term and long-term memory. As a result, LSTMs have become the backbone of many advanced applications in natural language processing, speech recognition, and time-series forecasting, where understanding context over time is crucial.

2.1.4 3D Convolutional Neural Networks

3D CNNs have been introduced as a powerful tool in computer vision, especially for activity recognition tasks and volumetric image analysis. In traditional 2D CNNs, convolutions are performed on 2D spatial data, which renders them suboptimal for video data as they fail to capture the temporal dynamics. 3D CNNs extend the concept of convolution to three dimensions, allowing the network to learn features across both space (width and height of the frame) and time (sequential frames) by processing a video clip as a 3D volume, where the third dimension is time [28]. This is crucial for activity recognition, where understanding the sequence of movements is as important as recognizing the spatial features.

Regarding their architecture, 3D CNNs are comprised of five distinct layers,

namely the input layer, the convolutional layers, the pooling layers, the fully connected layers and the output layer (Figure 2.3). The input layer refers to the input data in the network that are typically a sequence of video frames, representing a short clip, where each frame is a 2D image, and the sequence forms a 3D volume. The next layers are the convolutional layers that are responsible for performing 3D convolution, by capturing both spatial and temporal features. This way, the network learns filters that respond to specific types of movement or visual patterns over time. Similar to 2D CNNs, 3D CNNs also use pooling layers to reduce the spatial dimensions (width and height) and the temporal dimension, which helps in reducing the computational complexity and overfitting. Finally, after several convolutional and pooling layers, the network uses fully connected layers to perform high-level reasoning based on the learned features and, similarly to 2D CNNs, the final layer is typically a softmax layer for classification tasks, where each neuron corresponds to a potential activity class.

2.2 Pose Estimation

Pose estimation is the process of determining the position and orientation of objects or humans within images or videos [29]. This capability is not just a significant technological achievement, but also a key component in a wide array of applications, ranging from interactive gaming and AR to healthcare and autonomous driving. At its core, pose estimation is about understanding the spatial configuration of objects or subjects in visual data, a task that has evolved significantly with advances in technology and methodology [30].

Pose estimation first appeared when the fundamental need to interpret and understand visual data began. Early efforts in pose estimation were primarily focused on simple geometric models for extracting the position and orientation of objects in images [31]. These methods were limited in their ability to handle complex, real-world scenarios.

As computer vision evolved, the field witnessed a significant shift with the evolution of ML, which introduced more sophisticated, data-driven approaches [32]. Those approaches introduced part-based models for human pose estimation, which involved breaking down the human figure into individual parts and estimating their positions. These methods provided greater accuracy, but were still constrained by the computational complexities and the quality of available data.

The real transformation in pose estimation, however, came with the integration of DL. The evolution of CNNs and subsequent DL architectures dramatically enhanced the ability to interpret complex poses, even in dynamic and unstructured environments [33]. This progress was accelerated by the availability of large annotated datasets, such as the Human3.6M dataset for 3D human poses and significant advances in computational power [34].

2.2.1 Pose Estimation Methodologies

As mentioned above, pose estimation involves determining the spatial configuration of objects or human figures from visual data, with its methodologies having been significantly evolved over the years, transitioning from traditional geometric models to advanced DL techniques. However, two are the main methodologies used in pose estimation tasks, namely 2D and 3D pose estimation:

- **2D Pose Estimation**: It involves the detection of parts position, such as human joints, in a two-dimensional plane. This methodology is ideal in applications where depth information is secondary or unnecessary. The initial approaches in 2D pose estimation were grounded in basic image processing techniques, which included methods like outline analysis and color segmentation, being relatively simple and limited in their effectiveness, especially in complex scenes. However, with advancements in ML, part-based models began to gain ground. One such example is the Pictorial Structures model [35], which represented the human body as a collection of interconnected parts. These models utilized probabilistic frameworks to estimate the pose but were often constrained by high computational requirements.

- **3D Pose Estimation**: 3D pose estimation extends the task to three dimensions, providing depth information essential in various applications, from AR to bio-mechanical analysis. However, transitioning from 2D to 3D pose estimation introduces additional complexities. While 2D estimation focuses on x and y coordinates on the image plane, 3D pose estimation involves determining the z coordinate, which represents the depth. Early methods of 3D pose estimation relied on multi-camera setups or depth sensors, such as LiDAR and RGB-D cameras, to obtain depth information.

2.2.2 Key Algorithms and Models

In this section, an overview of some key algorithms and models that are most commonly used in pose estimation tasks are presented, including OpenPose [36], AlphaPose [37] and SimpleBaseline [38] for 3D Human Pose Estimation.

2.2.2.1 OpenPose

OpenPose, a revolutionary technology in the field of computer vision, operates through a complex pipeline that leverages the power of CNNs. The initial stage involves feeding an image or video frame into a CNN, which acts as a feature extractor. This network is trained on datasets of annotated images, learning to identify patterns and structures that correspond to human body parts. The extracted features are then passed to a second CNN, known as the Part Affinity Fields (PAF) network. The PAF network predicts the association between body parts, essentially creating a map of how different keypoints are connected. This stage is crucial in determining the pose of multiple individuals within a single image or frame, as it allows the algorithm to differentiate between limbs that belong to different people.

OpenPose is designed to identify and track multiple human figures in real time, providing keypoint coordinates for body parts, including limbs and facial features. The system is built upon CNNs and introduces the concept of PAFs, a novel approach to solve the problem of part-to-person association in multi-person pose estimation [19]. As a result, OpenPose consists of two main components, where the first one is used to predict confidence maps for body part locations, while the second part is used to predict PAFs that are essential for part association.

More specifically, confidence maps are generated to represent the likelihood of a body part's presence at each pixel. For each person k, a confidence map S_k is generated. If $x_{(j,k)} \in \mathbb{R}^2$ is the ground truth position of body part j for person k in the image, the value at location $p \in \mathbb{R}^2$ in $S^*_{(j,k)}$ is defined as follows [19]:

$$S^*_{(j,k)}(p) = \exp\left(-\frac{\|p - x_{(j,k)}\|_2^2}{\sigma^2}\right) \tag{2.3}$$

where σ is a parameter that controls the spread of the peak.

The next step in the OpenPose pipeline is the bipartite matching algorithm.

Figure 2.4: The concept of keypoints in pose estimation, showing the key joints and their connections in human figures performing different poses. Each silhouette represents a person, with keypoints marked at critical joints such as the head, shoulders, elbows, wrists, hips, knees, and ankles. The lines connecting these keypoints form a skeletal structure, highlighting the relationships between different parts of the body.

This sophisticated process takes the PAF predictions and associates them with the keypoint detections from the first CNN. By evaluating the confidence scores of each connection, the algorithm constructs a skeletal representation of each person in the image, effectively "connecting the dots" between the detected keypoints (Figure 2.4). The final stage of the OpenPose pipeline involves temporal filtering. This step is particularly relevant when dealing with video input, as it helps to smooth out any jittery or inconsistent pose estimations over time. By considering the pose information from previous frames, the algorithm can refine its predictions, leading to more accurate and stable tracking of human movement.

OpenPose is not limited to a single model but offers a variety of pre-trained models with varying levels of accuracy and computational complexity. This flexibility allows users to choose the model that best suits their specific application, whether it's real-time pose estimation on a low-powered device or high-precision tracking in a controlled environment. The technology is not confined to the detection of human body parts but has been extended to estimate the poses of hands, faces, and even feet. This multi-person, multi-part capability opens up a wide range of possibilities for applications in fields such as sports analysis, augmented reality (AR), and human-computer interaction (HMI).

2.2.2.2 AlphaPose

AlphaPose [39] is designed for real-time, multi-person pose estimation. It is well known for its ability to maintain high accuracy even in challenging scenarios with multiple individuals and potential occlusions. The model is structured to first detect individuals in a scene and then estimate their poses separately, ensuring precision in pose estimation. The core of the AlphaPose algorithm consists of two primary stages, the human detection stage, where the individual humans in a given image or video frame are identified, and the pose estimation stage, where the pose of each detected human is estimated [37].

As mentioned above, AlphaPose begins with a human detection module, often employing a pre-trained object detection model, such as You Only Look Once (YOLO) [40] or Faster Region-based CNN (Faster R-CNN) [41]. The object detection framework is used to identify human bounding boxes within an image and can be represented as $D(I) \rightarrow \{B_i\}$, where I is the input image and $\{B_i\}$ is the set of detected human bounding boxes.

Once individual humans are detected, AlphaPose estimates the pose for each person within the bounding box, firstly, by using a Single Person Pose Estimator (SPPE) that predicts a set of keypoints representing the human pose for each bounding box B_i. The SPPE function generates heatmaps for each keypoint, indicating the likelihood of each keypoint's position. If $h_k(x, y)$ represents the heatmap for keypoint k at position (x, y), the predicted position of the keypoint is the one with the highest value in the heatmap.

To handle inaccuracies in human detection and improve pose estimation, AlphaPose introduces the Regional Multi-Person Pose Estimation (RMPE) framework [39], which utilizes Spatial Transformer Networks (STNs) [42] to transform the detected bounding box into a better-fitting one. If $T(B_i, \theta) \rightarrow B_i'$ represents the STN operation, B_i' is the transformed bounding box, and θ are the transformation parameters learned to optimize the fit of the bounding box. To further refine the bounding box, AlphaPose also uses a Symmetric Spatial Transformer Network (SSTN) that is designed to be symmetric to ensure that the transformation is learnable and efficient [39].

Finally, to handle overlapping bounding boxes and keypoints, AlphaPose implements a parametric pose Non-Maximum Suppression (NMS) mechanism, which can be represented as NMS$(\{K_i\}) \rightarrow \{K_i'\}$, where $\{K_i'\}$ is the set of non-overlapping keypoints after the NMS operation.

2.2.2.3 SimpleBaseline for 3D Pose Estimation

SimpleBaseline [43] for 3D human pose estimation represents a shift towards simplification in the pose estimation domain, proving that high performance can be achieved without overly complex network architectures. It is a DL model designed for 3D human pose estimation that was developed in response to the trend of increasingly complex models in this field, demonstrating that a relatively uncomplicated architecture can offer high accuracy. The model operates by upscaling 2D keypoints detected in images to 3D space, leveraging deep neural networks for this purpose.

The architecture of SimpleBaseline is centered around a standard ResNet backbone followed by several deconvolutional layers [43]. This structure is simpler compared to other models that use more complex architectures and auxiliary branches. The Residual Network (ResNet) [44] backbone in SimpleBaseline is used for feature extraction and is characterized by its residual connections, which help in mitigating the vanishing gradient problem in deep networks. A standard ResNet [44] model is generally represented as:

$$y = F(x, \{W_i\}) + x \tag{2.4}$$

where y is the output of the residual block, F is the residual function, x is the input feature, and $\{W_i\}$ are the weights.

After feature extraction, the model employs a series of deconvolutional layers to gradually upsample the feature maps, helping in refining the spatial resolution for accurate keypoint localization. To perform 2D-to-3D lifting, SimpleBaseline uses a linear regression model that takes 2D keypoints as input and estimates their corresponding 3D positions [45]. The model does not directly process image data for 3D pose estimation. Instead, it relies on an efficient transformation from 2D to 3D using a transformation equation. If P_{2D} represents the 2D keypoints and P_{3D} represents the predicted 3D keypoints, the transformation can be modeled as [39]:

$$P_{3D} = T(P_{2D}; W_T) \tag{2.5}$$

Here, T is the transformation function parameterized by weights W_T, which are learned during training.

Finally, the training of the SimpleBaseline model is focused on accurately mapping 2D keypoints to 3D space. The model is trained using a dataset of

images with corresponding 2D and 3D keypoint annotations. The primary loss function used in SimpleBaseline is the mean squared error (MSE) between the predicted 3D keypoints and the ground truth [43], which is calculated by:

$$L = \frac{1}{N} \sum_{i=1}^{N} \left\| P_{3D}^{(i)} - \hat{P}_{3D}^{(i)} \right\|^2 \tag{2.6}$$

where N is the number of keypoints, $P_{3D}^{(i)}$ is the predicted 3D position of the i-th keypoint, and $\hat{P}_{3D}^{(i)}$ is the ground truth.

2.3 Activity Recognition

Activity recognition involves the identification and classification of actions or behaviors depicted in images and videos and has gained interest due to its wide array of applications, from surveillance to healthcare and sports analytics [20]. The task of recognizing activities within visual media is challenging due to the complexity and variability of human actions and the environments in which they occur [46].

Activity recognition firstly began with the emergence of motion detection and has since evolved into the complex task of understanding complex human behaviors and interactions [47]. Early efforts relied on simple temporal changes in pixel values to detect movement. However, these approaches lacked the sophistication to understand the context or type of activity [48].

With the advancement of ML and computer vision, more complex models were developed that included handcrafted features, such as Scale-Invariant Feature Transform (SIFT) [49] and Histogram of Oriented Gradients (HOG) [50], which provided more detailed information about shapes and movements in videos [51].

However, DL was the real transformative change in activity recognition. CNNs [52], RNNs [53], and more recently, 3D Convolutional Networks (3D CNNs) [54] have enabled more accurate and robust activity recognition by learning feature representations directly from data.

2.3.1 Activity Recognition Methodologies

Initially, activity recognition primarily relied on extracting and analyzing handcrafted features from images or videos to capture relevant information about motion and appearance, which could then be used to classify different activities. In this context, two main categories of feature-based methods were established, namely the optical flow method and other handcrafted spatial features methods, including HOG [50] and SIFT [49]. The first one method, optical flow, involves calculating the motion between two consecutive frames based on pixel intensity changes. The optical flow vectors can indicate the direction and speed of object movement, as presented in the following equation, which are crucial for recognizing certain activities. In Equation 2.7, V is the optical flow at pixel (x, y) at time t, with u_x and u_y representing the flow in horizontal and vertical directions, respectively.

$$V(x, y, t) = (u_x, u_y) \tag{2.7}$$

However, the evolution of DL brought an evolution in activity recognition, introducing models that could learn feature representations directly from data, leading to improved accuracy and generalization over traditional, feature-based approaches. The key advantage in these approaches is the ability of DL models, especially CNNs and RNNs, to automatically learn and extract features directly from large volumes of data. This capability enables the identification of complex and abstract patterns that are crucial to recognizing a wide array of human activities. DL models excel in handling spatiotemporal data, a critical aspect of activity recognition. CNNs effectively process spatial information within individual video frames, by identifying patterns and movements indicative of specific activities.

2.3.2 Key Algorithms and Models

This section focuses into several state-of-the-art models on the field of activity recognition in videos, each contributing unique innovations and improvements over traditional approaches. The following methodologies leverage deep learning techniques to automatically learn and extract meaningful features from video data, thereby significantly enhancing the accuracy and efficiency of activity

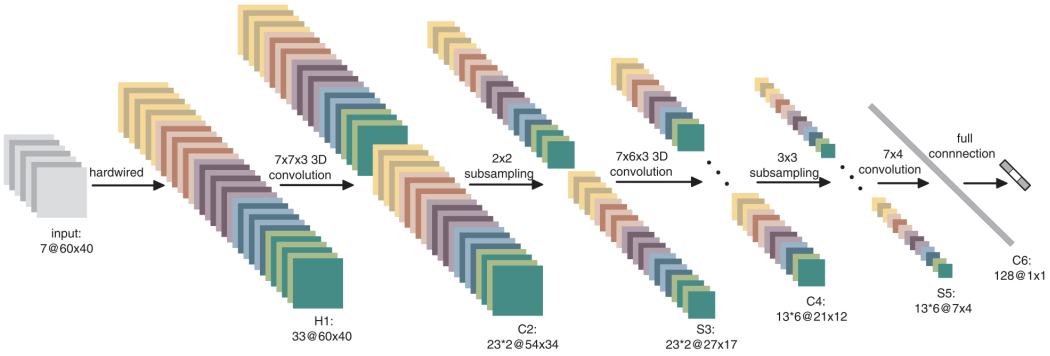

Figure 2.5: A 3D CNN architecture for human action recognition [55]. This architecture consists of one hardwired layer, three convolution layers, two subsampling layers, and one full connection layer.

recognition systems.

2.3.2.1 Convolutional 3D Networks

C3D, or Convolutional 3D networks, are among the fundamental models that extend the principles of 2D convolution to the temporal domain by applying 3D convolutions. This model effectively captures spatiotemporal features directly from the raw video data, making it a foundational approach for many subsequent advancements in video activity recognition. The C3D model operates by applying 3D convolutional filters over a sequence of video frames, rather than treating each frame independently. This enables the model to learn spatial features within each frame and temporal features across consecutive frames simultaneously. Specifically, the 3D convolutional layers are designed to capture both the spatial structure of objects and their motion patterns over time, providing a more holistic understanding of the video content. The architecture of C3D typically involves several layers of 3D convolutions, followed by pooling layers that reduce the spatial and temporal dimensions while preserving essential features. By stacking multiple layers, the network can progressively abstract higher-level spatiotemporal patterns. For example, early layers might detect simple motion and edges, while deeper layers can recognize complex activities and interactions (Figure 2.5).

One of the significant advantages of C3D networks is their ability to process and learn from raw video data without the need for hand-crafted features or extensive pre-processing. This end-to-end learning capability makes C3D models

highly adaptable to various video recognition tasks, such as action recognition, gesture detection, and video classification.

However, C3D networks are computationally intensive due to the high dimensionality of 3D convolutions. This has led to the exploration of more efficient architectures, such as I3D (Inflated 3D ConvNet) and X3D, which aim to reduce the computational cost while maintaining or improving recognition performance.

2.3.2.2 Temporal Segment Networks

Temporal Segment Networks (TSNs) [56] introduce a framework that divides video data into segments, extracting features from each segment and aggregating them to represent the entire video. This method allows for the efficient handling of long video sequences, ensuring that temporal dynamics and context are accurately captured. TSN is effective in various video understanding tasks, including action recognition and temporal action detection.

TSN operates by breaking down a video into a series of uniformly sampled segments. For each segment, a frame or a short snippet is selected, and features are extracted using convolutional neural networks (CNNs). This process captures both spatial and short-term temporal features within each segment. Once the features from all segments are extracted, they are aggregated to form a comprehensive representation of the entire video. This aggregation can be performed through various methods, such as average pooling or weighted sum, to ensure that both global and local temporal information is considered (Figure 2.6). The segmentation approach of TSN addresses the challenge of long video sequences by focusing on representative snippets rather than processing every frame. This significantly reduces computational complexity and memory requirements, making TSN suitable for real-time applications and large-scale video datasets. Moreover, by sampling from different parts of the video, TSN ensures that the model captures diverse temporal dynamics, such as the beginning, middle, and end of an action, providing a more complete understanding of the activity.

One of the key advantages of TSN is its flexibility in incorporating different types of CNN architectures for feature extraction. This modularity allows TSN to leverage the advancements in image recognition models, such as ResNet [44] and VGG [23], adapting them for video analysis. Additionally, TSN can be

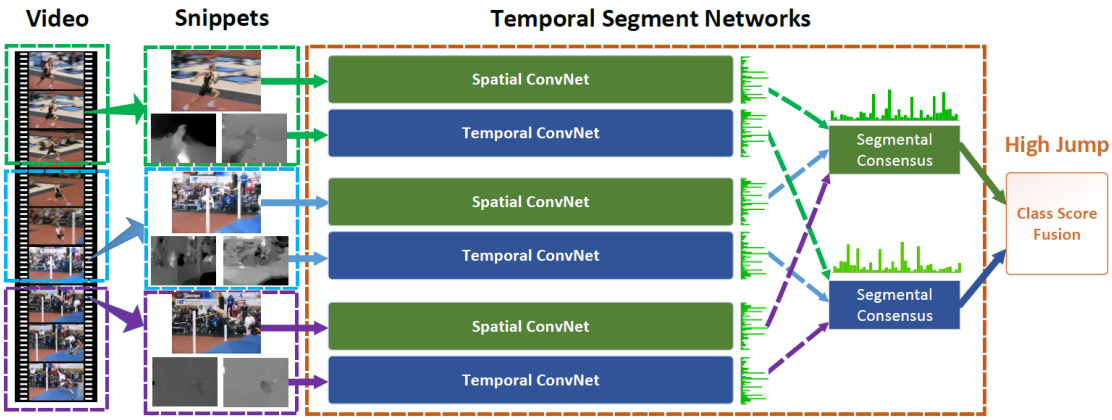

Figure 2.6: An overview of TSN for human action recognition. The video is divided into snippets, which are sampled uniformly from the entire video. Each snippet is processed by a Spatial ConvNet and a Temporal ConvNet to extract spatial and temporal features, respectively. These features are then aggregated through segmental consensus, which combines the segment-level predictions into a final prediction.

extended to multi-modal inputs, integrating features from RGB frames, optical flow, and even audio, to enhance the robustness of video understanding.

Despite its strengths, TSN also has limitations, particularly in capturing long-range dependencies and fine-grained temporal details. To address these challenges, researchers have explored hybrid approaches that combine TSN with recurrent neural networks (RNNs) or attention mechanisms, aiming to enhance temporal modeling capabilities without sacrificing efficiency.

2.3.2.3 Temporal Shift Module

The Temporal Shift Module (TSM) [57] is designed to enhance the temporal modeling capability of 2D convolutional networks without significantly increasing computational complexity. TSM shifts a portion of the feature map along the temporal dimension, enabling the network to capture temporal relationships effectively. This approach maintains a balance between performance and efficiency, making it suitable for real-time applications. TSM introduces a simple yet powerful mechanism to incorporate temporal information into 2D CNNs, which are typically limited to processing spatial information within individual frames. By shifting part of the feature map along the temporal axis, TSM allows adjacent frames to share information, thereby capturing motion and temporal dependencies between consecutive frames. This temporal shift is implemented in a way that does not require additional parameters or significant computa-

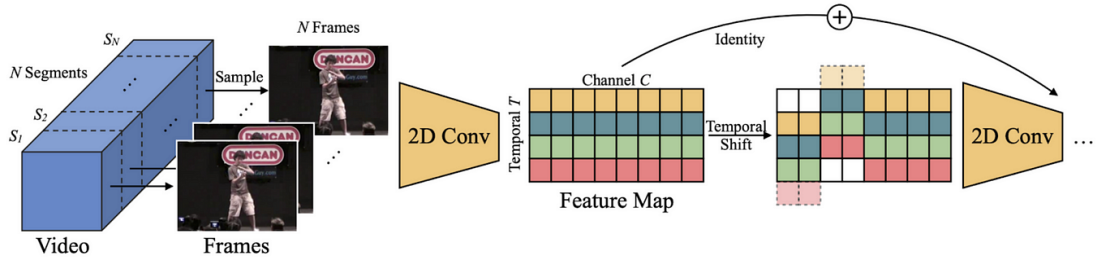

Figure 2.7: The TSM for video recognition. The video is divided into N segments, from which frames are sampled for processing. These frames are passed through a Conv2D network to extract spatial features, resulting in a feature map. The TSM enhances temporal modeling by shifting a portion of the feature map along the temporal dimension, mixing information across adjacent frames. The shifted features are combined with the identity mapping to preserve spatial information, followed by further 2D convolutions.

tional overhead, maintaining the efficiency of standard 2D convolutions.

The TSM architecture is built upon a typical 2D CNN backbone, such as ResNet [44] or MobileNet [58]. During the feature extraction process, a portion of the intermediate feature maps is temporally shifted forward and backward. For example, one-third of the channels are shifted to the previous frame, one-third to the next frame, and the remaining channels stay in the current frame. This temporal shift effectively mixes the features across time, enabling the network to model temporal relationships while preserving spatial feature extraction (Figure 2.7).

One of the key advantages of TSM is its ability to be seamlessly integrated into existing 2D CNN architectures. This makes it an attractive solution for enhancing temporal modeling without the need for redesigning the network architecture or significantly increasing the computational cost. Consequently, TSM can be applied to a wide range of video analysis tasks, including action recognition, gesture detection, and video classification, with minimal impact on inference speed.

Despite its simplicity, TSM is ideal for further research and enhancements. Researchers have explored combining TSM with other temporal modeling techniques, such as attention mechanisms and recurrent neural networks (RNNs), to capture more complex temporal dependencies and long-range interactions. These hybrid approaches aim to build upon the strengths of TSM while addressing its limitations in modeling extended temporal contexts.

2.3.2.4 SlowFast Networks

SlowFast Networks [59] have revolutionized video recognition by leveraging the temporal dynamics of video sequences. This approach utilizes two separate pathways: a slow pathway that captures semantic information over a longer duration and a fast pathway that focuses on rapid, fine-grained temporal changes. The knowledge sharing between these pathways allows for a more accurate and efficient analysis of video content, addressing the need for both temporal resolution and context. The architecture of SlowFast networks is designed to simultaneously process video data at different temporal resolutions. The slow pathway operates at a lower frame rate, providing a detailed and semantically rich representation of the video over an extended period. This pathway captures essential context and slower-moving objects, ensuring that the model understands the broader scene and activities. In contrast, the fast pathway processes video at a higher frame rate, focusing on capturing quick motions and fine-grained temporal details that might be lost by the slow pathway.

The interaction between these two pathways is facilitated through lateral connections, allowing the exchange of information and enhancing the overall understanding of the video. The slow pathway benefits from the high-frequency temporal details provided by the fast pathway, while the fast pathway gains contextual information from the slow pathway. This bidirectional flow of information ensures that the model can accurately capture and integrate both short-term and long-term temporal dynamics (Figure 2.8).

One of the significant advantages of SlowFast networks is their ability to balance computational efficiency with high performance. By distributing the computational load across two pathways operating at different temporal resolutions, the model can achieve a more comprehensive analysis without a substantial increase in complexity. This design makes SlowFast networks particularly effective for complex video recognition tasks, such as action detection, gesture recognition, and video classification, where both detailed motion and contextual understanding are crucial. In addition to their performance benefits, SlowFast networks are also highly adaptable. They can be integrated with different backbone architectures, such as ResNet [44], and can be extended to incorporate multi-modal inputs, including RGB frames, optical flow, and even audio signals. This flexibility allows SlowFast networks to be tailored to specific applications and data modalities, enhancing their utility across a wide range of video analysis tasks.

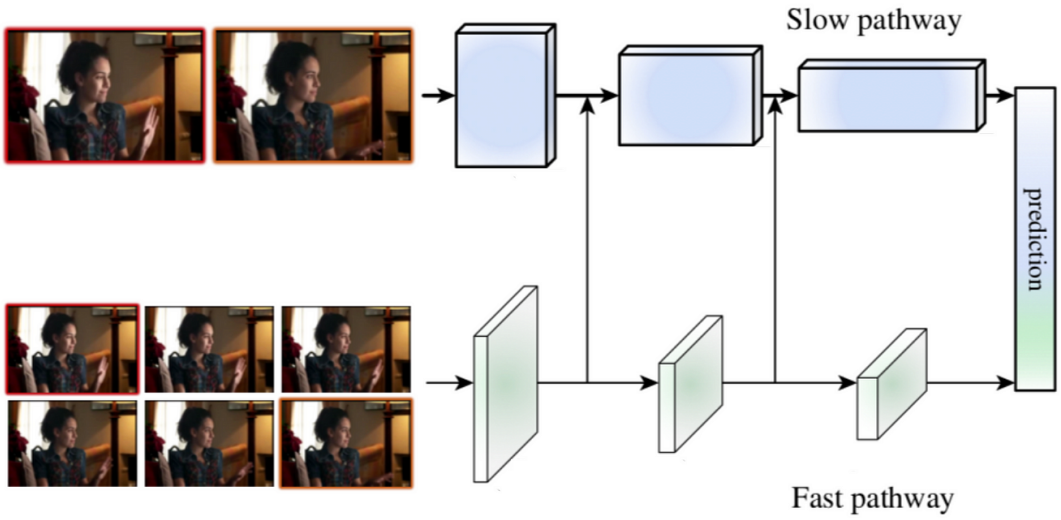

Figure 2.8: The architecture of SlowFast networks for video recognition. The video frames are processed through two separate pathways: the slow pathway and the fast pathway. The slow pathway (top) captures semantic information over a longer duration by processing fewer frames at a lower frame rate, providing a detailed and contextual understanding of the video. The fast pathway (bottom) captures rapid, fine-grained temporal changes by processing more frames at a higher frame rate, focusing on the dynamic aspects of the video. Information from both pathways is combined to make the final prediction.

Despite their advantages, SlowFast networks do have limitations, particularly in terms of the increased complexity of managing and synchronizing multiple pathways. Ongoing research aims to address these challenges by exploring more efficient ways to implement multi-pathway architectures and improving the integration of temporal information across different scales.

2.3.2.5 Expand, Excite, Extend, Depthwise Separable 3D Convolutions

X3D models [60], known for their Expand, Excite, Extend, and Depthwise Separable 3D Convolutions, represent a significant advancement in scaling video models. These models introduce a scalable approach to building 3D convolutional networks by expanding the network capacity, exciting relevant features, and extending temporal receptive fields. The use of depthwise separable convolutions further enhances efficiency, allowing for deeper and more expressive models without prohibitive computational costs.

X3D models aim to address the challenges of developing efficient yet powerful 3D convolutional networks for video recognition tasks. The key innovation of X3D lies in its progressive scaling strategy, which systematically increases the network's capacity and complexity to balance performance and computational

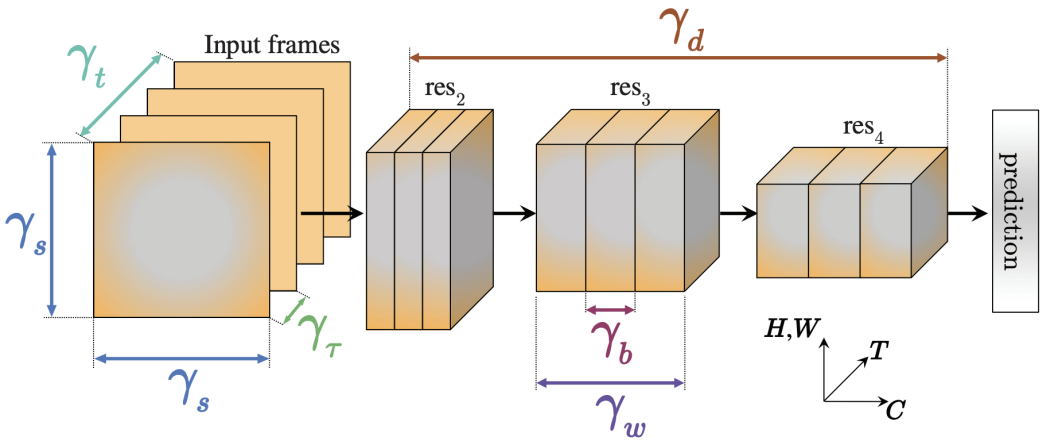

Figure 2.9: X3D networks progressively expand a 2D network across the following axes: temporal duration, frame rate, spatial resolution width, bottleneck width, and depth [61]. The input frames are processed through these expanded dimensions, resulting in a scalable network that captures detailed spatiotemporal features while maintaining computational efficiency.

efficiency. This is achieved through three main processes: Expand, Excite, and Extend (Figure 2.9).

Firstly, the "Expand" phase involves increasing the network's width, depth, and resolution to enhance its capacity to capture detailed spatiotemporal features. By carefully adjusting these dimensions, X3D models can scale from smaller, more efficient networks to larger, more powerful ones, depending on the specific requirements of the task and the available computational resources. This flexibility allows X3D to be tailored for various deployment scenarios, from mobile devices to high-performance servers.

Secondly, the "Excite" phase focuses on amplifying the most relevant features within the network. This is achieved using channel-wise attention mechanisms that dynamically adjust the importance of different feature channels based on the input video data. By emphasizing the most informative features, X3D models improve their ability to discriminate between different actions and activities, leading to higher accuracy in video recognition tasks.

The "Extend" phase involves expanding the temporal receptive fields of the network to capture long-range dependencies and temporal dynamics more effectively. By increasing the temporal dimension of the 3D convolutions, X3D models can analyze longer video sequences and understand complex temporal relationships. This extension is crucial for tasks that require a deep understanding of temporal context, such as action detection and event recognition.

One of the key components that enable the efficiency of X3D models is the use of depthwise separable 3D convolutions. This technique decomposes standard

3D convolutions into separate spatial and temporal convolutions, significantly reducing the number of parameters and computational operations required. As a result, X3D models can achieve deeper and more expressive architectures without incurring prohibitive computational costs, making them suitable for real-time applications. In addition to their performance benefits, X3D models are highly adaptable and can be integrated with other advanced techniques, such as multi-modal inputs and attention mechanisms. This adaptability enhances their robustness and versatility, enabling them to address a wide range of video understanding challenges.

Despite their advantages, X3D models also present opportunities for further optimization, particularly in terms of reducing computational overhead and improving scalability. Ongoing research aims to refine the scaling strategies and incorporate more sophisticated attention mechanisms to enhance the efficiency and effectiveness of X3D architectures.

2.3.2.6 Mobile Video Networks

As video processing increasingly moves towards edge devices, MoViNets (Mobile Video Networks) [62] have become critical. These networks are designed to be highly efficient, with a focus on reducing computational complexity while maintaining high accuracy. MoViNets achieve this through innovative architectures that balance performance and resource constraints, making real-time video analysis feasible on portable devices. MoViNets are specifically implemented to address the unique challenges of mobile and edge computing environments, where computational resources, power consumption, and latency are critical constraints. Traditional video recognition models, although highly accurate, often require substantial computational power and memory, making them unsuitable for deployment on resource-limited devices. MoViNets overcome these limitations by employing several key strategies.

Firstly, MoViNets utilize lightweight network architectures that are optimized for efficiency. This involves designing convolutional layers and other network components to minimize the number of parameters and computational operations. Techniques such as depthwise separable convolutions, which separate spatial and channel-wise operations, significantly reduce the computational load while preserving the model's ability to learn complex features.

Secondly, MoViNets incorporate efficient temporal modeling techniques to

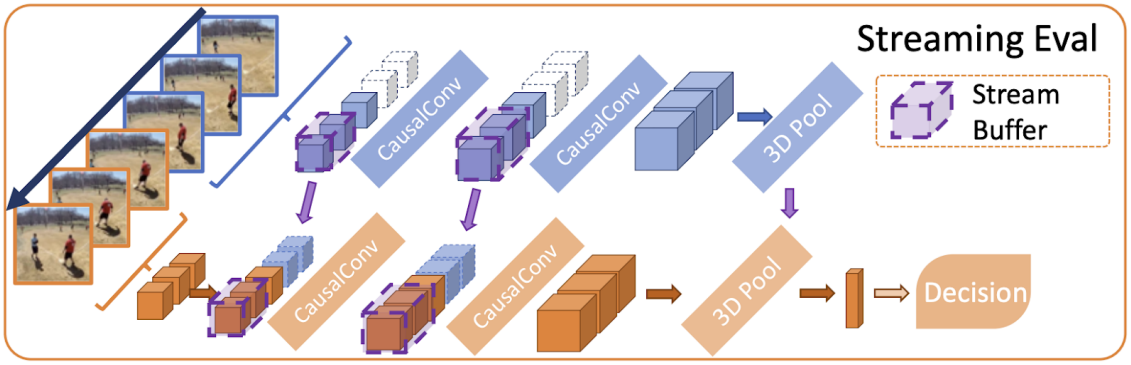

Figure 2.10: The architecture of streaming evaluation for video recognition using causal convolutions [62]. The video frames are processed in a streaming manner, with frames being sampled over time. Each sampled frame undergoes causal convolution (CausalConv), which ensures that only past and present information is used, preventing future information leakage. The intermediate feature maps are then pooled using 3D pooling (3D Pool) to reduce dimensionality and aggregate spatial-temporal features. This process continues with new frames, and the buffered feature maps are combined to form a final decision. The stream buffer stores intermediate states to facilitate continuous evaluation, enabling efficient real-time video analysis and decision-making.

capture motion and temporal dynamics in video data without excessive computational overhead. By strategically sampling frames and using temporal pooling methods, MoViNets can effectively analyze video sequences with fewer frames, reducing the amount of data that needs to be processed. This approach maintains temporal resolution and context while ensuring the model remains lightweight (Figure 2.10).

Another critical aspect of MoViNets is their ability to adapt to varying levels of computational resources available on different devices. This adaptability is achieved through techniques such as model scaling, where the network's depth, width, and resolution can be adjusted based on the target device's capabilities. For instance, a smaller version of MoViNet might be deployed on a smartphone, while a more extensive version could be used on a more powerful edge server. This scalability ensures that MoViNets can deliver optimal performance across a wide range of hardware configurations.

The design principles of MoViNets have influenced the development of other efficient neural network architectures for mobile and edge computing. Their success underscores the importance of optimizing models not just for accuracy but also for the practical constraints of deployment environments.

2.4 Facial Recognition

Facial recognition technology has been the center of attention in recent years due to its applications in various domains such as security, surveillance, authentication, as well as social media [63]. Facial recognition has a long history, with early attempts dating back to the 1960s, when Woody Bledsoe, Helen Chan Wolf, and Charles Bisson developed one of the first facial recognition systems, which used manual feature extraction and geometric measurements to identify faces in images [64]. However, it was only with the advent of DL that facial recognition made significant advances.

DL, particularly CNNs, has revolutionized the field of computer vision and, by extension, facial recognition, due to their ability to automatically learn and extract intricate features from images [65]. As a result, they are well-suited for facial recognition tasks. The development of deep neural networks, including the VGGNet [23], ResNet [44] and InceptionNet [66] has significantly improved the accuracy and efficiency of facial recognition systems [44].

The facial recognition process begins with capturing a facial image using a camera. However, since this process can occur under various conditions and angles, the complexity of the recognition process can be heavily influenced by it, as well as by the quality, the lighting and the background of the image. Once an image is captured, the system detects the presence of a face and the algorithms identify key facial features, mainly the eyes, nose and mouth. After face detection, the system analyzes the face to extract unique features, which are distinct aspects of the face that can be used to distinguish one person from another. These features may include the distance between the eyes, the shape of the cheekbones and the contour of the lips, among others.

The next step involves faceprint creation, which is a numerical print created by the identified features from the previous step. Since each face has a unique faceprint, similar to a fingerprint, this process creates a feature vector that is a set of numbers that represent these key features. For verification, the faceprint is compared to a specific faceprint stored in the database to confirm if they match. In identification, on the other hand, the faceprint is compared against multiple records to determine the identity among other options. Then, based on the degree of match, the system decides whether the presented face matches a stored faceprint, a process typically carried out by ML-based classifiers that can handle complex decision-making processes.

In summary, the effectiveness of a facial recognition system depends not only on the quality of the algorithms and models used, but also on factors such as the quality of input images, lighting conditions and the presence of occlusions [67]. Additionally, addressing ethical concerns, bias, as well as privacy issues still remains a significant challenge in the development and deployment of facial recognition technology [68].

2.4.1 Facial Recognition Methodologies

Facial recognition in images and video streams involves several key methodologies that leverage DL and computer vision techniques. Each methodology addresses different aspects of the process, from detecting and identifying faces to analyzing and comparing them against known faces.

Before a system can recognize a face, it must first detect its presence in an image or video frame. This can be accomplished through methods such as the Viola-Jones algorithm, or DL-based methods. The Viola-Jones algorithm [69] is one of the earliest and most famous face detection methods that uses Haar-like features and an integral image concept for rapid feature detection, combined with a cascade of classifiers for increased accuracy. On the other hand, modern DL approaches often use CNNs for face detection that can handle more complex and varied facial representations compared to traditional algorithms.

For feature extraction, there are two main types of approaches, namely the geometric feature-based and the appearance-based approaches. The geometric feature-based approaches focus on the geometric properties of facial features, such as the distance between the eyes, the nose width and the jawline shape, whereas the appearance-based approaches analyze the appearance and texture of the face. Techniques including the Local Binary Patterns (LBPs) [70], the HOGs [50] and other neural network-based feature extraction approaches fall under this category.

Deep convolutional networks, and especially CNNs, are mainly used for learning high-level features and representations of faces, since layers in CNNs automatically and hierarchically learn features from raw pixels to complex facial attributes. In addition, autoencoders for dimensionality reduction and Generative Adversarial Networks (GANs) for generating and learning complex facial distributions are also explored in advanced systems.

As mentioned above, after feature extraction, the system compares the de-

tected face with known faces using either template matching or ML classifiers. In simpler systems, facial features are compared with stored templates to find the best match. However, in more complex systems techniques such as Support Vector Machines (SVMs), k-Nearest Neighbors (k-NNs), and DL classifiers are used to classify the facial signature against a database of known faces.

2.4.2 Key Algorithms and Models

Facial recognition tasks leverage a range of algorithms and models, primarily from the fields of ML and computer vision. These models can be mainly classified either as traditional algorithms or DL-based models. For traditional algorithms, one of the earliest techniques used for face recognition, Eigenfaces [71] involves Principal Component Analysis (PCA) to reduce the dimensionality of the face image data, focusing on the most significant features. An extension of Eigenfaces is Fisherfaces [72] which use Linear Discriminant Analysis (LDA) for feature extraction, emphasizing on maximizing the between-class variance and minimizing the within-class variance. Finally, LBP is another algorithm in this category that involves summarizing local structures in images by comparing each pixel with its surrounding pixels and is particularly effective for texture analysis.

2.4.2.1 Siamese Neural Networks

As described in a previous section (section 2.1.1), a typical CNN architecture is comprised of several layers, each performing a specific function. However, in face recognition their working principle is quite different compared to that described for activity recognition tasks.

Siamese Neural Networks (SNNs) [73] represent a class of CNN neural architectures characterized by their twin sub-networks that share weights and parameters. These subnetworks, often referred to as "twin towers" or "sister networks", are designed to process distinct input samples and produce embeddings, which are compact vector representations capturing the essential characteristics of the input.

An SNN can be defined as a function that maps a pair of input samples to a similarity score, parameterized by the shared weights. The subnetworks are

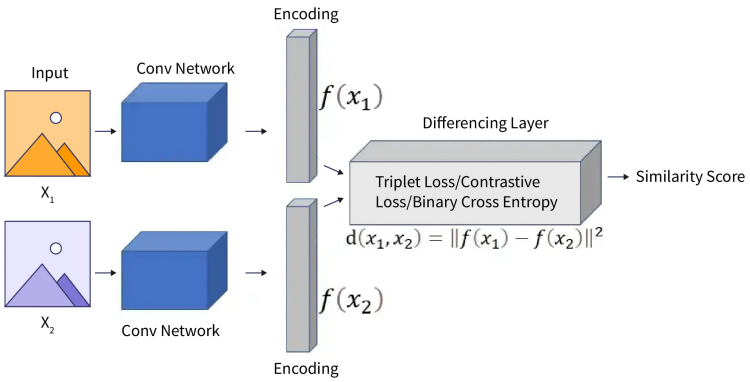

Figure 2.11: SNN architecture for similarity learning. This network processes two input images, x_1 and x_2, through identical convolutional neural networks to generate embeddings $f(x_1)$ and $f(x_2)$. These embeddings are compared using a differencing layer that computes the distance $d(x_1, x_2) = \|f(x_1) - f(x_2)\|^2$. The distance metric is then used to derive a similarity score, optimized using loss functions such as Triplet Loss, Contrastive Loss, or Binary Cross Entropy. The aim is to minimize the distance between similar pairs and maximize the distance between dissimilar pairs.

typically identical, meaning they have the same architecture and share the same set of weights during both training and inference (Figure 2.11). The output of the SNN is a similarity metric, which quantifies the degree of resemblance between the two input samples. This metric can be based on various distance or similarity measures, such as Euclidean distance or cosine similarity. The choice of metric depends on the specific application and the nature of the data being processed.

Training an SNN involves optimizing the shared weights to minimize a loss function that encourages the network to produce similar embeddings for similar inputs and dissimilar embeddings for dissimilar inputs. This is often achieved through contrastive learning, where the network is presented with pairs of positive (similar) and negative (dissimilar) samples and trained to maximize the distance between their embeddings.

The Siamese architecture offers several advantages over traditional neural networks. Firstly, it is designed to handle pairwise comparisons, making it well-suited for tasks involving similarity assessment. Secondly, the weight-sharing mechanism reduces the number of parameters that need to be learned, leading to faster training and improved generalization. Finally, SNNs can be easily adapted to different types of input data by modifying the architecture of the subnetworks.

In conclusion, SNNs provide a powerful and versatile framework for ad-

dressing a wide range of problems that involve comparing pairs of samples. Their unique architecture and training methodology enable them to learn discriminative embeddings that capture the underlying relationships between input samples.

2.4.2.2 DeepFace

DeepFace [65] is a DL facial recognition system developed by Facebook that was one of the first models to achieve near-human accuracy on the Labeled Faces in the Wild (LFW) benchmark [74], a standard for measuring the effectiveness of facial recognition technologies. Its architecture is based on CNNs and consists of multiple layers, each designed to extract and learn features from facial images. The key components of the DeepFace architecture include convolutional layers, responsible for extracting features from the input images; max-pooling layers, for reducing spatial size of the convolved features; fully connected layers, for integrating the learned features from the convolutional layers for classification; and a softmax output layer that is used for categorizing the input face into one of the known identities.

The algorithm's process can be broken down into four main steps. First, during the face alignment stage, DeepFace aligns the faces using a 3D model to ensure they are centered and properly rotated, which helps mitigate issues related to pose variations. Next, in the feature extraction and learning phase, DeepFace's convolutional neural network (CNN) extracts features from the aligned images through convolutional layers. Following this, the pooling layers reduce the dimensionality of the feature maps. Finally, in the fully connected layers and classification stage, the network combines the features to classify the face accurately.

DeepFace also introduced unique aspects for facial recognition including 3D face alignment, which is the use of a 3D model for face alignment so as to significantly improve performance by normalizing the pose variations in the input images. In addition, Deepface also employs a very deep architecture, which allows it to learn complex and high-level facial features. Finally, the model's raining on a vast dataset enables it to achieve high generalization, making it robust to a wide range of faces and expressions.

2.4.2.3 FaceNet

FaceNet [75] is a facial recognition system developed by Google that, unlike traditional facial recognition systems that focused on classifying faces or verifying a match, aims to directly learn a mapping from face images to a compact Euclidean space. In this space, distances directly correspond to a measure of face similarity. This approach represented a significant shift in how facial recognition was approached and has had a profound impact on the field.

FaceNet's architecture, similar to DeepFace's architecture, is also built upon deep CNNs. While it can utilize different CNN architectures, the inception model has been commonly used due to its efficiency and performance. The key components of the FaceNet system include convolutional layers to extract features from the input images; batch normalization, which is used for normalizing the inputs to each layer, thus speeding up the training; pooling layers to reduce the spatial dimensions of the feature maps; fully connected layers that contribute to creating the embedding representation; and L2 normalization, which is applied to the embedding layer to constrain the embedding on a hypersphere.

The algorithm's pipeline involves several key steps. First, in the feature extraction phase, FaceNet uses a deep convolutional neural network (CNN) to extract features from facial images. The convolutional layers in the network are responsible for this extraction. During training, FaceNet employs a triplet loss function designed to ensure that an anchor image (A) of a person's face is closer to all other positive (P) images of the same person than to any negative (N) images of any other person. The triplet loss is mathematically formulated as follows:

$$L = \sum_{i=1}^{N} \left[\|f(x_i^a) - f(x_i^p)\|_2^2 - \|f(x_i^a) - f(x_i^n)\|_2^2 + \alpha \right]$$

where $\|\cdot\|_2$ is the L2 norm, α is the margin enforced between positive and negative pairs, and $f(x)$ represents the embedding of an image x. Finally, the network generates an embedding, which is a vector representation of the face in a high-dimensional space. This embedding is designed such that the Euclidean distance between two embeddings directly corresponds to facial similarity.

In summary, FaceNet represents a major advance in facial recognition technology, introducing a powerful, embedding-based approach. Its use of the triplet loss function and focus on generating discriminative embeddings have set new standards in the field, while the model's architecture and training method-

ology have inspired subsequent research and development.

2.5 Object Detection

Object detection is a field of computer vision that has seen remarkable advancements in the last few decades. This technology, which allows computers to identify and locate objects in visual data, plays a pivotal role in a variety of applications, from AVs and security systems to medical imaging and content analysis.

Early object detection methods were largely based on simple pattern recognition and feature detection techniques. These included the use of edge detection filters, region-based methods such as blob analysis and feature-based methods using keypoints and descriptors, including SIFT and HOG [76]. These techniques were effective to a certain extent, but had limitations in terms of accuracy and robustness, especially in complex visual scenes. However, the introduction of ML brought significant improvements. Techniques such as SVMs [77] were employed for object classification, using features extracted from images [51]. While these methods showed better performance than their predecessors, they still struggled with challenges like variations in object scale, pose, lighting, and background clutter.

The real breakthrough in object detection came with the development of CNNs [78] and their ability to learn hierarchical feature representations directly from data, revolutionized the field. The first major success of CNNs in object detection was demonstrated by the Region-based CNN (R-CNN) model [79],which, along with its more efficient successors, Fast R-CNN and Faster R-CNN, used a two-stage approach where the first stage generated potential object bounding boxes, and the second stage classified these boxes using CNN features.

However, two-stage detectors, while accurate, were often slow, making them unsuitable for real-time applications. This led to the development of single-stage detectors, such as YOLO [40] and SSD [80], which simplified the detection pipeline by combining the two stages into a single network pass, drastically improving the speed of detection. YOLO, in particular, became famous for its ability to perform object detection in real-time with reasonable accuracy.

2.5.1 Early Object Detection Methodologies

The early methodologies in object detection, prior to the DL era, were fundamentally different from the current approaches, as they relied heavily on hand-engineered features and traditional ML-based algorithms.

There are four main categories of featured-based approaches, namely the SIFT, the HOG, the sliding window approach and the classifiers. The SIFT approach was developed by David Lowe in 1999 [81] to extract distinctive invariant features from images that are robust to changes in scale, noise, illumination and minor variations in viewpoint. This approach begins by identifying key locations in scale space by looking for stable features across scales and accurately localizing the keypoints and eliminating those with low contrast or poorly localized along an edge. Finally, the next steps include assigning orientation to each keypoint to achieve invariance to image rotation, as well as generating a unique fingerprint for each keypoint based on local image gradients.

The HOG approach was introduced by Dalal and Triggs [51] in 2005 and focuses on the structure or shape of an object. It works especially well for detecting pedestrians in images, a classic problem in computer vision. More specifically, this algorithm begins by dividing the image into small, connected regions known as cells and for each cell, it computes a histogram of gradient directions or edge orientations for the pixels within the cell. After the histograms are produced, they need to be normalized to enhance invariance to changes in illumination and contrast. Finally, the HOG feature descriptors are formed by concatenating these normalized histograms.

Finally, the sliding window approach involves systematically sliding a window of a fixed size over the image and, for each window position, features (like HOG or SIFT) are extracted and fed into a classifier (e.g., SVM) to determine whether the window contains an object of interest.

2.5.2 Deep Learning Evolution in Object Detection

As mentioned above, DL algorithms have revolutionized the field of object detection, offering substantial improvements in accuracy and efficiency over traditional image processing methods. In contrast to the aforementioned hand-engineered feature-based approaches, DL algorithms have the ability to learn

complex patterns and generalize across diverse datasets. Some of the most common DL algorithms for object detection include Region-Based Convolutional Neural Networks (R-CNNs), You Only Look Once (YOLO) and Single-Shot Detectors (SSDs).

2.5.2.1 Region-Based Convolutional Neural Network

R-CNNs marked a significant shift in object detection methodologies, introducing DL into a domain traditionally dominated by hand-engineered feature extraction and classifiers. They were developed by Ross Girshick et al. in 2014 [79] as a way of integrating high-capacity CNNs with region proposal methods to localize and classify objects within an image. Unlike prior methods that used sliding windows, R-CNNs focus on a selective search to propose potential object regions, significantly reducing the computational burden.

There are four core principles of a R-CNN model, the selective search for region proposals; the feature extraction with CNNs; the classification with SVMs; and the bounding box regression. The first step in a R-CNN model is to identify potential object regions, known as region proposals, using a selective search algorithm. This algorithm groups pixels based on various features like texture, color and size, to hypothesize about object locations. The selective search algorithm is a heuristic method combining the strengths of both exhaustive search and segmentation.

During the feature extraction process, each proposed region is then warped or cropped and resized to a fixed size, typically 227×227 pixels, to be fed into a CNN, which acts as a feature extractor, converting each region into a high-dimensional feature vector. After the conversion of each region into a feature vector, the extracted features are classified using class-specific linear SVMs rather than using the CNN's softmax layer for classification. Each SVM is trained to recognize a specific class. Finally, for each class, a linear regression model is trained to predict the bounding box coordinates, refining the localization of each object.

R-CNNs have been established in object detection and they inspired a series of innovations, leading to more sophisticated architectures, such as the Fast R-CNN [82], which proposes a region of interest polling layer to share convolutional features among region proposals to improve both speed and memory efficiency, and the Faster R-CNN [41], which integrates the region proposal net-

work into the architecture, making the process of generating region proposals part of the neural network, further improving efficiency and speed.

2.5.2.2 You Only Look Once

YOLO is another approach to object detection, developed by Joseph Redmon et al. [40], that differs significantly from the region proposal-based methods like the R-CNNs. YOLO redefines object detection as a single regression problem, straight from image pixels to bounding box coordinates and class probabilities. YOLO's primary objective lies in its speed and efficiency, as it processes the entire image at once (hence the name "You Only Look Once"), making it significantly faster than methods that process multiple regions separately. This unified approach is especially well-suited for real-time applications.

The architecture of the YOLO framework is mainly classified into three layers, the single convolutional network, the confidence scores and the class probabilities. Regarding the single convolutional network, YOLO employs a single CNN to predict multiple bounding boxes and class probabilities for those boxes. The network divides the input image into an $S \times S$ grid and, for each grid cell, the network predicts B bounding boxes and confidence scores for those boxes, as follows [40]:

$$\text{Confidence} = \Pr(\text{Object}) \times \text{IOU}_{\text{pred}}^{\text{truth}} \tag{2.8}$$

where $\Pr(\text{Object})$ denotes the probability of an object in the box and $\text{IOU}_{\text{pred}}^{\text{truth}}$ is the Intersection Over Union (IOU) between the predicted box and the ground truth.

Each bounding box consists of 5 predictions: x, y, w, h, and confidence. The (x, y) coordinates represent the center of the box relative to the bounds of the grid cell. Width (w) and height (h) are predicted relative to the whole image.

The confidence score reflects the model's certainty that the box contains an object and how accurate it thinks the box is that it predicts. If no object exists in that cell, the confidence scores should be zero. Lastly, each grid cell predicts C conditional class probabilities, one per class for the classes being detected, as follows [40]:

$$\text{C} = \text{classConditionalProbabilities} \times \text{Confidence} \tag{2.9}$$

In summary, YOLO's innovative approach to object detection demonstrates the power of rethinking traditional methodologies. It balances speed and accuracy, making it a foundational model in real-time object detection. Its evolution and the continued development of its variants underscore its significant impact on the field.

2.5.2.3 Single Shot Detector

Single Shot Detectors (SSDs) are prominent algorithms in the field of object detection, known for their speed and efficiency. Developed by Liu et al. [80], SSDs are designed to outperform similar models like YOLO by handling multiple aspect ratios and offering better accuracy, particularly for small objects. Just like YOLO, SSD is also a one-stage detector that performs object detection in a single shot, thus simplifying the detection process by eliminating the need for a separate region proposal network, striking a balance between speed and accuracy.

From an architectural point of view, the SSD framework consists of four main layers. First, the base network starts with a standard CNN, such as VGG16 [23], truncated before its classification layers. This base network is responsible for extracting feature maps from the input image. Next, additional convolutional layers are added to the base network, progressively decreasing in size, enabling the model to detect objects at multiple scales. These added layers, along with some high-level layers from the base network, are utilized to predict the presence of objects. Finally, SSD predicts bounding boxes with different aspect ratios and scales for each location in the feature maps.

Regarding the overall loss function of the SSD, this is a weighted sum of the localization loss (loc) and the confidence loss (conf), as follows [80]:

$$L(x, c, l, g) = \frac{1}{N} \left(L_{\text{conf}}(x, c) + \alpha L_{\text{loc}}(x, l, g) \right) \tag{2.10}$$

where N is the number of matched default boxes, and the localization loss is the Smooth L1 loss [82] between the predicted box (l) and the ground truth box (g) parameters.

In conclusion, SSDs represent a significant advancement in object detection, particularly in the context of real-time applications. Their innovative approach to handling multiple scales and aspect ratios in a single-shot framework set new benchmarks in the field.

2.6 Chapter Summary

This chapter explored key concepts in computer vision, focusing on pose estimation, activity recognition, facial recognition, and object detection. The integration of deep learning (DL) techniques has significantly enhanced these areas.

Regarding pose estimation, DL techniques have significantly improved accuracy and efficiency, moving from traditional geometric models to advanced algorithms like OpenPose, AlphaPose, and SimpleBaseline. These models enhance the precision and applicability of pose estimation, especially in dynamic environments. As far as activity recognition, the chapter highlights the shift from handcrafted features to robust DL models such as C3D, TSN, TSM, SlowFast Networks, and X3D. These models excel in capturing spatial and temporal dynamics, leading to accurate classification of human actions in videos. Facial recognition has also seen a major transformation with DL. The chapter discusses key models like DeepFace and FaceNet, emphasizing their use of convolutional neural networks (CNNs) to automatically learn complex facial features. This has greatly improved the accuracy of facial recognition, which is crucial for security and personal identification. Finally, in object detection, the chapter outlines the transition from traditional methods to DL algorithms, focusing on R-CNN, YOLO, and SSD models. These models have enhanced real-time object detection by increasing speed and accuracy, which is essential for autonomous vehicles and robotic vision.

3

Accelerated Multimodal Approach for Edge Computing

In line with the trend towards decentralization in data processing, edge computing has gained a lot of popularity [83]. By processing data locally on embedded devices, this approach reduces latency, conserves bandwidth and enhances data privacy in real-world applications. Real-world applications also require high accuracy and robustness, and multimodal processing generally provide more accurate and reliable outputs than unimodal systems by combining information from various sources [84]. Multimodal processing emerges significant challenges in edge computing systems regarding data alignment and standardization and the resource-constrained nature of embedded systems [85].

The main outcome of our research methodology addresses directly those challenges with the development of an accelerated multimodal AI framework for embedded devices. The proposed methodology handles the diversity of data types in embedded devices, including images, audio and sensor readings, required for a multimodal approach. This integration allows for a comprehensive analysis of different datastreams, enhancing the devices' ability to interpret complex scenarios and make informed decisions.

Regarding the resource constrained natures of the edge devices, our framework addresses these limitations by optimizing data processing, therefore con-

serving power and extending operational life — a crucial factor in remote or mobile applications. This optimization is particularly relevant when considering the need for real-time or near-real-time responses in many applications.

The cost-effectiveness of the system is also another challenge that should be addressed [86]. Integrating multiple functionalities into a single framework can reduce the development and deployment costs associated with creating separate systems for each data type. This integration is especially beneficial for small and medium-sized enterprises, which may have limited resources for technological investment.

Finally, in consumer-facing applications, such as smart homes and wearable technology, an accelerated multimodal AI framework can significantly enhance user experience [87]. By understanding and responding to a variety of user inputs such as voice, gestures and environmental cues, these systems offer a more intuitive and seamless interaction for the user.

In conclusion, an accelerated multimodal AI framework for embedded devices represents a holistic solution that stems from our design methodology and addresses a wide range of requirements and challenges.

3.1 Hardware Acceleration and Optimization

As the first step of our research, we investigated hardware accelerators in the field of image processing. One of the most popular class of hardware accelerators are the FPGAs, due to the high-speed and efficient processing not only in image processing but also in many other applications. Their ability to be reconfigured for specific tasks, offer significant advantages over traditional CPUs and GPUs, particularly in terms of parallel processing capabilities and lower power consumption. This is critical in image processing tasks, which often require real-time processing of large datasets, complex operations and high throughput. The parallelism in FPGAs allows for simultaneous processing of multiple image pixels or operations, dramatically reducing computation time.

The chosen methodology involved identifying computationally intensive algorithms commonly used in AI workflows and implementing them on suitable hardware platforms. The focus on grayscale conversion, color transformations, noise reduction, and edge detection stemmed from their widespread use and computational demands in image preprocessing pipelines across various AI domains [88]–[90]. These designs were implemented on the Altera DE2-115 FPGA

and Pynq-Z1 platform, respectively, and played a key role in AI model training by offloading data loading and augmentation tasks from the GPU.

3.1.1 Accelerated Designs for Color Transformation and Edge Detection

Starting with color transformations, grayscale conversion is a dimensionality reduction technique that discards color information, often irrelevant for tasks such as object detection or classification, while decreasing computational complexity. By implementing this transformations on the Altera DE2-115 FPGA, we can achieve substantial acceleration compared to software implementations, enabling more efficient data preprocessing during both training and inference [91].

VHDL Implementation

The core of the implementation is an Altera Cyclone IV EP4CE115 FPGA device. This device offers substantial computational resources, including 114,480 logic elements, up to 3.9-Mbits of RAM and 266 multipliers, making it well-suited for handling intensive image processing tasks. Additionally, the system includes a USB Host/Slave Controller (Cypress CY7C67200) for interfacing with external devices. A key component of the system is the NIOS II soft processor, a configurable CPU core synthesized on the FPGA. This processor is responsible for managing the USB protocol, enabling communication between the FPGA and a host computer. It uses the Avalon MM interface, a standard bus in Altera's FPGA architecture, to interact with the image processing peripheral.

The Grayscale module converts color images into grayscale images. The conversion adheres to the ITU-R BT.709 recommendation [92], which uses specific coefficients for the RGB channels to calculate the grayscale value. The formula used is:

$$out = (0.21 * R + 0.72 * G + 0.07 * B) \tag{3.1}$$

Moreover, the Edge Detection module implements the Sobel Edge Detection algorithm [93]. It uses a pair of 3×3 convolution masks, one for detecting

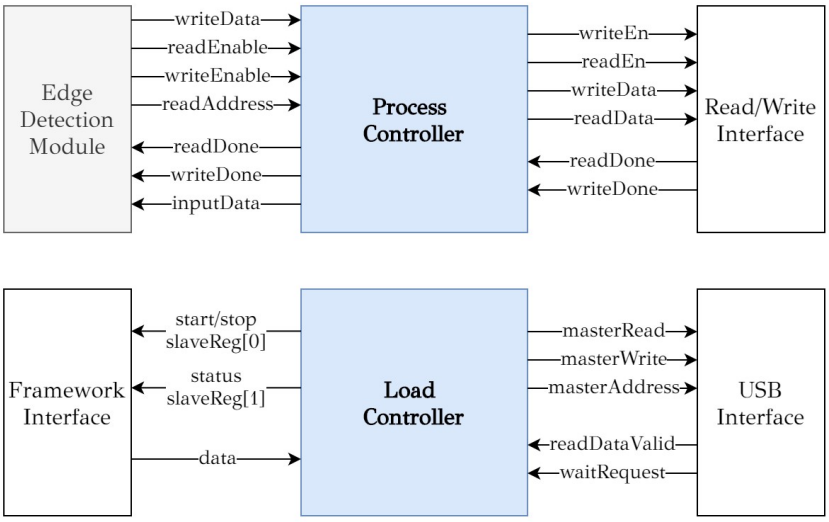

Figure 3.1: Edge Detection RTL Diagram. The top-level modules include the Edge Detection Module, Process Controller, and Read/Write Interface, which handle image data processing and memory interaction. The bottom-level modules, Framework Interface, Load Controller, and USB Interface, manage communication with external devices, control data loading, and facilitate data transfer with the host computer.

horizontal gradients (Gx) and the other for vertical gradients (Gy). The module processes the grayscale image and highlights the edges by calculating the gradient strength at each pixel.

This VHDL code (Listing 3.1) implements an image processing system designed for the Altera DE2-115 FPGA board, performing the Sobel edge detection. The Sobel edge detection process computes the gradient in both the x (gx) and y (gy) directions using the Sobel operator, and then calculates the gradient magnitude by summing the absolute values of gx and gy. The resulting grayscale value and the upper 8 bits of the Sobel edge detection value are assigned to the output signals out_gray and out_sobel, respectively. This design ensures efficient image processing suitable for FPGA implementation, taking advantage of the parallel processing capabilities of the hardware.

Listing 3.1: Our Sobel edge detection process computes the gradient in both the x (gx) and y (gy) directions using the Sobel operator, and then calculates the gradient magnitude by summing the absolute values of gx and gy.

```
-- Sobel edge detection process
process(clk, rst_n)
begin
    if rst_n = '0' then
        gx <= (others => '0');
        gy <= (others => '0');
        sobel_val <= (others => '0');
    elsif rising_edge(clk) then
        -- Sobel Gx calculation
        gx <= (-1 * conv_integer(in_r) + 1 * conv_integer(in_b) +
               -2 * conv_integer(in_g) + 2 * conv_integer(in_g) +
               -1 * conv_integer(in_b) + 1 * conv_integer(in_r));

        -- Sobel Gy calculation
        gy <= (1 * conv_integer(in_r) + 2 * conv_integer(in_g) + 1
            * conv_integer(in_b) +
                -1 * conv_integer(in_b) + -2 * conv_integer(in_g) +
                   -1 * conv_integer(in_r));

        -- Calculate the gradient magnitude
        sobel_val <= std_logic_vector(to_unsigned(
                        integer(abs(gx) + abs(gy)),
                        sobel_val'length));
    end if;
end process;
```

The hardware platform for this approach is the DE2-115 Cyclone IV E board, notable for its Altera Cyclone IV EP4CE115 FPGA device, which features an impressive array of logic elements, RAM and multipliers. The VHDL design encompassed image processing cores dedicated to grayscale conversion and Sobel edge detection. The grayscale module adhered to the ITU-R BT.709 recommendation for converting color to grayscale.

The edge detection module employed horizontal and vertical gradient detection kernels based on the Sobel Operator. This design was embedded into the FPGA through a NIOS soft CPU, which was instrumental in decoding packets and processing the image detection algorithms. The firmware aspect involved a NIOS II soft processor managing the USB protocol and interfacing with image processing peripherals. Additionally, a host software, developed in C, facilitated the interface with the USB Controller and managed the workflow of image file

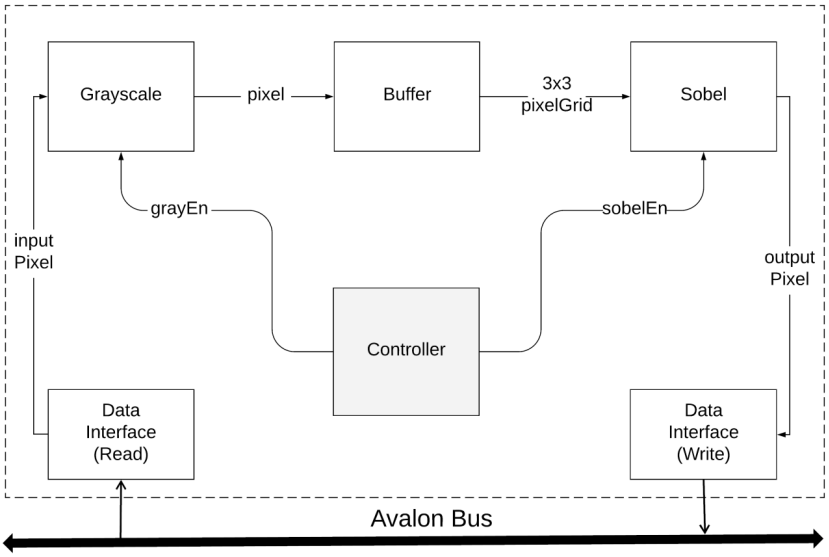

Figure 3.2: High-level architecture of an image processing accelerator on an FPGA, featuring a pipeline for grayscale conversion, pixel buffering, and Sobel edge detection. The controller manages data flow through the Avalon bus, coordinating input/output interfaces and processing modules.

inputs and outputs. This hands-on approach provided valuable insights into low-level hardware design and FPGA architecture.

High Level Synthesis Implementation

While our exploration of hardware-accelerated image processing began with a VHDL implementation of on the Altera DE2-115 FPGA, we continued to streamline the development process and explore higher levels of abstraction by transitioning to the PYNQ-Z1 platform. This allowed us to rapidly prototype using HLS and deploy our algorithms using Python, significantly reducing development time compared to traditional VHDL coding. By comparing resource utilization, performance, and development time between the two implementations, we gained a comprehensive understanding of the trade-offs between low-level control and high-level productivity in FPGA design. This comparative analysis will inform our future hardware acceleration projects, guiding us towards the most suitable methodology for achieving our specific goals.

The objective remained consistent with the VHDL approach, yet the execution differed significantly. The hardware platform used was the Pynq-Z1 board from Xilinx, equipped with advanced features like a dual-core Cortex-A9 processor.

In the second design, we exploited HLS capabilities to generate the Intellec-

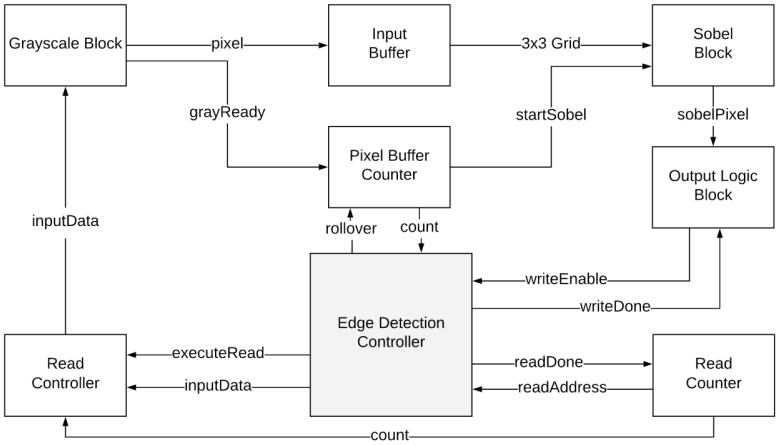

Figure 3.3: High-level architecture of an FPGA-based image processing pipeline for edge detection. The pipeline includes grayscale conversion, pixel buffering, 3×3 grid organization, and Sobel edge detection calculations, all orchestrated by a central controller for efficient data flow and processing.

tual Property (IP) cores, by using C++ code written by us. In the new design, we combined the Grayscale and Edge Detection modules together, to further reduce the resource utilization. To exploit parallelism at these levels, we needed each application's timing diagrams for various inputs and function calls, as well as an in-depth code analysis. In this paper, our goal is each function's optimization by exploiting a variety of design techniques, namely loop unrolling and pipelining.

The HLS implementation for the Edge Detection IP consists of four stages. The first (AXIS2GrayArray) and the last (GrayArray2AXIS) are responsible for transferring the image data through the AMBA AXI4-Stream Protocol Specification. We combine the grayscale conversion into the first stage to simplify the design. In the second stage, we applied the Sobel operator but, to improve the edge detection and provide more accurate results, we applied a Non-Max suppression algorithm, which helps to eliminate the points that do not lie in important edges. The second IP helps to improve image reconstruction performance by offloading CPU calculations to Programmable Logic (PL), as the three RGB channels are encoded into a 32-bit unsigned integer (Listing 3.2). This is a crucial step, because, as we discovered in our experiments, performing this computation on the CPU incurred 70ms of extra delay for every image of $640{\times}480$ pixels. Using our optimization, we reduced this operation to under 1 s. We implemented it through AXI4-Stream, further optimizing the transfer of the data by using a bidirectional DMA (read and write), benefiting our architecture with reduced resource usage, in contrast to other implementations, which use

one DMA for reading and one for writing

Listing 3.2: The formula that decodes an unsigned integer to three RGB channels

```
// Channel:
// 1 - Red, 2 - Blue , 3 - Green
out_data->data = (in_data->data & 0xff0000) >> 16;
out_data->data = (in_data->data & 0x00ff00) >> 8;
out_data->data = in_data->data & 0x0000ff;
```

In Listing 3.3, the provided C++ implementation demonstrates the color transformation using HLS for converting an RGB image to grayscale, specifically designed for use with the Vivado HLS tool by Xilinx. Within the top-level function ‘rgb2gray‘, we start by converting the input AXI stream (‘src_axi‘) to an HLS image format (‘img_0’) with the ‘hls::AXIvideo2Mat‘ function. The grayscale conversion occurs within nested loops, where each pixel’s RGB values are read, and the grayscale value is calculated using the formula ‘gray = 0.21 * R + 0.72 * G + 0.07 * B‘. This grayscale value is then assigned to a new pixel in the output image (‘img_1‘). After processing all pixels, the function ‘hls::Mat2AXIvideo‘ converts the grayscale HLS image back to an AXI stream (‘dst_axi‘). The code uses HLS pragmas to optimize the loop for efficient hardware execution, ensuring that each pixel’s operations are pipelined.

Figure 3.4 depicts a block diagram with the interface connections of the Edge Detection with the DMA input and output of data. Researchers have reported that, at a clock frequency of 100 MHz, data transitions may be established from both AXI4 master and AXI-Stream slave to AXI4 master at a data rate of 400 MBps and 300 MBps, respectively, being a quota of the theoretical bandwidths of 99.76% and 74.64%.

Listing 3.3: C++ block for the grayscale conversion HLS design using the AXI-Stream interface

```
1
2  void rgb2gray(AXI_STREAM& src_axi,
3                AXI_STREAM& dst_axi, int rows, int cols) {
4
5      // Convert AXI4 stream to HLS image format
6      hls::AXIvideo2Mat(src_axi, img_0);
7
8      // Custom RGB to Grayscale conversion
9      for (int i = 0; i < rows; i++) {
10         for (int j = 0; j < cols; j++) {
11             #pragma HLS pipeline
12             hls::Scalar<3, unsigned char> pixel;
13             hls::Scalar<1, unsigned char> gray_pixel;
14
15             // Read pixel from RGB image
16             pixel = img_0.read(i * cols + j);
17
18             // Compute grayscale value using the formula:
19             // gray = 0.299*R + 0.587*G + 0.114*B
20             unsigned char gray_value =
21                 (unsigned char)(0.21 * pixel.val[0]
22                 + 0.72 * pixel.val[1]
23                 + 0.07 * pixel.val[2]);
24
25             // Assign the computed grayscale value
26             gray_pixel.val[0] = gray_value;
27
28             // Write the grayscale pixel to the output image
29             img_1.write(i * cols + j, gray_pixel);
30         }
31     }
32
33     // Convert HLS image format to AXI4 stream
34     hls::Mat2AXIvideo(img_1, dst_axi);
35 }
```

Table 3.1 indicates the timing summary as reported by Vivado 2016 with the Pynq-Z1 project for our complete system implementation.

Figure 3.5 indicates the power consumption of our implementation, as reported by Vivado after synthesis. The passive power consumption is around 0.156 watt (10%) of the total dynamic power requirements (1.383 watts). We

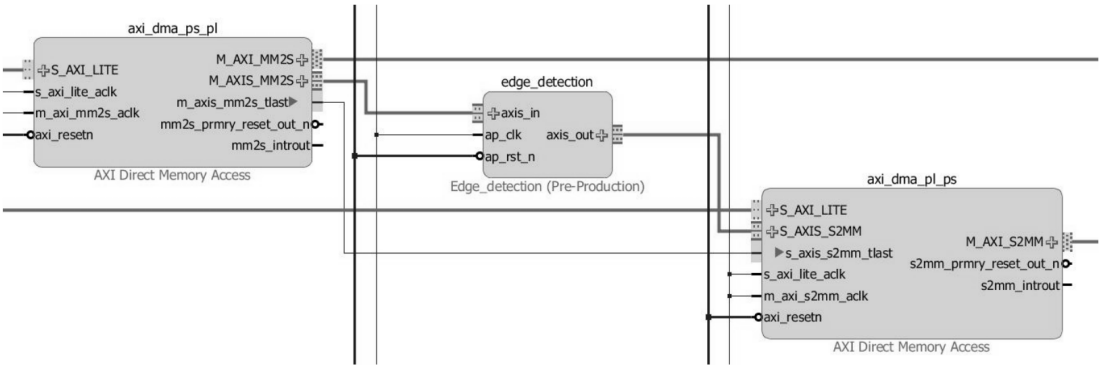

Figure 3.4: AXI direct memory access (AXI DMA) connections provide a high-bandwidth direct memory access between memory and AXI4-stream-type edge detection peripheral.

Table 3.1: Design timing summary, showing the worst slack and the number of failing endpoints for setup time, hold time, and pulse width constraints.

Setup	Hold	Pulse Width
Worst Slack: 0.358 ns	Worst Slack : 0.020 ns	Worst Slack: 3.750 ns
Total Slack: 0 ns	Total Slack: 0 ns	Total Slack: 0 ns
Failing endpoints: 0	Failing endpoints: 0	Failing endpoints: 0
Total endpoints: 39,835	Total endpoints: 39,835	Total endpoints: 39,835

can safely assume that, with the Zynq processing system (PS7) idle, we can expect an average power consumption of 0.5 watt and below. In comparison with other accelerators, such as that in [94], which often exceed 1.5 watts of power consumption, our implementation is about three times more energy efficient.

The processing time of the Pynq-Z1 implementation averaged around 0.0422 seconds per image and the DE2-115 implementation around 0.0983 seconds. However, the Pynq-Z1 board is running at 100 MHz instead of 50 MHz; thus, we provide a theoretical sample by degrading the Pynq-Z1 performance by half. The expected total processing time of the Pynq-Z1, running at a clock speed of 50 MHz, should be 0.0844 s (or 84 ms) approximately. The final result at the same clock speed is still better at the Pynq-Z1 board, resulting in a 14% faster

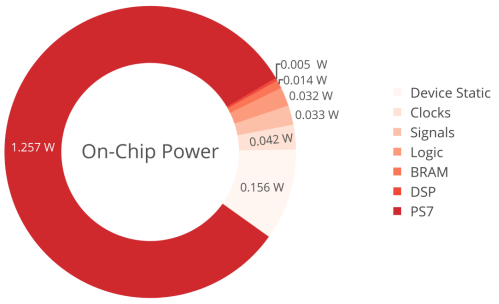

Figure 3.5: Power consumption per hardware component.

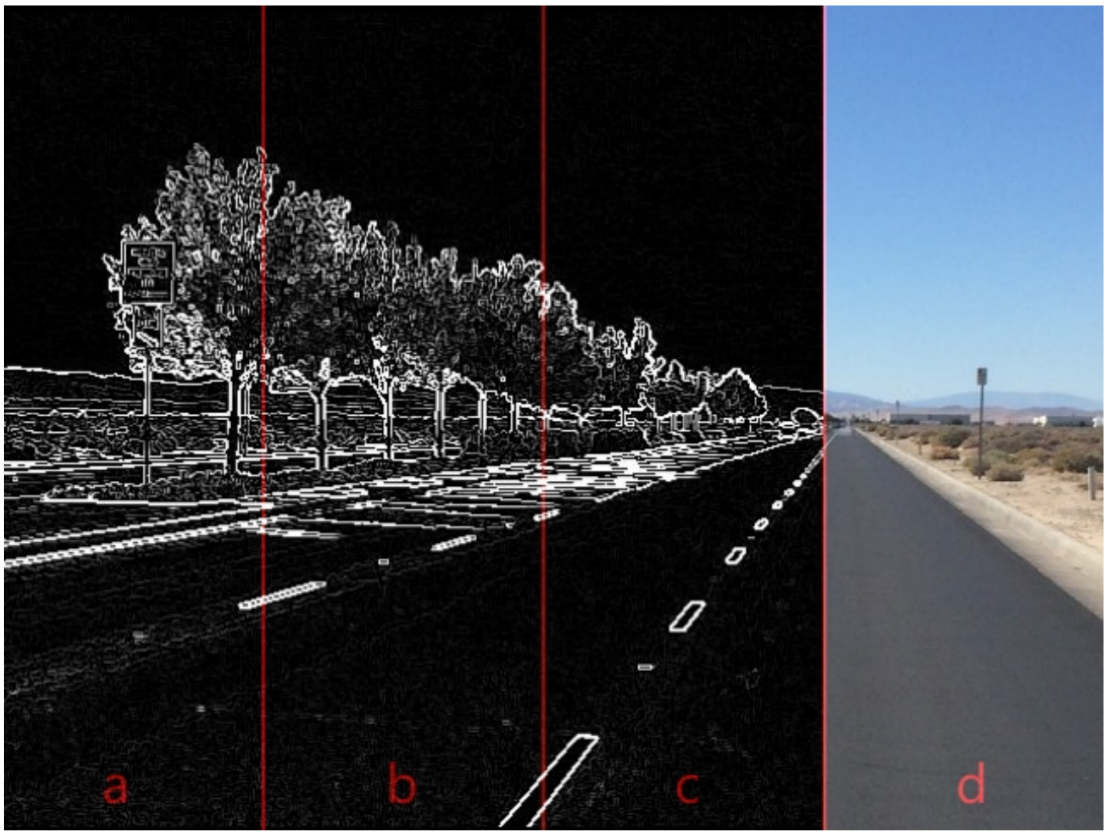

Figure 3.6: Combined results from all platforms: (a) the result of the Pynq-Z1 implementation; (b) the result of DE2-115 implementation; (c) the result of a C++ algorithm running on a traditional CPU; and (d) the original image.

computation time.

Finally, the study showcased the significant advancements in HLS tools, demonstrating their ability to provide competitive Quality of Results (QoRs) when combined with manual RTL designs. The HLS approach, with its superior optimization in performance and efficiency, showed promising potential of HLS within FPGA design communities for more rapid and efficient development processes. This study not only provides a comprehensive comparison of the two methodologies, but also opens avenues for further research and development in the field of image processing accelerators.

Figure 3.6 gives a typical processed image outcome partitioned in four subframes. The first subframe depicts the result of the Pynq-Z1 implementation and the second one the result of DE2-115 implementation. In the third subframe, we can see the outcome of the same algorithm running on a traditional CPU and in the fourth and final one, the original image. We need to address that our hardware designs provide accurate edge detection, identical to the output of the algorithm running on the CPU.

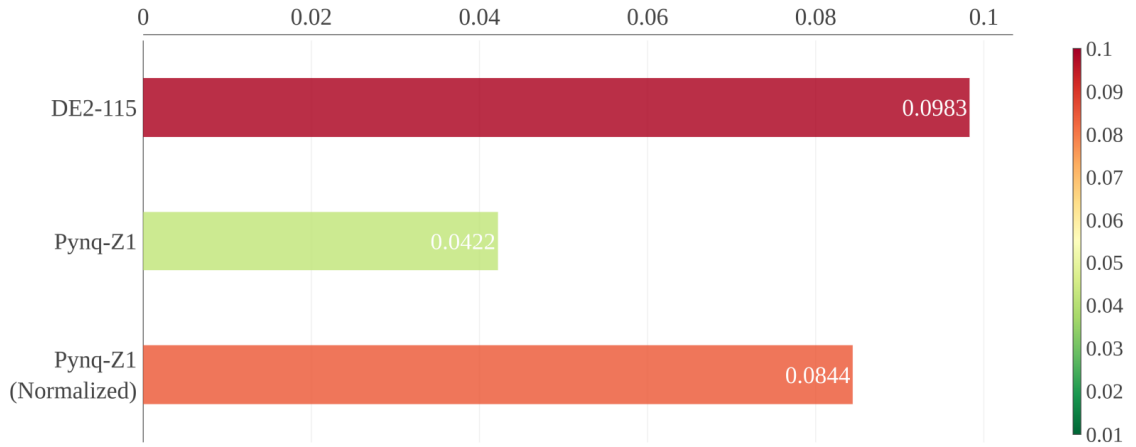

Figure 3.7: Processing time comparison between the two boards (lower is better). Normalized performance is the theoretical performance with both boards running at the same clock speed.

Based upon our results, it seems that the HLS design performed slightly better in comparison with the bare VHDL implementation (Figure 3.7). There are some minor hardware differences between the two platforms but we can safely assume that the HLS tools did an excellent job, producing a highly competitive design comparable and slightly better than the manual counterpart. In addition, the HLS tool made a huge increase in productivity, since many of the supported hardware underlying functions (e.g., AXI4 protocol) are pre-implemented and ready to use.

With the correct configuration, the HLS tool produced highly optimized VHDL code for the specific design. The two operators can be "chained" together in a single cycle by the designer, by performing operation scheduling within the target clock periods, so that false paths [95] are avoided. It can reduce the number of bits required by datapath operators by making bitwidth optimization, reducing the area and power and increasing the performance of the design.

Finally, HLS can also make use of the multiple BRAMs to store data structures for fast memory accesses at a low cost. These memory elements have a limited number of memory ports and the customization of memory accesses may require the creation of an efficient multi-bank architecture to avoid limiting the performance. In addition, there are numerous loop optimizations by pipelining, allowing a loop iteration to start before the completion of its predecessor, provided that data dependencies are satisfied. There are many more HLS optimizations besides the above, which do not affect our current design implementation, but in other cases may dramatically increase the performance [96].

3.1.2 Accelerated Noise Reduction Design

During our the exploration of AI applications, we have observed that their performance is often hindered by the presence of noise in raw data. This noise can obscure relevant features, degrade accuracy, and increase computational demands. Recognizing the need for an additional preprocessing step to address this issue, we worked on the development of an accelerated noise reduction system implemented on the Xilinx Pynq-Z1. By exploiting the parallel processing capabilities and hardware acceleration of FPGAs, we aim to efficiently mitigate noise in real-time, therefore enhancing the quality of data fed into AI models.

This section details the noise reduction technique that employs image stacking, a process where multiple frames are captured in quick succession and then algorithmically combined. By averaging pixel values across these frames, the technique enhances signal quality and reduces noise. The theoretical underpinnings of image stacking, including the statistical basis for noise reduction through averaging, are explained. This section also discusses the conditions under which image stacking is most effective and the parameters that influence its performance.

Noise in digital photography can be broadly categorized into two types: (a) shot noise, which arises from the discrete nature of photon capturing and (b) thermal noise, stemming from sensor and circuitry imperfections. Low-light conditions enhance these noise types, leading to grainy and poor-quality images. Existing noise reduction techniques, including spatial and temporal methods, come with significant trade-offs. Spatial filters, for instance, may reduce noise, but also tend to blur the image, leading to loss of detail. Temporal averaging can be effective, but often requires static scenes and is computationally intensive.

To tackle the inherent trade-offs present in conventional photography techniques, such as aperture limitations in cell phone cameras or the inadequacies of synthetic aperture formation and optical image stabilization, a camera system was developed capable of capturing and merging multiple images. This system effectively addresses the limitations of low light photography, offering a means to enhance image quality without the drawbacks of traditional methods.

The core of this research lies in the methodology of burst photography, where multiple images are captured and merged to create a single, high-quality image. This process involves capturing images with lower exposure to avoid highlight clipping, thus enabling the capture of a more dynamic range. Shorter exposure times are also selected to reduce camera shake blur. The system selects

a "reference" frame from the burst and merging patches from "alternative" frames into this frame to reduce noise and enhance overall image quality. The hardware implementation of this approach is carried out on a Xilinx PYNQ-Z1 board, chosen for its compatibility with PYNQ, an open-source framework that allows for the exploitation of the capabilities of Xilinx Zynq All Programmable SoCs. The board's features, including a dual-core Cortex-A9 processor and numerous logic slices, make it an ideal platform for this application. The authors used C++ for HLS of the HDL blocks and configured the peripheral blocks for system integration. We exploit HLS capabilities to generate the Intellectual Property (IP) cores, by using C++ code written by us. In order to exploit the hardware parallelism, we need each application's timing diagrams for various inputs, function calls, as well as an in-depth code analysis. Our core consists of two main components:

- **Tile Merging**: The HLS implementation for the Tile Merging IP consists of 4 stages. The first (AXIS2ImgArray) and the last one (ImgArray2AXIS) are responsible for transferring the image data through the AMBA AXI4-Stream Protocol Specification. We combine the array conversion into the first stage to simplify the design. We are storing the images as FIFO (First In First Out) queues and at a second stage we split each image to a predefined set of 3×3 tiles. We are trying to match features between coherent image tiles in order to minimize ghosting and provide more accurate results. Based on some heuristics (color channel isolation, brute-search neighboring pixels etc.) we are trying to match at least one 3×3 pixel grid with minimal color variation (Listing 3.4).

- **Int2RGB**: The second IP helps to improve image reconstruction performance by offloading CPU calculations to Programmable Logic (PL), as the three RGB channels are encoded into a 32-bit unsigned integer. This is a crucial step, because, as we discovered in our experiments, performing this computation on the CPU incurred 280 seconds extra delay for every image of 4000×3000 resolution (12 megapixels). Using our optimization we reduce this operation's latency to under one second. We implement it through AXI4-Stream, furthermore optimizing the transfer of the data by using a bidirectional DMA (read and write), benefiting our architecture with reduced resource usage, in contrast to other implementations, which use one DMA for reading and one for writing.

Listing 3.4: C++ block for tile-based merging with feature matching and noise reduction for high-quality image creation.The process involves tile splitting, feature matching between reference and alternative frames using heuristics based on pixel intensity differences.

```
// Tile-based merging
#pragma HLS PIPELINE
// Extract tiles
...

// Feature matching and noise reduction
bool match = false;
for (int ti = 0; ti < TILE_SIZE; ti++) {
    for (int tj = 0; tj < TILE_SIZE; tj++) {
        // Heuristic for feature matching: comparing pixel
            intensities
        if (abs(tile_ref.val[ti][tj].val[0]
                - tile_alt.val[ti][tj].val[0]) < 10 &&
            abs(tile_ref.val[ti][tj].val[1]
                - tile_alt.val[ti][tj].val[1]) < 10 &&
            abs(tile_ref.val[ti][tj].val[2]
                - tile_alt.val[ti][tj].val[2]) < 10) {
            match = true;
        } else {
            match = false;
            break;
        }
    }
    if (!match) break;
}
// If match is found, average the tiles for noise reduction
if (match) {
    for (int ti = 0; ti < TILE_SIZE; ti++) {
        for (int tj = 0; tj < TILE_SIZE; tj++) {
            hls::Scalar<3, unsigned char> avg_pixel;
            avg_pixel.val[0] = (tile_ref.val[ti][tj].val[0]
                               + tile_alt.val[ti][tj].val[0]) / 2;
            avg_pixel.val[1] = (tile_ref.val[ti][tj].val[1]
                               + tile_alt.val[ti][tj].val[1]) / 2;
            avg_pixel.val[2] = (tile_ref.val[ti][tj].val[2]
                               + tile_alt.val[ti][tj].val[2]) / 2;
            img_1.write(i + ti, j + tj, avg_pixel);
        }
    }
} else { ... }
```

In the experimental section, the system is evaluated using devices with 13-megapixel sensors, capturing bursts of up to 15 frames. This evaluation demonstrates the system's capability to process a significant volume of image data efficiently. The results show that the hardware-accelerated approach is significantly faster than a software implementation running on a standard PC, with the hardware accelerator being 8.5 times faster on average.

We compared the results using the Mean Square Error (MSE) and the peak signal-to-noise ratio (PSNR), as shown in Eq. (3.2) and (3.4), respectively [97]. The PSNR is often used to measure the quality between a compressed and the original image, denoting a peak error measure, whereas the MSE is used to measure the cumulative squared error between the original and the compressed image. Note that the higher the PSNR, the better the quality of the reconstructed or compressed image, while the lower the MSE value, the lower the error, respectively.

$$MSE = \frac{1}{mn} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} (||f(i,j) - g(i,j)||)^2 \tag{3.2}$$

$$T_0(\omega) = \frac{1}{N} \sum_{z=0}^{N-1} T_z(\omega) \tag{3.3}$$

$$PSNR = 20 \log_{10} \left(\frac{MAX_f}{\sqrt{MSE}} \right) \tag{3.4}$$

Finally, MAXf is the maximum signal value existing in the original known to be a high-quality image. We use the MSE metric for practical purposes, being the comparison between the true pixel values of a high-quality image, captured with a professional camera, and the respective ones of the algorithm, in order to minimize the MSE between the two aforementioned images, with respect to the image's maximum signal value.

Our results have less noise than the images derived by a typical imaging pipeline, especially in low-light scenes. In Figure 3.8, we can clearly see the benefits of this approach. We managed to decrease the ISO by a factor of three and still capture a better exposed and without noise image. In our experiment, we used a shutter speed of 1/10 sec per frame, which we decided that is an average value. In a complete camera system those parameters will be automatically adjusted in real-time using sophisticated algorithms, to fully exploit the hardware capabilities.

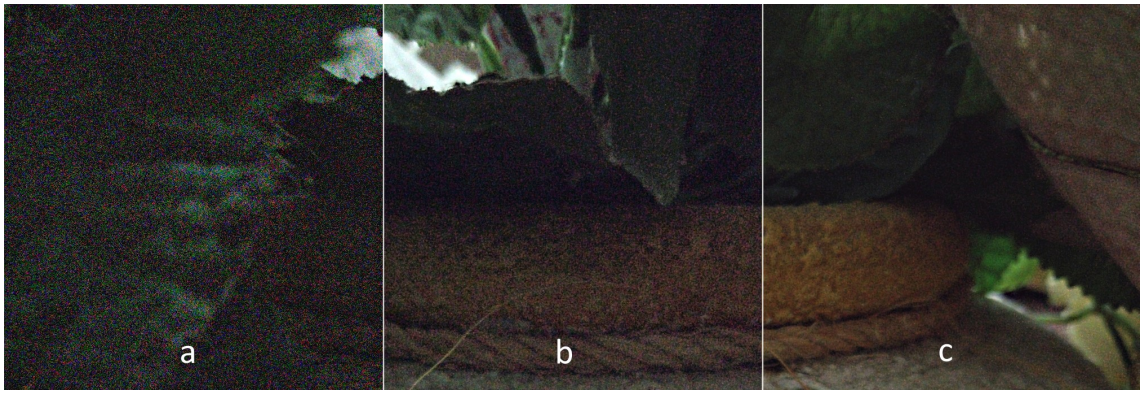

Figure 3.8: Combined image results from a mobile device with Sony IMX363 sensor (f/1.9 3.94mm) in low light (1 lux). At the left (a) is the original image captured (1/10 sec. ISO 9600). In the middle (b) we have an FPGA processed result (1/10sec. ISO 3200, 8 frames). Finally, on the right side (c), the output of the FPGA accelerator is shown (1/10sec. ISO 3200, 15 frames). Clearly the output result (c) is very acute, bright and crisp.

3.2 Optimized Multimodal Approaches

As the need for intelligent, real-time decision-making grows, the focus expands beyond preprocessing to encompass sophisticated AI models capable of running on edge devices. Edge computing refers to processing data at or near the source of data generation, reducing latency and bandwidth usage compared to centralized data centers. This shift necessitates models that are both powerful and optimized for the limited computational resources typical of edge devices. In this section, we will explore highly optimized AI approaches, stemmed from our design methodology, in the domain of activity recognition, abnormal event detection, object detection and facial identification. These AI models feature a novel multimodal design employing different fusion levels and techniques to exploit the complementary data captured by a wide range of sensors.

3.2.1 Multimodal Stream Classification

Real-world data is rarely confined to a single modality. Images, audio, and other sensor data often coexist, offering complementary insights into complex phenomena. Multimodal stream classification addresses the need to analyze and interpret this diverse data holistically. By integrating information from multiple modalities, we can unlock a deeper understanding of events, behaviors, and patterns that would remain hidden when considering each modality in isolation.

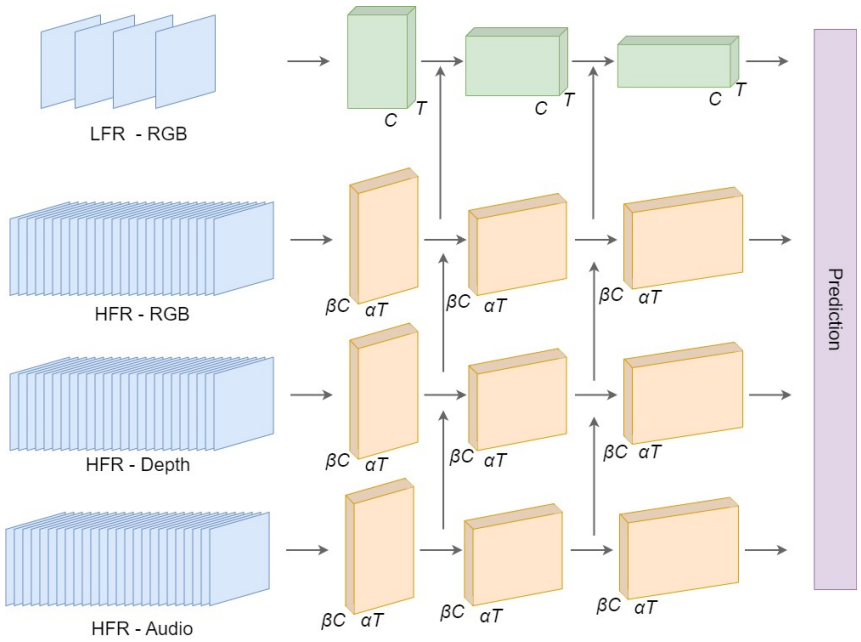

Figure 3.9: Architecture of a multimodal DL model for abnormal event detection. The model processes LFR RGB video (LFR-RGB) and high-frame-rate RGB video, depth maps, and audio (HFR-RGB, HFR-Depth, HFR-Audio) using separate convolutional (C) and transformer (T) blocks before merging their outputs for a final prediction.

The architecture of our model is designed with two separate pathways: the Low Frame Rate (LFR) and High Frame Rate (HFR) pathways. The LFR pathway is adept at processing spatial information at a reduced temporal resolution, capturing detailed spatial features and contextual information within the transportation vehicle. This is essential for understanding the static components of the scene. The HFR pathway, in contrast, is tailored to capture temporal dynamics and rapid movements, which are critical for detecting fast-occurring events such as theft or physical altercations. The integration of these two pathways allows the model to leverage both spatial and temporal data, significantly enhancing its accuracy and robustness in real-time event detection.

A critical component of the model is the sophisticated data fusion strategy employed to combine data from the LFR and HFR pathways. This strategy involves multi-level feature fusion, where features from different layers of both pathways are combined, ensuring that the model benefits from both low-level and high-level feature representations. The synchronization mechanism within the model aligns the outputs of the LFR and HFR pathways, maintaining temporal coherence and ensuring that the fused data is temporally consistent and accurate.

To optimize the model for deployment on edge devices like the NVIDIA Jet-

son AGX Xavier, several enhancements and optimizations were implemented. Model quantization plays a pivotal role in this optimization process. The model employs INT8 quantization, converting 32-bit floating-point parameters into 8-bit integers, significantly reducing the model's size and computational complexity. This allows for faster inference times without substantially impacting accuracy. Quantization-aware training is used to minimize the loss in accuracy near 5%d ue to quantization, adjusting the model's weights during training to accommodate the reduced precision. The use of CUDA acceleration is another key aspect of the model's optimization. By utilizing the CUDA cores for parallel computation, the model processes multiple datastreams simultaneously, substantially reducing inference time and making the model suitable for real-time applications. Custom CUDA kernels are designed to maximize the utilization of the GPU's resources, enhancing the model's computational efficiency. Layer fusion is employed to further optimize the model's performance. This technique combines adjacent layers in the neural network into a single operation, reducing the total number of operations and memory accesses required during inference. This not only minimizes latency, but also increases the throughput of the model.

Asynchronous data loading is implemented to ensure that data loading does not become a bottleneck in the model's performance. By employing multithreading for data loading, the model loads data in parallel with the processing of other datastreams, keeping the GPU continuously fed with data and eliminating idle times. A pre-fetching mechanism is used to load data into memory before it is needed for processing, reducing the waiting time for data retrieval. Moreover, preprocessing steps such as image cropping, resizing and normalization are performed directly on the GPU to optimize the data pipeline. This minimizes the data transfer times between the CPU and GPU, a critical factor in reducing overall latency. The preprocessing pipeline is streamlined and optimized for GPU execution, ensuring that these operations do not hinder the model's performance.

A critical aspect of the model is the fusion of data from the LFR and HFR pathways, ensuring comprehensive video understanding. This fusion allows the model to leverage both spatial and temporal data, enhancing its accuracy and robustness. The architecture utilizes lateral connections between the pathways at various depths, enabling a rich multi-level exchange of spatial-temporal features. This methodological approach ensures that the model is not only able to detect abnormal events accurately, but also do so in real-time, which is vital for the prompt response required in public transportation settings.

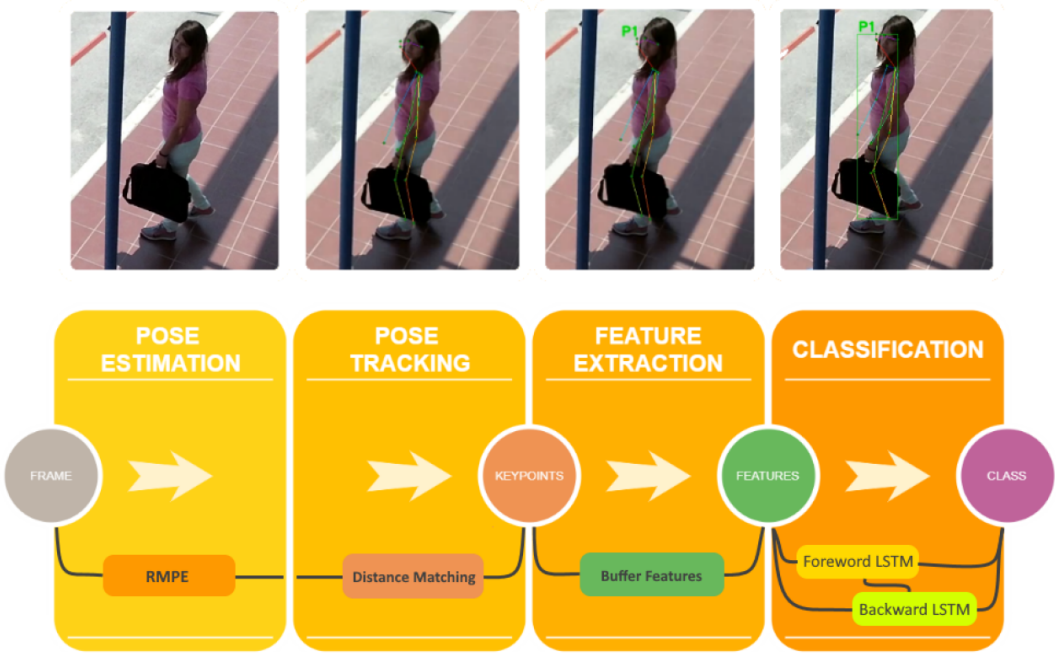

Figure 3.10: Pipeline of the pose estimation classification. First, key points of human figures are identified within each frame. Then, pose tracking is performed by matching key points between frames. Features are extracted from the tracked key points, buffered, and fed into a bidirectional LSTM network for analysis. The LSTM output is then used to classify the pose into predefined categories.

3.2.2 Pose Classification

Human pose estimation, the task of identifying key points on a person's body, has opened doors to numerous applications. However, understanding the meaning behind these poses is equally vital. Pose classification addresses this need by interpreting the arrangement of body key points and assigning them to predefined categories like "standing", "sitting", or "jumping". This enables a deeper understanding of human actions, behaviors, and intentions. Pose classification bridges the gap between recognizing body positions and understanding their significance, expanding the possibilities for technology to interact with and learn from human movement.

The pipeline of our pose classification approach consists of 4 stages as shown in Figure 3.10. In stage one, the pose of each person in the frame is extracted (15 keypoints). In the second stage, a skeleton tracking algorithm is performed, associating persons across multiple frames. Regarding the third stage, the detected and tracked human body key-points are represented as trajectories and during the fourth stage they are "fed" into a Multi-Layer Classifier which classifies each action into normal or abnormal.

3.2.2.1 Pose Estimation

For the first stage, we are using a custom implementation of the RMPE by Fang et al. [39] for training (better accuracy) and OpenPose [19] for testing (higher performance). The integrated implementation uses VGG19, a CNN model proposed by K. Simonyan and A. Zisserman [98]. We are using this model to improve the accuracy of the skeleton extraction for the training process and further perform data augmentation without sacrificing the data integrity. We generate noisy data with variable intensities, based on the extracted data from the backend, and we combine these data with the original ones as an augmentation technique. Extensive tests indicated that the model generalizes better and the accuracy improves. Although we initially implemented AlphaPose using VGG for training our model, multiple tests shown that we could use it for evaluation too, when specific parameters (resolution, heatmaps etc.) are tuned. We apply the pose estimation framework in the presence of inaccurate human bounding boxes. The generated pose proposals are refined by parametric pose NMS to obtain the estimated human poses. In this stage, 17 different human body keypoints are detected and the number of people in each frame is obtained. The number of N frames for the feature generation along with the evaluation accuracy are depicted in the graph below (Figure 3.11). As we can see, a buffer size (window size) of 5 frames achieved the best accuracy on the evaluation test. Higher values may result in lower accuracy as the tracker may fail to consistently detect people when the shuttle is overcrowded. We are currently investigating some optimizations of the tracker, in order to further increase the size of the buffer.

The detected and tracked human body key-points are converted into features and forwarded to an LSTM Neural Network. For extracting features, every person's skeleton data are stored into a circular (ring) buffer deque (double-ended queue) of N frames (window size) into the Feature Generator class.

The buffer T is considered as invalid if the newest appended skeleton does not contain at least the neck (Point 0) or one of the thigh bones (Point 7 or 10) shown in Figure 3.12, as the height of the skeleton (used for normalizing features) cannot be calculated. The feature extraction process occurs when the buffer is full.

The number of N frames for the feature generation along with the evaluation accuracy are depicted in Figure 3.11. A buffer size ($window\_size$) of 5 frames achieved the best accuracy on the evaluation test [99]. Higher values may result

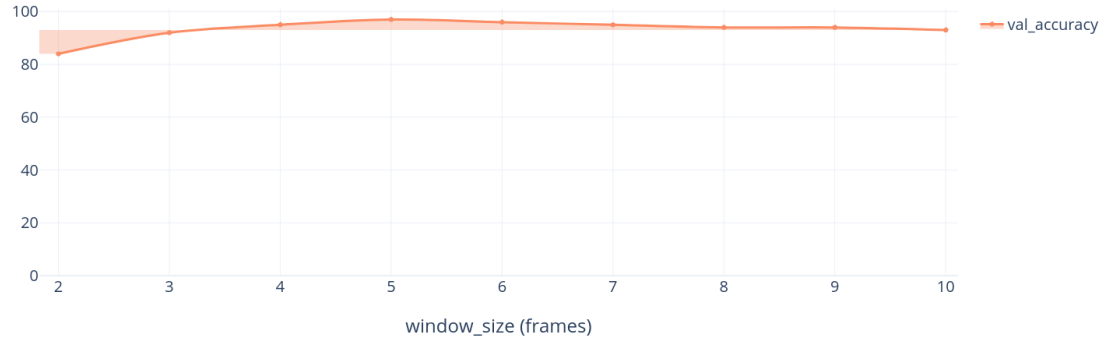

Figure 3.11: Performance evaluation across different buffer sizes. The validation accuracy increases as the window size increases from 2 to 3 frames, peaks around 3-5 frames, and then slightly decreases before stabilizing for larger window sizes. A shaded area around the line indicates the variability of the validation accuracy across different experiments or data splits.

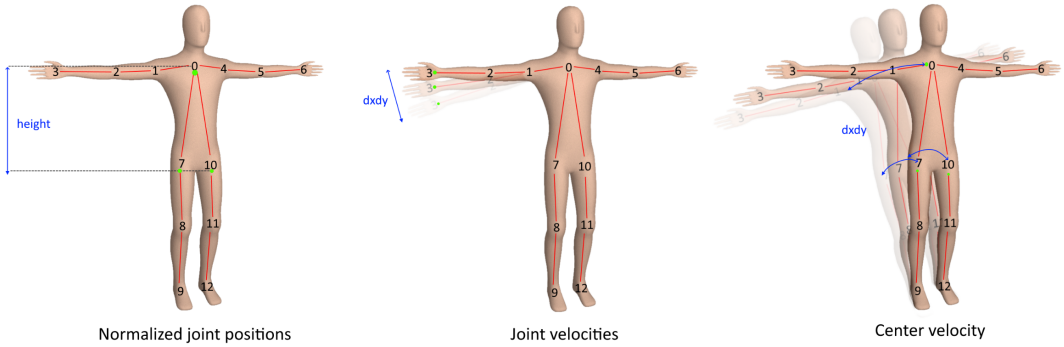

Figure 3.12: Representation of the extracted features: Normalized joint positions relative to body height, joint velocities between frames represented as arrows, and center velocity as a curved arrow depicting overall body movement.

Table 3.2: Features extracted from pose classification.

Feature Extraction	
$\mathbf{X_s}$	A direct concatenation of joints positions of the N frames.
$\mathbf{H}$	Average height of the skeleton of the previous N frames. This height equals the length from Neck to Thigh. Used for normalizing.
$\mathbf{X}$	Normalized joint positions $[\mathbf{X_s} - \text{mean}(\mathbf{X_s})]/\mathbf{H}$
$\mathbf{V_j}$	Velocities of the joints $\{\mathbf{X}[t] - \mathbf{X}[t-1]\}$
$\mathbf{V_c}$	Velocity of the center $\{\text{sum}(\mathbf{X_c}[t] - \mathbf{X_c}[t-1])\}$

in lower accuracy as the tracker occasionally fails to consistently track people when the shuttle is overcrowded. Finally, the LSTM model is capable of binary or multi-class softmax classification and contains three hidden layers of size (32×64) with the Rectified Linear Unit (ReLU) activation function. An overview is shown in Figure 3.10.

3.2.3 Spatiotemporal Autoencoder

Raw video data can be overwhelming, consisting of countless frames and intricate details. Spatiotemporal autoencoders offer a powerful tool for understanding the spatiotemporal patterns hidden within video sequences. By encoding the visual information from consecutive frames into a compressed representation and then reconstructing it, these models learn to capture the essential motion dynamics, object interactions, and scene changes. This enables them to filter out noise, identify salient objects, and predict future frames. In video analysis, spatiotemporal autoencoders find applications in tasks such as anomaly detection, where they can identify unusual events or behaviors by recognizing deviations from the learned patterns.

The approach is based on the principle that the most recent frames of video will be significantly different than the older frames, in case of an abnormal event. Our goal, inspired by Lu et al. [100], is to train an end-to-end model consisting of both a spatial feature extractor and a temporal encoder-decoder that combined learn the temporal patterns of the input volume of frames. So as to minimize the reconstruction error between the input and the output video volume reconstructed by the learned model, we trained our model using video volumes of only normal scenes. After our model's proper training, we expect to have low reconstruction error in a normal video volume as opposed to a video volume containing abnormal scenes. Finally, our system will be able to detect the occurrence of an abnormal event, by thresholding on the error produced by each testing input volume.

3.2.3.1 Architecture

There are two stages that form an autoencoder: encoding and decoding. Autoencoders set the number of encoder input units to be less than the input;

thus, they were first used to reduce dimensionality. Usually, unsupervised back-propagation is used for training, helping the reconstruction error of the decoding results from the original inputs to decrease. Generally, an autoencoder can extract more useful features when the activation function is non-linear rather than some common linear transformation methods, such as PCA.

3.2.3.2 Spatial Convolution

In a deep CNN, the main objective of convolution is to extract information from the input frame. The convolution process maintains the spatial relations of pixels by using kernels to extract low level features. In raw mathematics, the convolution operation performs dot products across filters of partial input. Supposing an nxn input layer, followed by a convolutional layer, then if we use an $m{\times}m$ filter W, the output size will be $(n-m+1){\times}(n-m+1)$. Through the training stage, a CNN learns the values of these filters by itself, although some parameters such as the filter size and the number of layers still need to be defined. The larger the number of filters used, the more information that gets extracted and the better the network generalizes. Yet, there is a trade-off and balance is a critical factor when it comes to the number of filters used, as more filters would impact the performance negatively and require more resources.

3.2.3.3 Preprocessing

At this stage, our task is to convert raw data into aligned and acceptable input for the model. To do so, each frame extracted from the raw videos is then resized to $64{\times}64$. To ensure that the input images are all on the same scale, the pixel values are scaled between 0 and 1 and each frame subtracted by its global mean image for normalization. The mean image is calculated by averaging the pixel values at each location of each frame in the training dataset. After that, the images are converted to grayscale to reduce dimensionality. Finally, the processed images are then normalized to have zero mean and unit variance. As mentioned before, we use video volumes as input to our model, where each volume consists of 10 consecutive frames with various skipping strides (Figure 3.13). As the number of parameters in this model is large, we also need a large amount of training data. Following Lu's [100] practice, to increase the

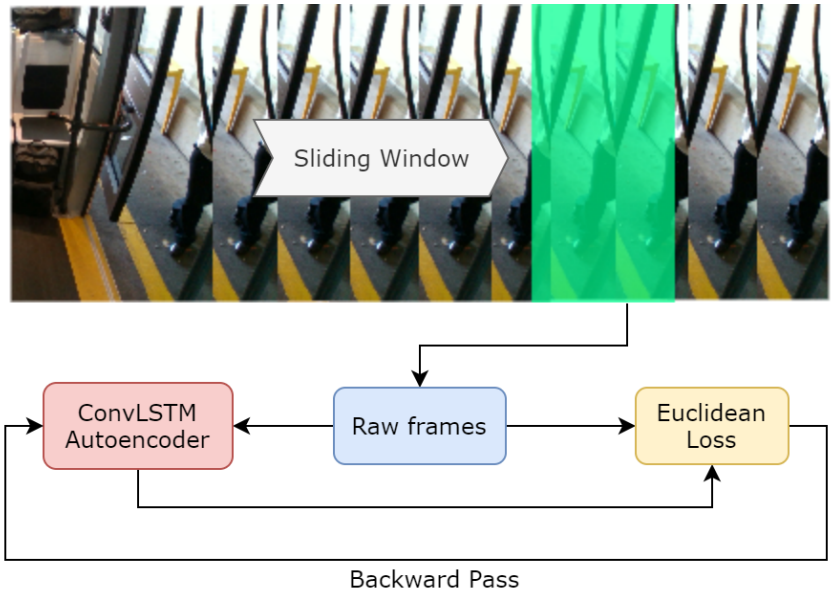

Figure 3.13: Illustration of an abnormal event detection system using a sliding window approach and a ConvLSTM autoencoder for video analysis. The system learns normal patterns from video sequences and identifies anomalies based on the reconstruction error of the autoencoder.

size of the training dataset, we perform data augmentation in the temporal dimension. To generate these volumes, we concatenate frames using sequences, namely being stride-1, stride-2, and stride-3. For example, the stride-1 sequence consists of frame numbers $1, 2, 3, 4, 5, 6, 7, 8, 9, 10$, whereas the first stride-2 sequence contains frame numbers $1, 3, 5, 7, 9, 11, 13, 15, 17, 19$. Now the input is ready for model training.

3.2.3.4 Feature Learning

In order to learn the regular patterns in training videos, we propose a convolutional spatiotemporal autoencoder. Our architecture consists of two parts - a spatial autoencoder for learning spatial structures of each video frame, and a temporal encoder-decoder for learning temporal patterns of the encoded spatial structures. As illustrated in Figure 3.13, the spatial encoder and decoder have two convolutional and deconvolutional layers respectively, while the temporal encoder is a three-layer ConvLSTM model. Convolutional layers are well-known for their superb performance in object recognition, while the LSTM model is widely used for sequence learning and time-series modelling and has proved its performance in applications such as speech translation and handwriting recognition.

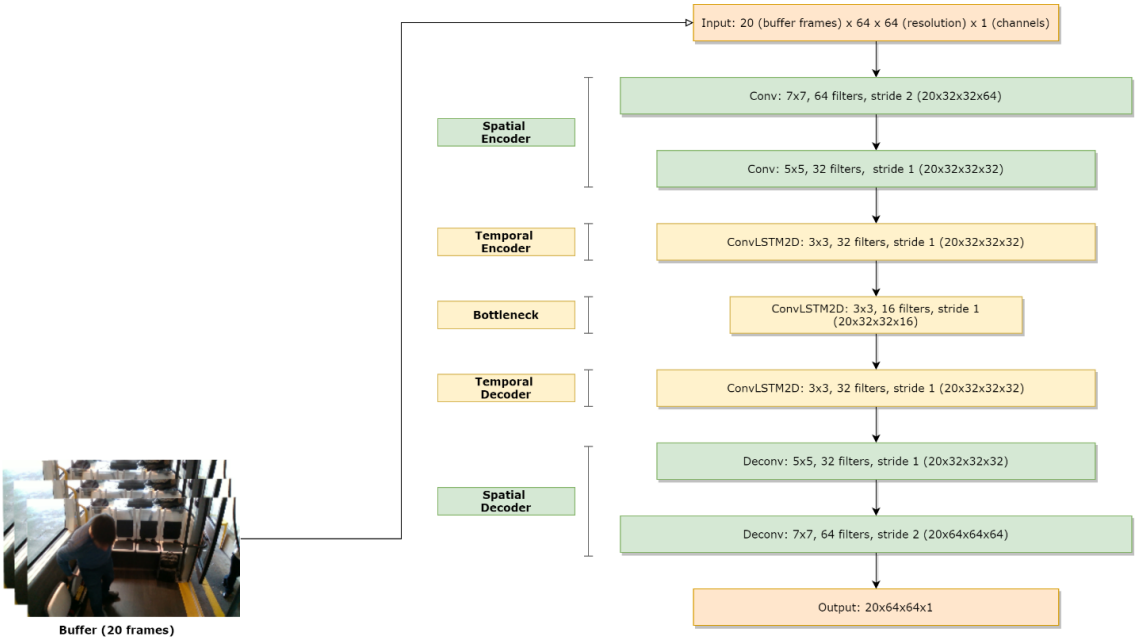

Figure 3.14: Our Autoencoder model pipeline: We start with a buffer of 20 input frames, each with a resolution of 64×64 pixels and 1 channel. The spatial encoder extracts spatial features from the input frames. Then, the temporal encoder captures temporal dependencies between frames. The bottleneck layer further compresses the temporal information. Subsequently, the temporal decoder decodes the temporal information back to its original dimensions. Finally, the spatial decoder reconstructs the output frames with the same resolution as the input.

3.2.3.5 Autoencoder

There are two stages that form an autoencoder: encoding and decoding. Autoencoders set the number of encoder input units less than the input; thus, they were first used to reduce dimensionality. Usually, unsupervised back-propagation is used for training, minimizing the reconstruction error of the decoding results from the original inputs. Generally, an autoencoder can extract more useful features when the activation function is non-linear rather than some common linear transformation methods, such as PCA.

3.2.3.6 Spatial Convolution

The primary purpose of convolution in a CNN is to extract features from the input image. Convolution can preserve the spatial relationships between pixels by using small squares of input data to learn image features. Mathematically, convolution performs dot products between the filters and local regions of the

input. Assuming that we have some n x n square input layer, followed by the convolutional layer, then if we use a $m \times m$ filter W, the convolutional layer output will be of size $(n - m + 1)(n - m + 1)$.

During the training process, a CNN learns the values of these filters on its own, although parameters such as the number of filters, filter size, the number of layers before training still need to be specified. The larger the number of filters used, the more image features get extracted and the better the network becomes at recognizing patterns in unseen images. However, balance is key when it comes to the number of filters used, as more filters would add to computational time and exhaust memory faster.

3.2.3.7 Recurrent Neural Network

In a traditional feedforward neural network, we assume that all inputs (and outputs) are independent of each other. However, in tasks involving sequences, learning temporal dependencies between inputs are important, as e.g., a model of word predictor should be able to derive information from the past inputs. An RNN works just like a feedforward network, except that the values of its output vector are influenced not only by the input vector, but also on the entire history of inputs. In theory, RNNs can make use of information in arbitrarily long sequences, but in practice, due to vanishing gradients, they are limited to looking back only a few steps.

3.2.3.8 Long Short-Term Memory

To overcome this problem, a variant of RNN is introduced: a LSTM model (Figure 3.15) that incorporates a recurrent gate called forget gate. With the new structure, LSTMs prevent backpropagated errors from vanishing or exploding. Therefore, LSTMs can work on long sequences and can be stacked together to capture higher level information.

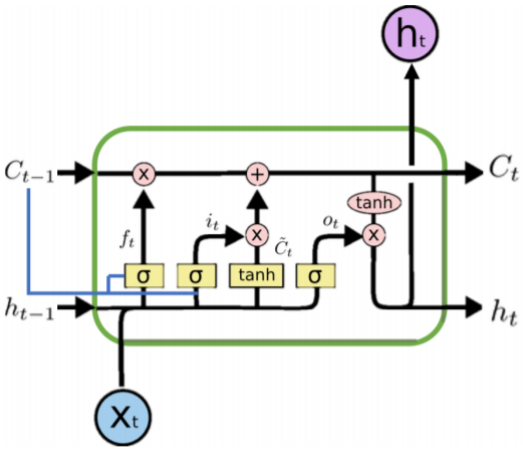

Figure 3.15: The structure of a typical LSTM unit. The blue line represents an optional peephole structure, which allows the internal state to look back (peep) at the previous cell state $C_{(t-1)}$ for a better decision.

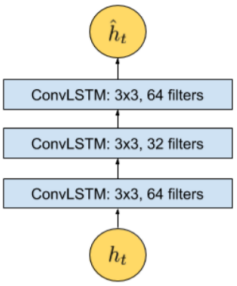

Figure 3.16: The zoomed-in architecture at time t, where t is the input vector at this time step. The temporal encoder-decoder model has 3 ConvLSTM layers.

3.2.3.9 Convolutional Long Short-Term Memory

The Convolutional Long Short-term Memory (ConvLSTM) model, considered a variant of the LSTM architecture, was introduced by Shi et al. in [101] and has been recently utilized by Patraucean et al. [102] for video frame prediction. Compared to the usual fully connected LSTM, ConvLSTM, as shown in Figure 3.16, has its matrix operations replaced with convolutions. ConvLSTM requires fewer weights and yields better spatial feature maps, by using convolution for both input-to-hidden and hidden-to-hidden connections.

3.2.3.10 Regularity Score

We define the reconstruction error as the Euclidean distance across an input and a reconstructed frame. Specifically, the following equations describe the reconstruction error, where t denotes the frame in a sequence:

$$e(t) = |x(t) - f_w(x(t))|^2 \tag{3.5}$$

where f_w is the learned weights by the spatiotemporal model. The irregularity score s_a (t) is calculated by scaling between 0 and 1. Eventually, the regularity score s_r (t) can be derived by subtracting the reconstruction score from 1:

$$s_a(t) = \frac{e(t) - e(t)_{\text{min}}}{e(t)_{\text{max}}} \tag{3.6}$$

$$s_r(t) = 1 - s_a(t) \tag{3.7}$$

Once the model is trained, its performance can be evaluated by feeding in testing data and checking whether it can detect abnormal events while maintaining a low false alarm rate. For a better comparison, we used the same formula as Hasan et al. [103] to calculate the regularity score for all frames. Our only difference is that the learned model is of a different kind. The reconstruction error of all pixel values in a frame of the video sequence is taken as the Euclidean distance between the input frame and the reconstructed frame (Figure 3.16). The reconstruction error or cost of a frame sequence is calculated as the difference between the ground truth (original frames) and the reconstructed frames (prediction – model output). The frame sequence is flagged as "abnormal" if the reconstruction cost exceeds a certain threshold. An example of a prediction with a low regularity score is shown in Figure 3.17.

3.2.3.11 Thresholding

Using a threshold on the reconstruction error, we are able to determine whether a video frame is normal or abnormal. A predefined threshold is not a robust method since our solution should operate in real-time and support multiple camera sensors as we mentioned earlier in the design principles. A fixed

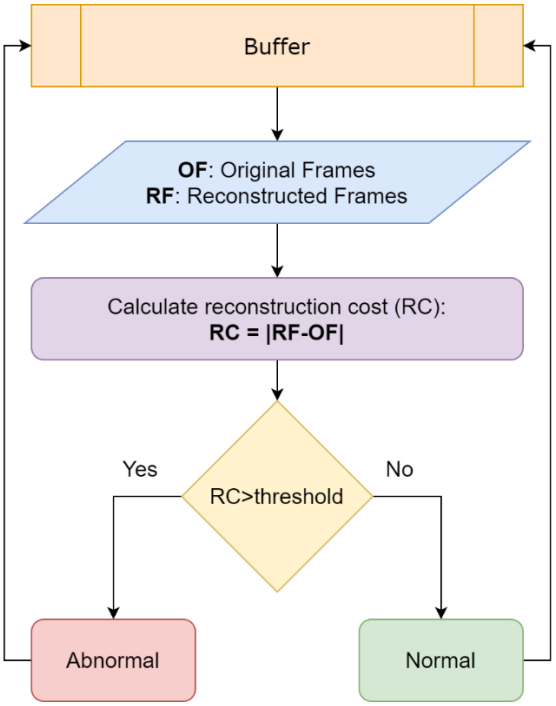

Figure 3.17: Flowchart illustrating the abnormal event detection process using a reconstruction cost-based approach. Original frames are compared to reconstructed frames from an autoencoder, and if the calculated reconstruction cost exceeds a predetermined threshold, the event is classified as abnormal.

threshold value can alter the sensitivity of the event detection, rendering it inappropriate in some scenarios. In addition, a wrong threshold can prevent the detection of certain abnormal events or produce false positives. In order to solve this issue, we introduce a variable thresholding technique in order to find the optimal value in real-time. The initialization procedure now includes a "warm-up" session, in which we aggregate the individual regularity score of each frame. During that session, no detections are performed, as we consider the events as regular. As the buffer continues to fill, we are able to calculate the average reconstruction error and provide a threshold value tailored to the specific conditions. Figure 3.17 indicates an abnormal scenario with the aforementioned metrics.

3.2.4 Hybrid LSTM Classification

Hybrid LSTM classification offers a sophisticated solution by combining the strengths of convolutional autoencoders (CAEs) and RNNs. This approach begins by training a CAE on regular data, allowing it to learn a compressed repre-

Figure 3.18: Example of a prediction with a lower regularity threshold. Red blocks in the image represent a high reconstruction cost.

sentation that captures essential features while discarding noise. This encoder is then integrated with a LSTM network, forming a hybrid architecture that excels at sequence modeling. The resulting model is fine-tuned on specific classification tasks, leveraging the encoder's ability to extract meaningful features and the LSTM's capacity to model temporal dependencies.

This approach works with the principle of having an unbalanced dataset with rare anomalous events. Therefore, it is possible to manually go through the anomaly outputs and flag some of them as false positives. In this way, we can let the previous autoencoder neural network model act as a High Recaller.

3.2.4.1 Semi-supervised Learning

The threshold is decreased so that almost all the actual anomalies are detected (high recall) along with other false positive anomalies (low precision). To achieve the semi-supervised approach, we designed a new model which includes the previous Encoder and an LSTM which acts as a classifier.

In real-time inference the anomalies predicted by the high recaller model (autoencoder neural network) are sent through the false positive reduction model (hybrid model). This combination of neural networks (Figure 3.19) should

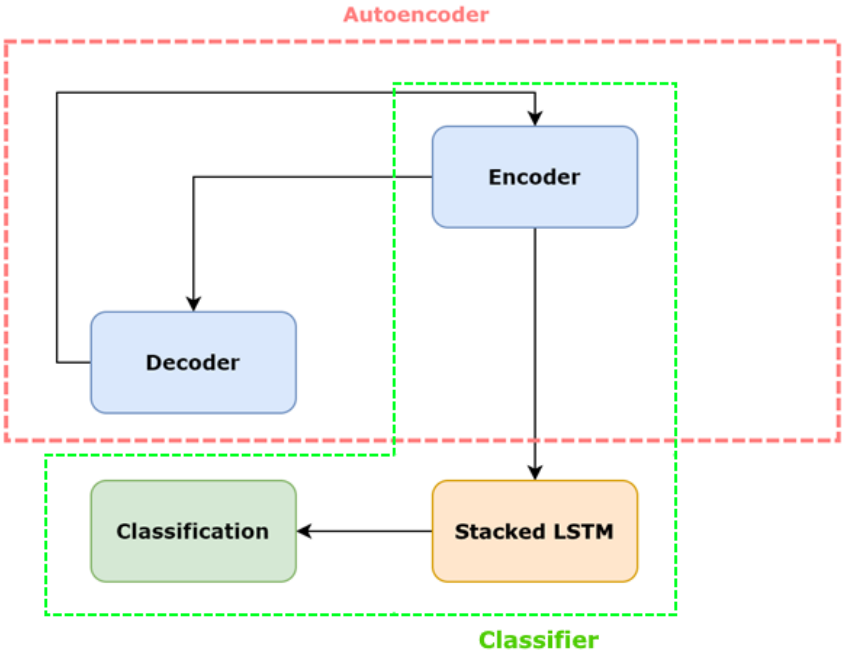

Figure 3.19: Model architecture of the hybrid model. The red box contains components of the previous autoencoder approach. The green components indicate the new hybrid model which acts as a classifier.

provide a deep neural network model with high recall and high precision.

3.2.4.2 Training

The training process of the new experiment consists of 3 stages and it is depicted in Figure 3.20. At first, the autoencoder model (encoder + decoder) is being trained with unsupervised data to learn regularity. In the second stage the encoder weights are transferred to the hybrid model. The encoder's layers are marked as non-trainable. Finally, we perform supervised training using only the LSTM classifier.

3.2.5 Two-stream Action Classification

Another challege in action classification is the varying duration of the detected action. For this reason, capturing both the subtle movements and the rapid dynamics of motion is crucial. Two-stream action classification, named as

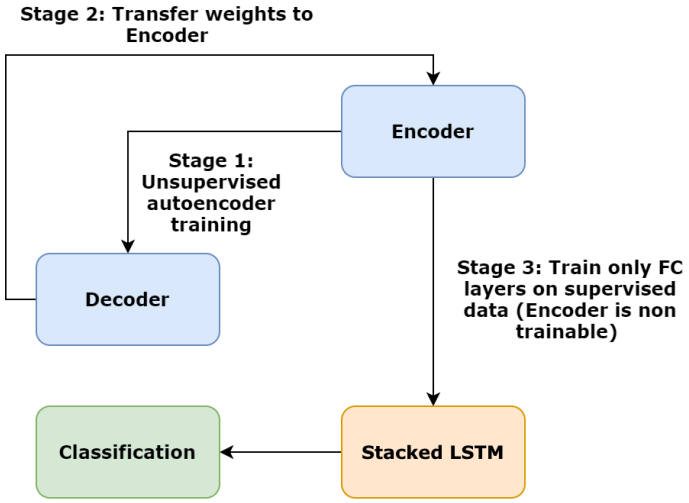

Figure 3.20: A three-stage hybrid training approach for classification tasks. Stage 1: Unsupervised pre-training of an autoencoder on unlabeled data. Stage 2: Transfer of learned encoder weights to a separate encoder module. Stage 3: Supervised training of the classifier using the fixed encoder on labeled data.

SlowFast, addresses this challenge by introducing a dual-pathway architecture. The "Slow" pathway focuses on analyzing the spatial information in individual frames at a lower frame rate, extracting detailed appearance features. Simultaneously, the "Fast" pathway operates at a higher frame rate, capturing the temporal evolution and motion patterns present in the video. This separation of pathways allows the model to effectively model both the static and dynamic aspects of visual content, leading to improved performance in action recognition tasks.

The SlowFast algorithm is based on 3-dimensional well-known convolutional networks ResNets that investigate contrasts of action speed along the time axis of frames. The algorithm consists of two parallel streams with different framerates: a slow pathway and a fast pathway as Figure 3.21 presents. These two pathways own some significant different temporal speeds, able to recognize action in a sequence of frames. In particular, the fast pathway was designed to understand fast changing motions, but with fewer spatial details. On the other hand, the slow pathway was designed to recognize slow changing motions, but with a high level of spatial details. The SlowFast algorithm was configured to process rectangle RGB images of 224×224 dimensions. Moreover, Adam optimizer with 0.001 learning rate was selected to minimize the categorical cross-entropy loss

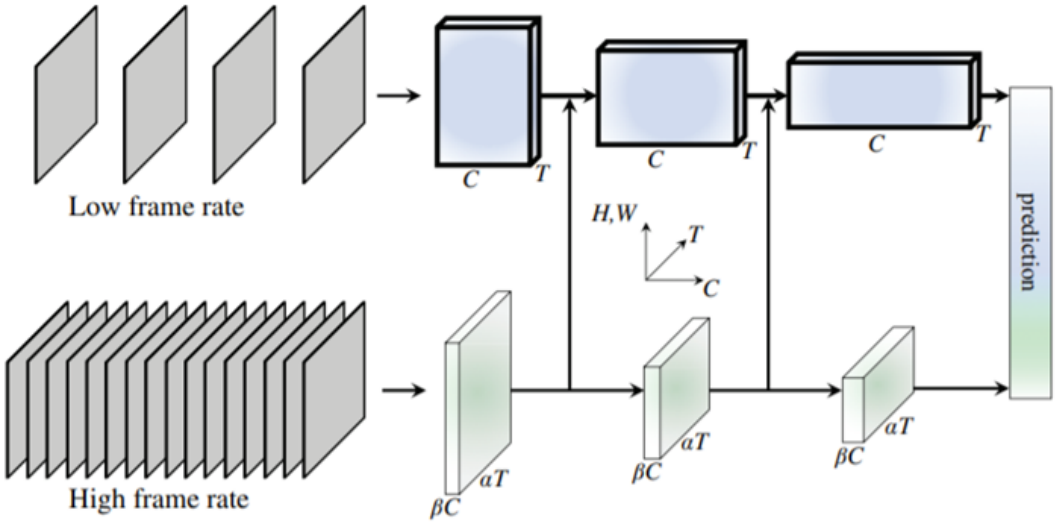

Figure 3.21: The network employs two pathways: a slow pathway that processes LFR RGB frames to capture spatial semantics, and a fast pathway that processes HFR RGB frames at a lower resolution to capture temporal information. The two pathways are then fused together before making a final prediction. C denotes convolution operation, T denotes temporal pooling/sampling operation, α is the frame rate ratio between the slow and fast pathway, β is the channel ratio between the slow and fast pathway, HW is the spatial size of the feature map, and lateral connections are used for feature fusion between the two pathways.

function. The SlowFast trained for a large number of epochs, equal to 300 aiming to allow the neural network to learn from scratch since it does not employ pre-trained weights.

3.2.6 Overhead Abnormal Event Detection

The ability to detect abnormal events from an overhead perspective offers a unique advantage for monitoring large areas but it comes with the significant challenge of barrel distortion. Overhead abnormal event detection using fisheye cameras addresses this need by leveraging the wide field of view and distortion characteristics of fisheye lenses. These cameras capture a panoramic view of a scene, allowing for the observation of a wider area with fewer cameras compared to traditional lenses. The distortion introduced by fisheye lenses can be corrected using specialized algorithms, enabling accurate object detection and tracking. By analyzing the motion patterns, object interactions, and scene dynamics captured by fisheye cameras, this approach can identify unusual events.

For this reason, our approach utilizes a CAE with two main parts: an encoder, which compresses the input data into a lower-dimensional latent space;

and a decoder, which reconstructs the data from the latent space back to the original input space. The CAE learns to capture the important features of the input data while removing noise or redundancy.

The encoder part of the CAE consists of a series of convolutional layers that apply learned filters to the input data. Convolutional layers are particularly effective for extracting extracting spatial features from images due to their ability to learn hierarchical patterns. The encoder's architecture can be formalized as a function:

$$h_{\text{enc}}(x) = f(W_{\text{enc}} * x + b_{\text{enc}}) \tag{3.8}$$

where x is the input data, W_{enc} represents the weights of the convolutional filters, b_{enc} denotes the biases, $*$ indicates the convolution operation, and f is a non-linear activation function such as ReLU or Sigmoid.

The ConvLSTM2D layer is a recurrent layer that processes data in both space and time. It is specifically designed for problems where the context in both dimensions is crucial, such as videos. The layer not only applies a convolution operation to the input data, but also maintains a hidden state that captures temporal information. The ConvLSTM2D layer can be expressed mathematically as:

$$h_t, c_t = \text{ConvLSTM2D}(h_{t-1}, c_{t-1}, x_t) \tag{3.9}$$

where h_t is the hidden state at time t, c_t is the cell state at time t, and x_t is the input at time t.

The decoder mirrors the encoder structure, but uses deconvolutional (transposed convolution) layers to reconstruct the input data from the latent space representation. The reconstruction can be quantified using the proposed Center Weighted Loss (CWL) loss.

3.2.6.1 Center-Weighted Loss

Regarding our loss function, we combined the traditional MSE loss with a spatial weighting mechanism that assigns higher weights to pixels closer to the center of the image. This weighting can help the autoencoder focus more on accurately reconstructing the central part of the image, which has higher impact based on the distribution of passengers in the cabin space. The proposed loss function is

defined as follows:

$$L(Y, \hat{Y}) = \frac{1}{N} \sum_{i=1}^{H} \sum_{j=1}^{W} w(i,j)(Y_{ij} - \hat{Y}_{ij})^2 \tag{3.10}$$

where $L(Y, \hat{Y})$ is the loss function, Y is the ground truth image, $\hat{Y}$ is the reconstructed image produced by the autoencoder, and H and W are the height and width of the images, respectively. The weight function $w(i,j)$ is defined using a Gaussian distribution:

$$w(i,j) = \exp\left(\frac{-(i-i_c)^2 - (j-j_c)^2}{2\sigma^2}\right)$$

With (i_c, j_c) representing the coordinates of the center of the image and σ controlling the spread of the Gaussian function. The normalization factor N, often the total number of pixels in the image, is used to keep the loss value scale consistent.

Incorporating the Gaussian function into the weighting mechanism allows for a smooth transition of importance from the center towards the edges of the image, which aligns with the goal of enhancing focus on the central areas during the autoencoder's training process.

3.2.6.2 Training Stages

At the first stage (Figure 3.22), the CAE is trained on regular regular events using back-propagation to minimize the reconstruction loss. The process optimizes the weights and biases to capture the regular patterns of the input data.

After the initial training, a second stage supervised training follows with a slightly modified architecture (Figure 3.23). In this training stage, we use the encoder part of the model along with a classification head for supervised training.

We perform fine tuning via transfer learning using the weights from the previous training session, which can ensure the ability of the encoder to compress critical features to the latent space to improve the robustness of our method. The head is trained using categorical cross-entropy loss:

$$\mathcal{L}_{CE} = -\sum_{c=1}^{M} y_{o,c} \log(p_{o,c}) \tag{3.11}$$

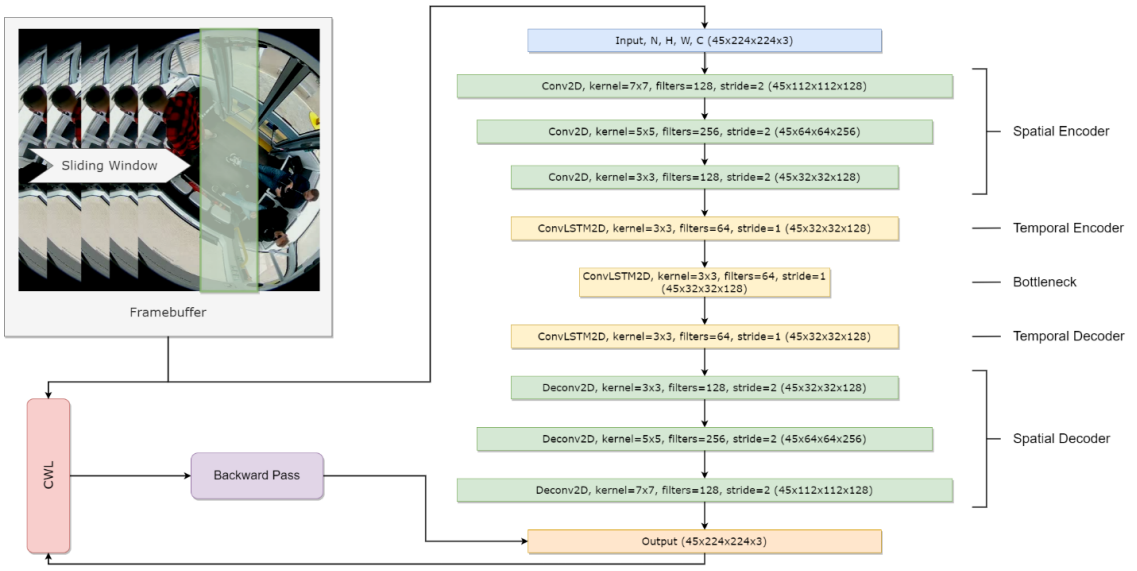

Figure 3.22: Model architecture of the autoencoder: the convolutional layers are spatial encoders, followed by temporal encoder and decoder. Bottleneck compress the features to eliminate non useful information. At the end, we perform spatial decoding, reconstructing the input image to the same format.

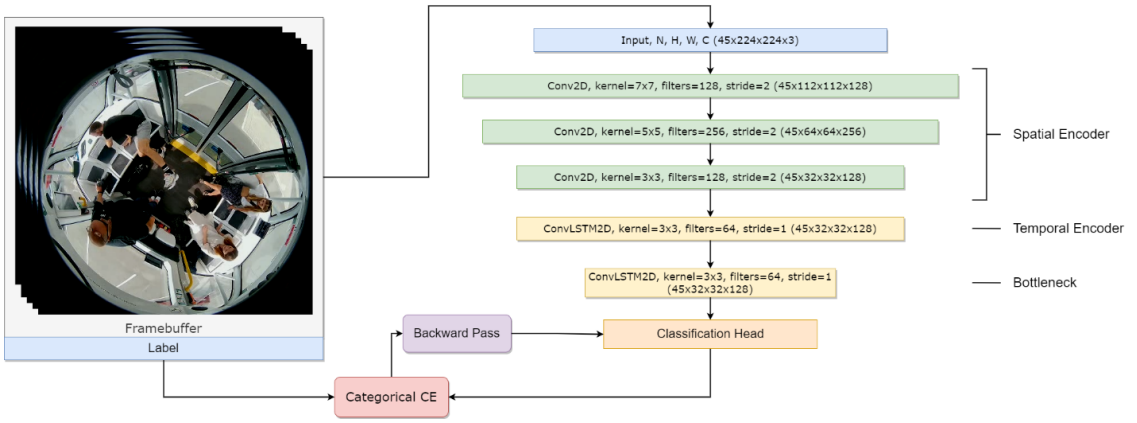

Figure 3.23: Fine-tuning hybrid classifier: the pretrained encoder weights from the previous stage are transferred and a classification head is added for detecting the abnormal events.

where M is the number of classes, $y_{o,c}$ indicates the presence of class c in observation o, and $p_{o,c}$ is the predicted probability of class c for observation o.

The two-phase training strategy of the CAE allows the model to be capable of distinguishing irregular changes in the sequences and then to identify deviations from this baseline as anomalies. The integration of ConvLSTM2D layers enables the model to capture temporal dependencies in addition to the spatial features learned by the Conv2D layers. This method provides a robust approach to anomaly detection in video sequences.

3.2.7 Overhead Object Detection

Side cameras, while useful for capturing object details, have a limited field of view and are prone to occlusions, making them less effective for comprehensive object detection. Overhead cameras, on the other hand, offer a wider field of view, minimizing occlusions and providing a more comprehensive perspective of the scene. This makes them ideal for detecting and tracking multiple objects, especially in crowded or complex environments. Additionally, the consistent perspective of overhead cameras simplifies the comparison of object positions and movements across different frames. Due to these advantages, we prioritized overhead cameras for our object detection system, ensuring greater accuracy and reliability, particularly in scenarios where monitoring large areas or tracking multiple objects is essential.

Inspired by RAPiD [104], the detection network consists of three stages: the backbone network, the feature pyramid network (FPN), and the bounding box regression network. The backbone network works as a feature extractor that takes an image as input and outputs a list of features from different parts of the network. In the next stage, we pass those features into the FPN, in order to extract features related to object detection. Finally, at the last stage, a CNN is applied to each feature vector in order to produce a transformed version of the bounding-box predictions (Figure 3.24).

As presented in the diagram above, the backbone of the network is responsible for the initial feature extraction and is typically a pre-trained CNN such as ResNet, VGG, or a similar architecture. More specifically, given an input image I, the network produces a set of multi-dimensional feature maps $\{P_k\}_{k=1}^{3}$ at different scales, denoted as P_1, P_2, and P_3 for high, medium, and low resolutions, respectively, as follows:

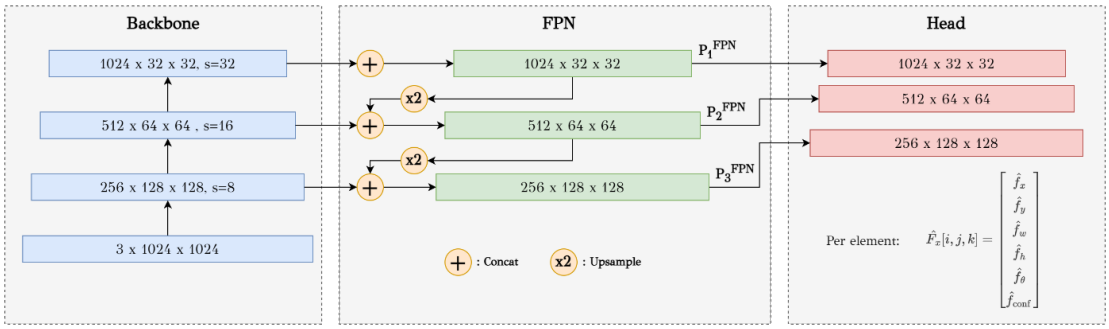

Figure 3.24: Overhead object detection model architecture. Each arrow represents multiple convolutional layers, and the colored rectangles represent multi-dimensional matrices, i.e., feature maps, whose dimensions correspond to an input image of size $h \times w = 1024 \times 1024$.

$$P_k = \text{Backbone}(I), k \in \{1, 2, 3\}$$

The FPN enhances the backbone's feature maps by integrating high-level semantic information from deep layers with spatial information from earlier layers based on Eq. (3.2.7). This choice was made due to the significant variations in object sizes due to the distortion of the wide-angle fisheye lenses. FPN's multi-scale feature representations allow for effectively detecting both passengers in the center, who appear larger, and those who appear much smaller due to the peripheral distortion. This is achieved by upsampling spatially coarser, but semantically stronger, feature maps from higher pyramid levels and merging them with lower-level maps through element-wise addition.

$$P_k^{\text{fpn}} = \text{FPN}(P_k), k \in \{1, 2, 3\}$$

Post feature enhancement by the FPN, the detection head predicts the transformed bounding boxes and class scores by applying a separate CNN to each FPN feature vector to generate a transformed bounding box predictions $\overline{T_k}$ and an objectness score, which are relatively parameterized to anchor pre-defined boxes at different scales and aspect ratios, as follows:

$$t_{x,k} = s_k(i + \text{Sigmoid}(f_{x,k}))$$

$$t_{y,k} = s_k(j + \text{Sigmoid}(f_{y,k}))$$

$$t_{w,k} = w_k^{\text{anchor}} \exp(f_{w,k})$$

$$t_{h,k} = h_k^{\text{anchor}} \exp(f_{h,k})$$

$$\tilde{o}_k = \text{Sigmoid}(f_{o,k})$$

where i and j represent the center of the anchor box, s_k is the stride of the feature map at scale k, and $f_{x,k}, f_{y,k}, f_{w,k}, f_{h,k}, f_{o,k}$ are the outputs of the CNN applied to the feature vector from the FPN. The variables w_k^{anchor} and h_k^{anchor} represent the width and height of the k-th anchor box, and Sigmoid is the logistic function applied to constrain the outputs to a range between 0 and 1. The objectness score $\tilde{o}_k$ denotes the probability that an object is present within the predicted bounding box. The training loss function of the network combines the regression loss for the bounding box coordinates and a classification loss for the objectness score.

3.2.7.1 Rotation-Aware Bounding Box Regression

The rotation-aware bounding box regression is a critical component of the proposed method, enabling the detection of people in overhead fisheye images with a high degree of accuracy in both position and orientation by incorporating a novel angle prediction mechanism that accounts for the unique properties of angles as cyclic quantities. It involves the prediction of a bounding box's orientation, along with its center and size, which are normalized relative to the feature map dimensions as follows:

$$L_{reg} = \sum_{i}^{N} (\lambda_{coord} \text{Scale}_{L_1}(b_i, \hat{b}_i) + \lambda_{angle} \text{Periodic}_{L_1}(\theta_i, \hat{\theta}_i)) \tag{52}$$

Here, L_{reg} represents the regression loss, b_i the predicted bounding box, $\hat{b}_i$ the ground truth box, θ_i the predicted orientation, and $\hat{\theta}_i$ the ground truth orientation. The terms Scale_{L_1} and Periodic_{L_1} are the scale L1 loss and periodic L1 loss, respectively, with λ_{coord} and λ_{angle} as the balancing weights. This allows the network to effectively learn the orientation of objects, taking into consideration the periodic nature of the angle, thus providing a more robust and accurate object detection in fisheye images.

3.2.7.2 Scale-Invariant Loss Function

To address scale variance in object detection, particularly for fisheye images, we propose the integration of a scale-invariant term in the loss function that aims to

stabilize the learning across different object sizes, a common challenge in fisheye image datasets due to perspective distortion. The scale-invariant loss term L_{scale} could be formulated as follows:

$$L_{\text{scale}} = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{w_i h_i} L_{\text{obj}}(o_i, \hat{o}_i) \tag{53}$$

where N is the number of objects, w_i and h_i are the width and height of the bounding box of the i-th object, L_{obj} is the objectness loss for the i-th object, o_i is the ground truth objectness score, and $\hat{o}_i$ is the predicted objectness score.

3.2.7.3 Angle Loss Function

The angle loss is crucial for the model's ability to learn the orientation of objects, a fundamental aspect when dealing with fisheye images. It ensures accurate angle predictions for bounding boxes by incorporating binary cross-entropy (BCE) for foreground-background classification with a specialized term for angle regression as follows:

$$\begin{aligned} L_{\text{angle}} = & \sum \text{BCE}(\sigma(t_{\text{cls}}), y_{\text{cls}}) \\ & + \sum \text{BCE}(\sigma(t_{\text{obj}}), y_{\text{obj}}) \\ & + \sum \lambda_\theta L_{\text{periodic}}(\theta_p, \theta_g) \end{aligned} \tag{3.12}$$

where L_{angle} is the composite loss for classification and angle prediction, σ the sigmoid function, t_{cls} the class logits, t_{obj} the objectness logits, y_{cls} and y_{obj} the class and objectness labels, λ_θ the weight for angle loss, L_{periodic} the periodic angle loss, θ_p the predicted angle, and θ_g the ground truth angle. This harmonized loss function facilitates the network to learn not only the presence of an object, but also its precise rotational alignment.

3.2.7.4 Periodic Angle Prediction Loss

The periodic angle prediction loss mitigates angle discontinuity by employing a periodic loss. The network learns to effectively predict angles in a rotation-invariant manner, which is critical for maintaining consistency when angles form

a full circle.

$$L_{\text{periodic}}(\theta_p, \theta_g) = \min_{k \in \mathbb{Z}} f(\theta_p - \theta_g + 2\pi k)$$

Here, θ_p is the predicted angle, θ_g is the ground truth angle, and f is a distance metric, such as the L2 norm. The loss function ensures smooth transitions across the angle boundary, effectively treating angles 2π radians apart as equivalent. The minimization over k accounts for the multiple equivalent representations of the same angle due to periodicity, thereby encouraging the network to learn angle predictions that are robust against rotational variances.

3.2.8 Facial Identification

Facial verification offers a robust and secure method for confirming an individual's identity, gaining traction due to its non-intrusive nature and high accuracy. Among the various techniques, we chose Siamese networks for their unique advantages in facial verification. These networks excel in learning a similarity metric between image pairs, making them ideal for verification tasks. Their architecture, consisting of two identical subnetworks sharing weights, enables effective learning of discriminative features that distinguish between individuals while being robust to variations in pose, lighting, and expression. Additionally, Siamese networks require fewer training examples compared to traditional models, making them suitable for scenarios with limited labeled data. In the context of facial verification, Siamese networks have demonstrated superior performance in accurately determining whether two face images belong to the same person. Their ability to learn robust representations and compare them effectively makes them a compelling choice for enhancing security and user experience in various applications.

The process begins with two input images, Input A (the database image) and Input B (the image cropped from the video stream). These images are preprocessed and then fed into a shared backbone feature extractor. The backbone used here is the same CNN feature extractor implemented in ArcFace, with one modification: the last fully connected layer and the preceding flatten layer are replaced with a 1×1 convolutional layer to accommodate the $xCos$ module.

The core of the system lies in the $Lcos$ calculation using the $xCos$ module. This module calculates patch-wise cosine similarities between the feature maps

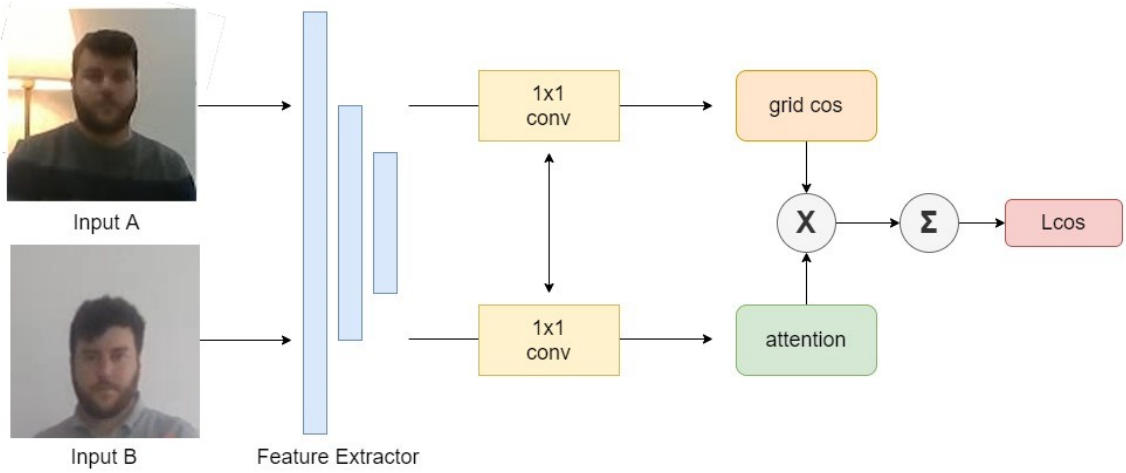

Figure 3.25: Two facial images are processed by a feature extractor to extract features. The features undergo two parallel processes: a) grid-based cosine similarity, measuring facial feature similarity in corresponding grid cells, and b) an attention mechanism, emphasizing relevant facial regions. The results of these processes are combined and summed to calculate a loss value, which assesses the accuracy of the system in determining if the two faces belong to the same individual.

generated by the backbone for both input images. These similarities are then element-wise multiplied by attention maps. The resulting values are summed up to compute the final $Lcos$, a metric representing the overall similarity between the two input images.

As current face verification models use fully connected layers, spatial information is lost along with the ability to understand the convolution features in a human sense. To address this obstacle, the plug-in $xCos$ module is integrated as described below and presented in Figure 3.25:

- **Input**: The two input images are preprocessed and passed into the feature extractor. The Input A is the database image while the Input B is the image cropped from the video stream.
- **Backbone**: We implement the same CNN feature extractor as in ArcFace [18]. However, in order to employ the $xCos$ module, the last fully connected layer and the previous flatten layer are replaced with a 1×1 convolutional layer.
- $Lcos$ **calculation (**$xCos$**)**: Patch-wise cosine similarity is multiplied by the attention maps and then summed to calculate the $Lcos$.

3.2.9 Audio Classification

Audio surveillance plays a critical role in detecting anomalies, as sound often provides valuable clues about ongoing events. However, the vast amount of audio data generated by surveillance systems necessitates automated methods for detecting abnormal events. Audio classification for abnormal event detection addresses this need by enabling machines to recognize and categorize sounds, distinguishing normal occurrences from potentially dangerous or unusual events.

In this Section, we will describe the CNN architectures used, the optimizer that was selected and the activation function between the convolutions and max-pooling operations. CNNs are known to perform well in image classification and have achieved similar results in the field of audio-based event detection. In general, the input image is passed through various convolutional layers, before it is flattened and fed to a fully connected neural network that outputs the probabilities of the target classes.

Initially, the image is passed through a convolutional layer, which is meant to reveal the structures and shapes in the image. The way that this is performed is to introduce a filter that slides over the entire image and multiplies with all the pixels (with overlap). Every time the filter is multiplied with a set of pixels, we sum all the multiplications and add the value to an activation map (convolution operation). The activation map is completed after the filter is multiplied with the entire image. A typical filter has dimensions of 16×16 or 32×32, depending on the shape of the input image. It is important in this case, to choose a filter that is large enough to cover all the structures of the image. A pooling layer is a way to reduce dimensionality of the representation (down-sampling) such that there are not many parameters that need optimization, but it also helps to control overfitting. The activation map is divided into regions of equal size and represents each region with one single number. Max-pooling is one of the most popular pooling techniques, which just represents each region with the largest number in that region. However, it is possible to use average pooling, global pooling, etc. To increase the stability of a neural network, batch normalization normalizes the output of a previous activation layer by subtracting the batch mean and dividing by the batch standard deviation. Consequently, batch normalization adds two trainable parameters to each layer, so the normalized output is multiplied by a "standard deviation" parameter (γ) and add a "mean" parameter (β). Dropout is widely used in neural network layers and is another way to prevent overfitting. This layer is simple in the sense that it randomly

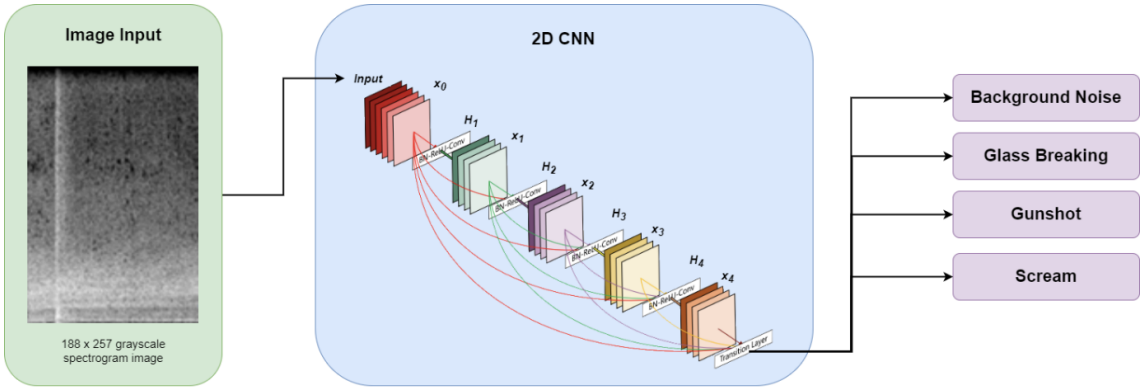

Figure 3.26: Audio classification approach using a 2D Convolutional Neural Network (CNN). A spectrogram image, representing frequencies over time, is input into the CNN. The network processes the image through multiple convolutional layers (X1 to X4), each extracting features. The output layer then classifies the audio into different categories: background noise, glass breaking, gunshot, or scream.

drops out units (activation maps) in the current layer by setting them to zero. Finally, the last part of a CNN is often referred to as a fully connected layer, which is a regular feed-forward neural network (FNN). The output from the other layers needs to be flattened out before it is passed into the FNN. The activation function that is commonly used for classification is Softmax. This activation function assigns probabilities between the target classes and the probability that is closest to one is selected. For audio classification a DenseNet-121 CNN architecture is used as seen in Figure 3.26.

DenseNets [105] were introduced in order to solve the problem of the vanishing gradient in neural networks. The vanishing gradient occurs when the CNNs become big that the path for information from the input layer to the output layer significantly increases that the gradient vanishes during back-propagation. In DenseNets, we connect every layer directly with each other. This process ensures maximum information kept throughout the architecture. Additionally, by using this connection the DenseNets require fewer parameters to train than vanilla CNNs, since there is no need to learn redundant feature maps.

The DenseNet-121 was used for classification, as the most basic DenseNet yet powerful architecture. Each dense layer consists of two convolutional operations as follows:

- 1×1 Convolutions (for feature extraction)
- 3×3 Convolutions (bringing down the feature depth/channel output)

The DenseNet-121 consists of six such dense layers in a dense block. This resulted in approximately seven million parameters, compared to the 44 million

parameters of a vanilla CNN architecture.

After the non-linear function describing the dataset is found, one typically wants to find the minimum or maximum to optimize various parameters. In order to perform that there are numerical optimization methods. In our experiments, we have used the Adam optimization method with an initial learning rate of 0.001. Adam is often the preferred optimization method in machine learning, due to its computational efficiency, little memory occupation and implementation ease. Adam is a momentum-based method that only relies on the first derivative of the cost function. Required inputs are exponential decay rates $\beta 1$ and $\beta 2$, cost function $c(\theta)$ and initial weights θ, in addition to the learning rate η and eventually a regularization λ. Iteratively, the gradient of the cost function is calculated and this is used to calculate the first and second moment estimates.

The activation function is used to activate the outputs, which often need to take some certain properties. For example, when doing classification, the outputs are probabilities and therefore take values between zero and one. A nonlinear activation function is often preferred to reinforce the non-linearity of the neural network. Traditionally, the logistic function and the tanh function have been used as activation functions, since they were believed to work in the same way as the human brain. However, in 2012 Alex Krizhevsky et al. introduced AlexNet [106], taking image recognition to a new dimension. They used a ReLU function, which is zero for negative values. The ReLU function is a modification of the pure linear function for positive values. This makes the function able to recognize the non-linearity in the model, providing it a "clean" derivative given by the step function.

3.3 Framework Architecture

For our development platform, we selected the NVIDIA Jetson AGX Xavier due to its unique combination of high-performance computing capabilities and energy efficiency. Its powerful GPU and specialized AI accelerators enable us to run complex DL models in real-time, meeting the computational demands of our object detection and classification tasks. Moreover, the Jetson AGX Xavier's compact form factor and low power consumption make it ideal for deployment in edge devices, allowing us to process data directly on the device without relying on cloud-based resources. This not only reduces latency but also enhances

privacy and security. Additionally, the Jetson AGX Xavier's extensive software support, including the JetPack SDK and various DL frameworks, streamlines the development process and facilitates seamless integration with our existing workflows.

Although optimized for the NVIDIA Jetson platform, our system is adaptable to similar hardware configurations, requiring at least a GPU with CUDA support, 16 GB RAM and 32 GB disk space. Software prerequisites include an Linux-based Ubuntu OS, Python and OpenCV. The installation process involves system package installations, building the v4l2loopback [107] device for creating virtual camera devices and installing necessary Python packages. The structure comprises various directories for assets, common code, service modules, shared modules, utilities, a configuration file and a launcher script. The configuration settings are adjustable in the config.py file, which includes constants like shared modules, service modules, vehicle ID, camera sources and parameters for different services like passenger counting, object detection and face verification. The system is launched by spawning service modules as separate processes and allowing inter-process communication (IPC) through the av_ipc module. The dashboard, accessible locally or over a network, displays video feeds, audio spectrograms, status of different modalities, event history, environmental metrics, passenger counts and other crucial data related to security and monitoring in AVs. The framework is designed to be robust, adaptable and user-friendly, catering to the complex needs of autonomous vehicle control and monitoring systems.

3.3.1 Interfaces

Seamless interaction between software and hardware is crucial for our framework. To achieve this, well-defined interfaces to hardware are essential. These interfaces serve as bridges, enabling software applications to communicate with and control various hardware components, such as sensors, actuators, and processing units.

Our framework integrates an advanced hardware abstraction layer (HAL), designed to interface seamlessly with a diverse array of sensors, thus providing a versatile and comprehensive sensory platform. This layer functions as a crucial intermediary, standardizing communications and data exchange between the framework's core processing units and various external hardware devices. This

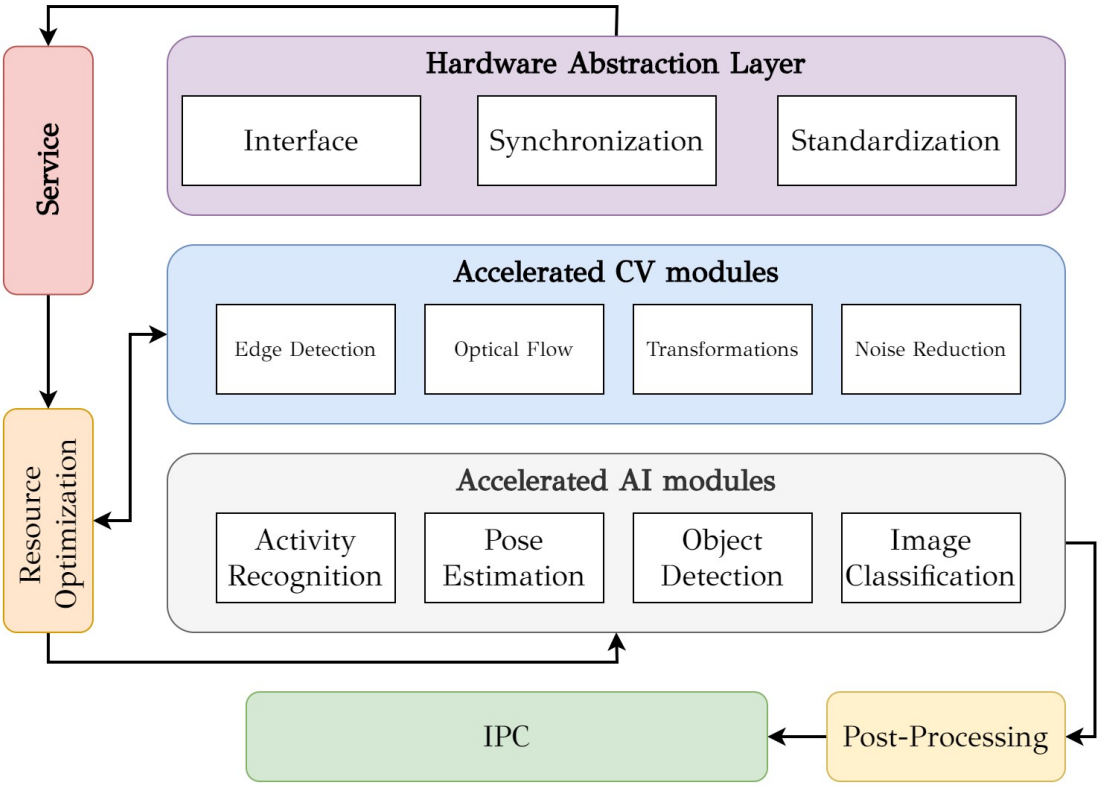

Figure 3.27: Framework illustration diagram for hardware acceleration of computer vision and AI tasks. A HAL provides interface, synchronization, and standardization for the underlying hardware. The accelerated CV modules encompass edge detection, transformations, and noise reduction. The accelerated AI modules include activity recognition, pose estimation, object detection, and image classification. IPC facilitates communication between the modules, and post-processing refines the final output. Resource optimization and service feedback loops ensure efficient utilization of the hardware resources.

level of abstraction not only simplifies integration, but also enhances the system's adaptability and scalability. For visual data acquisition, the HAL supports a range of camera types. It is capable of interfacing with IP cameras using multiple protocols such as RTSP, HTTP and ONVIF, ensuring compatibility with a wide array of network-based camera systems. This flexibility allows the framework to be deployed in environments with pre-existing surveillance infrastructure, or in scenarios requiring remote or distributed monitoring setups.

Additionally, the HAL's support for USB cameras facilitates plug-and-play connectivity, making it ideal for portable or temporary setups where ease of deployment is key. In terms of audio capture, the HAL integrates USB microphones, allowing for high-quality audio recording. This feature is particularly useful in environments where audio data provides critical context or cues, such as in security applications or interactive systems requiring voice recognition.

Moreover, the HAL extends its support to a variety of USB sensors, enriching the framework's sensory capabilities. This includes advanced devices, including Lidar for precise spatial mapping and navigation, stereo cameras for depth perception essential in 3D modeling or autonomous navigation systems and depth cameras, that provide detailed environmental awareness crucial for obstacle detection and avoidance. The inclusion of ultrasonic sensors further augments the system's spatial understanding, enabling applications in proximity detection and collision avoidance. The HAL's design ensures that data from these diverse sources is standardized into a format that is readily consumable by the framework's processing modules. This standardization streamlines the data processing pipeline, facilitating efficient and accurate data analysis and decision-making processes. Furthermore, the HAL is engineered with a focus on extensibility and modularity. This approach allows for the easy integration of new types of sensors as technology evolves, ensuring that the framework remains future-proof. Additionally, this modular design enables customization to meet specific needs or constraints of various applications, ranging from AVs and industrial automation to smart home systems and environmental monitoring.

In addition to its comprehensive sensor support, the framework's HAL incorporates an advanced data synchronization mechanism, crucial for coordinating inputs from multiple sensors. This synchronization ensures that datastreams from diverse sources, such as IP cameras, USB microphones and various other sensors, are aligned in time, providing a coherent and unified view of the environment. This temporal alignment is vital in scenarios where the timing of events captured by different sensors is critical, such as in security applications,

autonomous vehicle navigation, or complex robotic systems. The synchronization module intelligently manages time-stamps, handles latency variations and aligns data sequences, thereby maintaining the integrity and relevance of the information being processed. This capability not only enhances the accuracy and reliability of the system's analyses and responses, but also enables more sophisticated functionalities, including event detection and multimodal data fusion, where the interplay of sensory data provides deeper insights than isolated sensor readings.

In summary, this proposed framework's HAL stands as a proof to modern engineering, offering a harmonious blend of versatility, compatibility and forward compatibility. It serves as a robust foundation, enabling the framework to effectively harness the full potential of various sensory inputs, thereby paving the way for sophisticated and intelligent system applications.

3.3.2 Streaming to Virtual Devices

The framework makes use of the v4l2loopback [107] module, which is a kernel module for Linux, designed to create virtual video loopback devices. It allows for the creation of virtual video devices that normal Video4Linux2 (v4l2) applications can interact with as if they were standard video devices. However, unlike typical video devices that capture video from hardware like cameras, the video in these virtual devices is provided by other applications. The v4l2loopback module plays a crucial role particularly in scenarios requiring the handling of multiple video streams or the integration of various video sources. The v4l2loopback driver is employed to create virtual video devices, enabling the manipulation and redirection of video streams from physical devices like cameras to other applications or processes.

At its core, v4l2loopback functions as a kernel module for Linux systems, designed to facilitate the creation of virtual video loopback devices. These virtual devices act as standard video devices from the perspective of Video4Linux2 (v4l2) applications, but instead of capturing video from physical hardware like a camera, they receive video data from other applications. This feature is critical in scenarios where multiple applications need to access the same video stream simultaneously or when the video stream needs to be processed, modified, or redirected before being consumed by the end application. Managing and processing the video streams from multiple cameras, especially in systems where

multiple applications need simultaneous access, can be challenging. This challenge is where v4l2loopback and the streaming module becomes invaluable.

As the video output needs to be made available to multiple applications, simply accessing the camera's video stream directly would limit the stream's availability to a single application at a time, given the typical constraints of video device handling in Linux environments. By using the streaming module and virtual devices, it is possible to redirect the video stream from the RealSense camera to these virtual devices. Other applications can then access the video stream from these virtual devices as if they were accessing a regular video device. This setup not only allows multiple applications to use the camera's output simultaneously, but also offers flexibility in processing the video stream.

For instance, one application could use the stream for real-time video analysis, while another could record the stream for later review or processing. This flexibility is particularly useful in complex systems where different components or modules of the system have varied requirements regarding video processing and consumption. Furthermore, the streaming module illustrates the use of the v4l2loopback module in conjunction with ffmpeg, a powerful multimedia framework used for handling and processing video and audio data. In this context, ffmpeg can be used to further manipulate the video stream, such as by encoding, decoding, or transforming the stream before it is made available to other applications through the virtual video devices created by v4l2loopback.

3.3.3 Inter-Process Communication

In a framework consisting of various modules running as distinct processes within the Linux operating system, implementing effective IPC is crucial to ensure seamless operation and data exchange among these modules. Two primary IPC methods are employed in this framework: shared memory and sockets, each serving specific purposes based on their inherent characteristics and use-case scenarios. The choice between shared memory and sockets depends on several factors, including the nature of the data being shared, the required speed of communication, the complexity of the setup and the need for network transparency. Shared memory is typically faster, but more complex, to implement and manage, especially with regards to synchronization and ensuring data consistency. Sockets, while potentially slower due to the overhead of the network stack, are more flexible and easier to scale across different environments and

configurations.

3.3.3.1 Shared Memory

Shared memory is a method of IPC that allows multiple processes to access a common memory space. It's highly efficient for data transfer between processes since it avoids the overhead of data being copied between the client and server, unlike other IPC methods. In the context of this framework, shared memory is utilized for high-speed, low-latency communication where large amounts of data need to be transferred quickly and frequently between processes. Technically, implementing shared memory in Linux typically involves the use of system calls like shmget() to allocate a shared memory segment and shmat() to attach the segment to the process's address space. For synchronization of access to the shared memory, mechanisms like semaphores or mutexes are employed to avoid race conditions and ensure data integrity. Shared memory is particularly effective for applications that require rapid, real-time data processing and exchange, such as in video processing or complex computational tasks where multiple processes are handling different aspects of the computation concurrently.

3.3.3.2 Sockets

Sockets, on the other hand, are more versatile in terms of communication setup. They allow for both local (interprocess) and network communication. In this framework, sockets are used for their flexibility and ease of use, especially when the communication needs to extend over a network or between processes that do not share a common parent. In Linux, socket programming involves the use of the Berkeley sockets API. This API provides a set of function calls for creating sockets, binding them to addresses, listening for connections, accepting connections and reading/writing data. There are different types of sockets, like stream (TCP) and datagram (UDP), chosen based on whether reliable or fast, connectionless communication is required. For IPC within the same machine, UNIX domain sockets are typically used, as they are faster than IP sockets used for network communication. These sockets use the local file system as their address space, offering a higher performance for IPC. In a multi-module framework like the one described, sockets offer the advantage of scalability and

network transparency. They can handle communication between a large number of processes distributed across different systems or within the same system, making them ideal for distributed applications.

3.4 Chapter Summary

In this chapter, we dive into the key result of our research methodology: our accelerated multimodal AI framework. The initial research focused on accelerating fundamental image processing algorithms on FPGAs. Implementations of color transformations, edge detection, and noise reduction techniques on both Altera and Xilinx platforms showcased significant performance gains compared to traditional software implementations. Notably, the chapter highlights the transition from VHDL to High-Level Synthesis (HLS) using PYNQ-Z1, emphasizing the efficiency and productivity gains achieved.

Beyond preprocessing, the chapter delves into sophisticated AI models optimized for edge devices. Multimodal stream classification using LFR and HFR pathways, coupled with advanced data fusion strategies, enables accurate real-time event detection. Further, the framework incorporates pose classification, spatiotemporal autoencoders, hybrid LSTM classifiers, and audio classification techniques, all optimized for resource-constrained environments. Each model's architecture, training process, and performance benefits are thoroughly discussed.

The framework's robust architecture, built upon the NVIDIA Jetson AGX Xavier platform, ensures real-time performance and adaptability. The integration of a Hardware Abstraction Layer (HAL) allows seamless interaction with various sensors, including cameras, microphones, Lidar, and ultrasonic sensors. Moreover, the use of v4l2loopback for virtual devices and efficient Inter-Process Communication (IPC) mechanisms like shared memory and sockets guarantees smooth data flow and resource utilization.

In conclusion, this chapter presents a holistic solution for accelerated multimodal AI processing on edge devices that stems from our research methodology. The framework's ability to handle diverse data, its optimized AI models, and its robust architecture renders it as a significant contribution towards enabling intelligent real-time applications in various domains.

4

Application of the Edge Computing Framework in Public Transportation

While the theoretical and methodological foundations for our edge computing framework have been established, this chapter marks a significant step towards realizing its practical potential. In this chapter, we focus on deploying a real-world solution based on our multimodal framework within the context of public transportation and more specifically Autonomous Vehicles (AVs).

Our deployment focus on AVs stems from their demanding operational environment and energy constraints, which align perfectly with our edge computing methodology. AVs require real-time decision-making and responsiveness, which our decentralized architecture and low-latency processing can deliver. Additionally, edge computing's energy efficiency complements the battery-powered nature of many AVs, reducing communication needs and extending their range. This approach also enhances privacy, ensuring that no sensitive data are transmitted outside the vehicle.

This deployment will serve as a crucial test-bed for evaluating the framework's effectiveness and performance in a real-world setting, providing valuable insights into its capabilities and limitations. By translating our theoretical advancements into a practical application, we aim to demonstrate the transformative impact of our edge computing framework on passenger safety, system

efficiency, and overall transportation experience.

In this chapter, a detailed description of the proposed multimodal AI framework is presented in the field of automotive and public transportation. The following sections offer the problem definition regarding the arisen concern in passenger safety inside Autonomous Vehicles (AVs), as well as the proposed AI framework developed specifically for augmenting passenger safety and trust inside fully autonomous public transportation vehicles.

4.1 Problem Definition

AVs are transforming urban transportation, presenting a future where the dynamics of travel and mobility are radically redefined [108]. As these vehicles start navigating streets across the globe, they also bring various potential benefits, which can range from reducing the risk of accidents commonly associated with human driving to offering cost efficiencies and improved accessibility to transportation services [109]. By reducing travel costs and seamlessly integrating with existing mass transit systems, AVs promise to enhance the efficiency to public transportation, thereby reshaping urban mobility.

However, alongside these promising advancements, the rapid deployment of AVs surfaces a spectrum of contemporary concerns, primarily focusing on the broader implications of such technologies on emerging transportation systems, societal norms, as well as the daily lives of individuals. Notably, a central concern is the level of automation in AVs – a debate between fully automated systems and those retaining some degree of human control. The base of this debate lies in ensuring passenger safety, which is a rather multidimensional challenge encompassing physical protection and psychological comfort within these autonomous systems.

The traditional driving experience, characterized by a human driver, provides a sense of control and reassurance. The driver not only navigates the vehicle, but also manages unexpected situations, ensuring passenger comfort and offering a sense of security. In contrast, AVs eliminate this element of human supervision, leading to unknown implications for safety and security within vehicles. This raises critical questions regarding whether these automations can ensure a comparable level of safety and security in the absence of a human driver or effectively substitute for the safety supervision traditionally provided by human drivers.

In AVs, the absence of a human driver introduces various psychological and physical safety concerns. Psychologically, passengers may experience anxiety and discomfort when an automated system is in control. This discomfort is increased in scenarios requiring complex, real-time decision-making, such as navigating heavy traffic or responding to sudden road hazards. The uncertainty about an AV's ability to handle such dynamic and unpredictable situations can increase passenger discomfort, especially in the absence of a human presence to offer reassurance or intervene if necessary.

Physically, ensuring safety in AVs extends beyond preventing external accidents. It involves securing an internal environment of the vehicle, particularly in response to internal threats or emergencies. Traditional vehicles benefit from a driver who can address situations including passenger health emergencies or altercations. In contrast, AVs must rely on onboard systems to detect and manage a wide range of safety-related incidents.

Addressing these safety challenges in AVs requires a comprehensive approach, encompassing the development of advanced safety systems, ensuring reliable decision-making capabilities, building passenger trust and addressing legal and ethical considerations. Firstly, AVs must be equipped with sophisticated safety and monitoring systems, capable of not just navigating and operating the vehicle, but also attentively monitoring its internal environment. This involves deploying an array of sensors and cameras to detect potential internal threats or emergencies, thereby ensuring a secure and safe environment for passengers.

Secondly, the decision-making capabilities of AVs must be highly sophisticated and reliable. These systems should be equipped with advanced AI algorithms capable of processing extensive data to make safe, efficient and ethical decisions in real time. The challenge lies in ensuring these systems can navigate unpredictable and dynamic road conditions and make decisions that align with established human safety standards.

Moreover, building passenger trust in AVs is crucial for their acceptance and widespread adoption. This can be achieved through transparent communication regarding the vehicle's capabilities and safety features. Informing passengers about the rationale behind the vehicle's actions in real time can significantly reduce anxiety and enhance trust in the automated system.

Finally, the deployment of AVs raises substantial legal and ethical considerations, particularly concerning liability in accidents or system failures. Establishing clear legal frameworks and ethical guidelines is imperative for addressing

these concerns, ensuring accountability and providing adequate protection for passengers. However, ensuring the safety of AVs is an ongoing endeavor that necessitates continuous testing, evaluation and improvement.

As a result, while AVs present a promising future for urban mobility, ensuring the safety of passengers within these vehicles is a complex and multifaceted challenge. It requires a holistic approach involving the development of advanced safety systems, reliable decision-making algorithms, transparent communication, and the establishment of robust legal and ethical frameworks. As technology continues to advance, these challenges must be met with continuous innovation and improvement, ensuring that the safety and well-being of passengers remain at the forefront of the autonomous driving experience.

For this purpose, the aforementioned accelerated multimodal AI framework was developed as a way of providing a sense of security and trust to the passengers inside an autonomous shuttle by implementing novel AI algorithms. The proposed framework can be applied in various use case scenarios, including passenger counting, real-time abnormal event detection (passenger fighting, vandalism, bag-snatching, falling down), passenger facial recognition with masks, as well as detecting deviations from passenger proximity regulations established during the COVID-19 pandemic. In the following subsections, an in-depth description of the developed framework in the aforementioned public transportation use case scenarios is presented.

4.2 Functional Requirements

The functional requirements and system specifications for the proposed multimodal AI framework aimed at enhancing passenger safety and security in public transportation are various. As mentioned in the previous section, this framework is designed to boost the confidence of all passengers, including those who are not fully autonomous, such as children, elderly individuals, and people with disabilities. By doing so, it enables every passenger to utilize public transportation independently, without the need for guidance or supervision from a third party. This aspect of the framework is crucial as it not only advances inclusivity, but also promotes the autonomy of these individuals, ensuring that public transportation is accessible and safe for everyone.

At the core of the proposed framework is an end-to-end service built upon advanced DL models. These models facilitate a range of functionalities, in-

cluding passenger facial recognition and proximity assessment. Moreover, they are capable of detecting various abnormal events such as passenger altercations, falls, acts of vandalism and incidents of bag-snatching. The integration of these features into the framework is pivotal for ensuring a safe and secure environment for passengers. Notably, the framework incorporates novel attention-based techniques in facial recognition algorithms, which effectively address the challenges posed by occlusions, particularly those caused by medical face masks. This adaptation is especially relevant in the context of ongoing health concerns and the increasing prevalence of face masks in public settings.

Another critical aspect of the proposed framework is its deployment on embedded devices within AVs. Given the nature of these devices, power consumption is a significant consideration. To address this challenge, the proposed framework includes smart techniques, including adaptive inference, which optimize power usage without compromising on performance. This approach not only ensures the efficiency and sustainability of the system, but also maintains its effectiveness in real-time environments.

In terms of data acquisition and quality enhancement, the framework introduces specialized overhead cameras designed specifically for the AV environment, in addition to the conventional side cameras. The incorporation of these overhead cameras is a strategic decision aimed at capturing a more comprehensive view of the interior of the shuttle, thereby enhancing the system's ability to monitor and analyze activities within the vehicle. This improvement in data quality is essential for accurate event detection and contributes significantly to the overall effectiveness of the system.

A distinctive feature of the proposed framework is its multimodal nature, combining both video and audio analysis. This integration facilitates a more robust and effective crime detection mechanism, significantly enhancing passenger safety. By leveraging the complementary strengths of video and audio data, the system can identify a wider range of abnormal events with greater accuracy. For instance, audio analysis can detect sounds indicative of distress or confrontation, which may not be visually apparent. Similarly, video analysis can identify visual cues of potential security threats. This multimodal approach ensures a more comprehensive surveillance and monitoring system, offering a higher level of security to passengers inside the shuttle.

In summary, the proposed approach based on our multimodal AI framework is a comprehensive system designed to enhance passenger safety and security in public transportation. Its focus on inclusivity and independence for all pas-

sengers, combined with advanced DL models for facial recognition and event detection, makes it a cutting-edge solution. The consideration of power consumption for deployment on embedded devices, the introduction of specialized cameras for improved data quality and the fusion of multiple modalities of video and audio analysis for enhanced crime detection, all contribute to its efficacy and relevance in the current transportation landscape. This framework represents a significant step forward in the use of AI for public safety and security, particularly in the context of the evolving capabilities and adoption of AVs.

4.3 In-Cabin Monitoring Services

Building upon the strong foundation of our multimodal AI framework, this section focuses into the architecture of an in-cabin monitoring system specifically designed for AVs. This software framework aims to further enhance passenger safety and operational efficiency through a network of interconnected modules. By integrating diverse functionalities such as audio and video processing, passenger counting, facial recognition, and environmental monitoring, this system offers a comprehensive solution for real-time in-cabin management. In the following discussion, we will explore the individual components of this framework, their interactions, and their collective contribution to creating a safer and more efficient autonomous transportation experience.

This work was performed at the Centre for Research and Technology Hellas (CERTH) as part of the Horizon 2020 AVENUE project, which provided essential support and resources for the entirety of this research. Moreover, this research involved human subjects and the approval of all ethical and experimental procedures and protocols was granted by the CERTH Ethical Committee.

The provided architecture diagram represents a comprehensive software framework specifically designed to enhance the operational effectiveness and safety of AVs. The framework is structured around several core components, each serving distinct, but integrated functions, crucial for the advanced operation of AV systems:

- The launcher acts as the central hub for initiating and managing the system's various modules. It is responsible for bootstrapping and lifecycle management, ensuring all components are synchronized and can be

restarted or shut down smoothly when required.

- The modules section includes a series of specialized sub-modules, each tailored to perform specific tasks. Each submodule implements the approaches described in Chapter 3 and the results will be presented and evaluated in-detail in the next sections. For instance, the av_audio and av_video handle audio and video modalities for environment and passenger interaction monitoring. The av_counting, av_facemask, av_facenet modules are crucial for passenger detection, facial recognition, and health regulation compliance, such as mask detection. The av_depth and av_object are integral for occupation monitoring and luggage detection. The av_environmental, and av_ultrasonic modules contribute to environmental data handling and ultrasonic sensing necessary for close-range detection tasks.

- In the common component, modules like httpstream, interfaces, trt_engine play crucial roles in network communication and streaming through the Hypertext Transfer Protocol (HTTP) protocol, data allocation from sensors and running optimized AI inference using TensorRT.

- The shared resources, including av_dashboard Figure (4.23), av_ipc, av_mqtt, are essential for monitoring, internal communications and data sharing across modules, leveraging MQTT for effective messaging and IPC for inter-process communication.

- The Utils segment, particularly through avcap.py, manages multimedia data capturing, which is vital for monitoring the vehicle's environment and engaging with passengers.

- The assets, managed by the script assets.sh, handle static files or resources like configuration files or multimedia assets that are essential across various modules.

- Finally, the config module (through config.py) serves as the central configuration manager, defining settings for other modules to facilitate central management of system behavior adjustments without altering individual module codes.

The architecture in Figure 4.2 illustrates a robust AV system, stemmed from our methodology, for real-time processing of multimedia data and constant environmental monitoring, emphasizing modularity and facilitating maintenance,

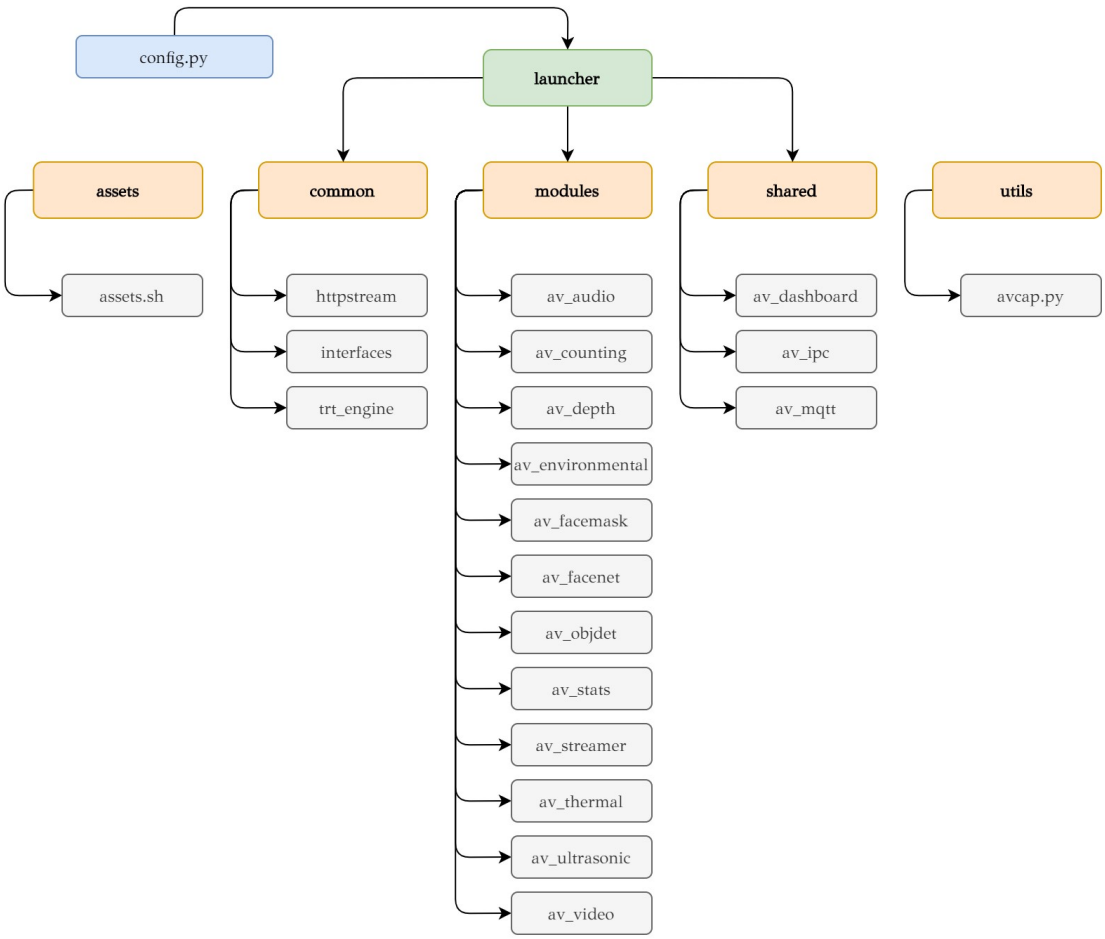

Figure 4.1: The diagram illustrates a structured software architecture for the AV system. It highlights a central "launcher" that manages a hierarchy of modules, each referring to specific functionalities like audio/video processing (av_audio, av_video), object detection (av_objdet), environmental monitoring (av_environmental, av_thermal), and passenger interactions (av_counting, av_facemask, av_facenet). Shared resources like a dashboard (av_dashboard) and communication protocols (av_ipc, av_mqtt) enable inter-module coordination. Supporting components like utilities (avcap.py) and assets (assets.sh) aid in data capture and resource management. The config.py file centralizes system-wide configurations for streamlined management.

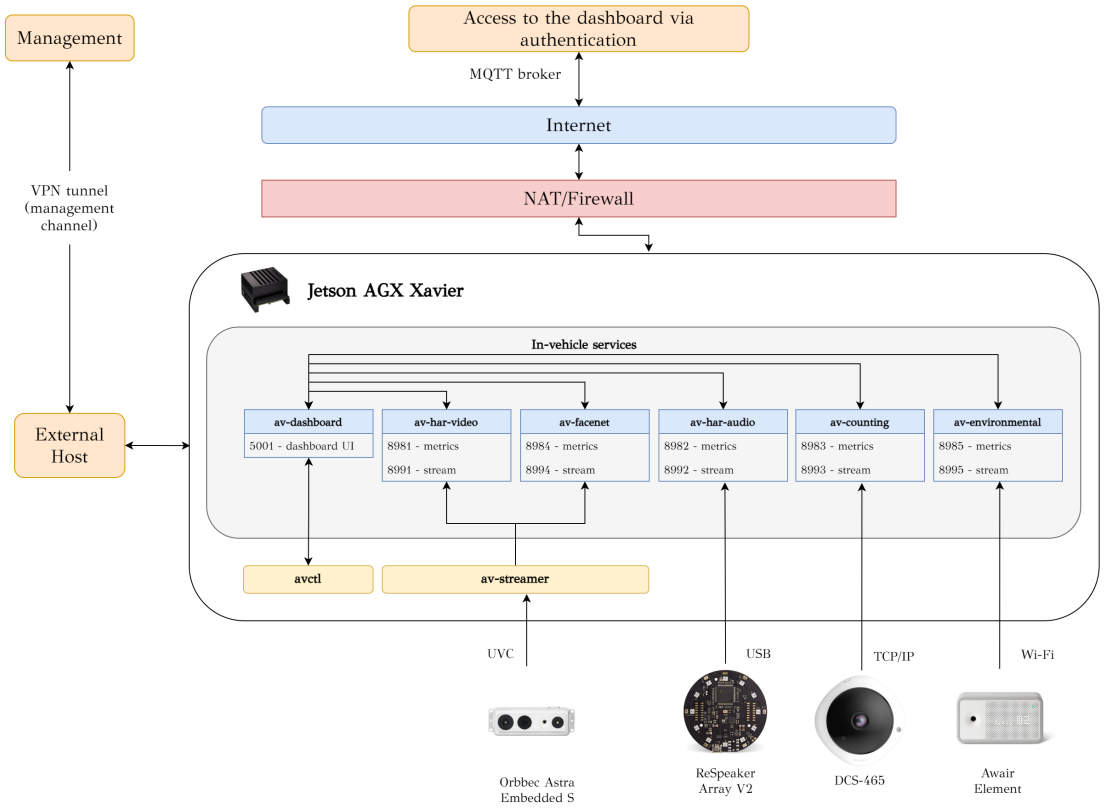

Figure 4.2: The Jetson AGX Xavier device serves as the central component, collecting data from various sensors (Orbbec Astra, ReSpeaker Array V2, DCS-465, Awair Element) through UVC, USB, TCP/IP, and Wi-Fi connections. The collected data is then processed by different services (av-facenet, av-har-audio, av-counting, av-environmental) and streamed to an external host for visualization and further analysis. A VPN tunnel ensures secure communication between the Jetson AGX Xavier and the external host.

updates, and continuous improvement, adapting to new challenges and regulations efficiently. The framework integrates cutting-edge AI technologies to address operational and safety challenges effectively, enhancing passenger trust and ensuring compliance with evolving regulatory requirements. This approach showcases a commitment to developing a reliable autonomous transportation solution that prioritizes safety, efficiency, and adaptability in dynamic urban environments.

4.3.1 Abnormal Event Detection

Having explored the foundational in-cabin monitoring system architecture, we now shift our focus to a critical aspect of autonomous vehicle safety: abnormal event detection. AVs hold the potential to transform transportation, offering increased safety, improved efficiency, and greater accessibility [110]. The autonomous shuttles are operating without a human driver, promising convenient on-demand transportation solutions. However, the absence of an onboard authority figure dedicated to passenger well-being poses unique safety and security risks [109]. The development of systems capable of autonomously monitoring the shuttle's interior and detecting potential threats in real time is crucial in protecting passengers and enabling timely interventions when needed. Overhead fisheye cameras offer a compelling solution for comprehensive interior surveillance in autonomous shuttles [111]. Their wide field of view inherently minimizes occlusions frequently encountered in confined spaces and under dynamic conditions, offering enhanced visibility of the passenger area.

The integration of advanced computer vision and ML techniques facilitates the automated detection of anomalous events based on video data, contributing to onboard safety. Autoencoders [112], are employed for anomaly detection by learning compressed representations of normal passenger behavior from video data. Deviations from this learned normality result in higher reconstruction errors, indicating potential anomalies.

While autoencoders excel at detecting unseen anomalies [113], the lack of readily available labeled datasets for anomalous events in AVs presents a challenge. Semi-supervised learning paradigms address this limitation by effectively capitalizing on abundant unlabeled data alongside a smaller set of labeled examples, leading to improved model generalization and robustness. This approach is particularly suitable in autonomous shuttles as collecting large amounts of

normal operational data is comparatively simple, while incidents requiring intervention occur less frequently.

4.3.1.1 Datasets

Having outlined the theoretical foundations in abnormal event detection, we now turn our attention to the practical aspect of data acquisition and preparation. The foundation of any successful machine learning or AI project lies in the availability and quality of datasets. These collections of data serve as the training ground for algorithms, allowing them to learn patterns, recognize features, and ultimately make accurate predictions or decisions. In our pursuit of robust and reliable abnormal event detection systems for public transportation, the collection, curation, and preparation of diverse datasets are crucial. These datasets, spanning controlled simulations and real-world scenarios, provide the critical raw material upon which our AI models are built, trained, and evaluated.

At the first data capture at CERTH facilities, 13 different scenarios were simulated using two different camera perspectives for each one. The dataset contains 6650 frames that, in conjunction with some augmentation techniques described later, were sufficient to train the LSTM Classification via Pose Estimation experiment and obtain decent results. During the second data capture performed in TPG depot, 29 video sequences were captured with 46127 frames, demonstrating real conditions in an autonomous vehicle. The merging of these two datasets led to substantially better results and allowed to perform more experiments. Samples from NTU-RGB [114] were dataset further included, which helped to access the performance on unknown data and fine-tune the model to generalize. In addition, samples from P-REACT [115] dataset were used to verify the results and implement the spatiotemporal autoencoder.

Apart from the data captured in Figure 4.3, there are also various other well-established datasets for anomalous activity recognition within video surveillance footage, with the most prominent among them including the UCSD Pedestrian [116], Subway [117] and CUHK Avenue [118] datasets. However, despite their extensive application, these datasets show various limitations, including the simplicity of the depicted scenes, their restricted range of anomalous activities and the lack of detailed spatial annotations, that may lead to unsatisfactory outcomes in real-world scenarios.

On the other hand, datasets such as the UCF-Crime [119] and the Shang-

haiTech [120] offer a comprehensive collection of videos sourced from various online platforms, recorded across multiple surveillance systems under a wide array of environmental conditions, thereby introducing additional layers of complexity. More specifically, the UCF-Crime dataset includes approximately 1900 extensive, unedited videos, evenly split between normal and abnormal events, and covers 13 types of real-world anomalies, including, but not limited to, abuse, burglary and vandalism. Additionally, the ShanghaiTech dataset aims to address real-world applicability issues by including anomalies characterized by sudden movements, like chases and fights, including 130 anomalous events across 13 settings in 437 videos, totaling over 270,000 frames for training. It specifically labels unusual activities such as bag snatching and unauthorized vehicle use.

However, while the UCF-Crime dataset initially contributed video-level annotation, being especially useful for weakly supervised learning approaches, the ShanghaiTech dataset has been utilized for unsupervised learning to determine regular patterns. The availability of frame-level annotations for both datasets facilitates the adoption of fully-supervised learning methods. The UCF-Crime dataset, with its varied activity rate and environmental settings, is particularly practical for anomaly detection tasks, enhancing the development of surveillance systems operating in real-world environments.

Besides the two benchmark datasets mentioned previously, we collected a real-world dataset (CERTH-AV) using the D-Link DCS-4625 fisheye camera with an overhead panoramic perspective. The camera has a dome design, featuring a 5-megapixel 1/2.5" CMOS sensor, paired with 1.37 mm F2.0 fisheye lens. It has a maximum image resolution of 2560×1920 pixels and has Wide Dynamic Range (WDR) support along with IR lighting for night vision. During the data collection, several abnormal events were simulated (bagsnatch, falldown, fighting, vandalism) with a duration of approximately 30 minutes. Moreover, we collected regular vehicle operation, stationary or in motion, with various lighting conditions and passengers. Table 4.1 presents some metrics about this dataset. It is important to note that in the initial training phase of the CAE only the regular "normal" class is used, while a subset of the class is used for fine-tuning the hybrid approach. Consents were obtained from all passengers contributed to this dataset for the purpose of this research.

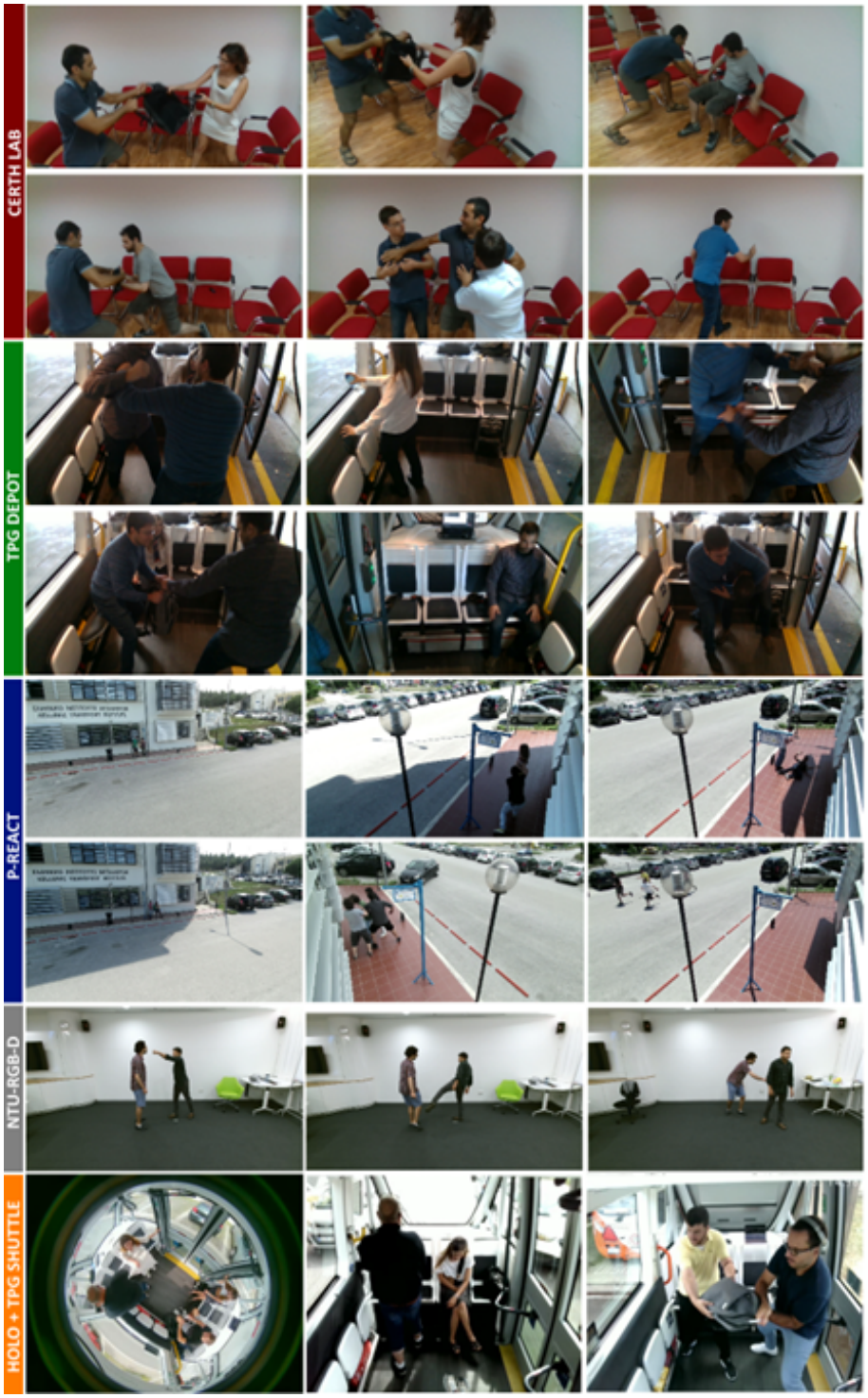

Figure 4.3: Dataset samples with abnormal events, which highlight the use cases. Fighting, aggression, bag-snatching, and vandalism scenarios are illustrated. The red section contains simulated data in lab, green section depicts captured data from TPG shuttles, blue section shows scenarios from P-REACT dataset, and the gray section indicates additional data imported from the NTU-RGB dataset.

Table 4.1: Statistics of the overhead fisheye camera dataset (CERTH-AV).

Class	Size (GB)	Samples	Duration	FPS
Bagsnatch	4.5	125	0h 30m 40s	15
Falldown	3.9	109	0h 26m 55s	15
Fighting	5.1	142	0h 35m 35s	15
Normal	39.5	1098	4h 15m 20s	5-15
Vandalism	6.2	172	0h 40m 35s	15

4.3.1.2 Evaluation Metrics

The experimental results of this work were evaluated against established benchmarks in the field using universally recognized metrics for detecting anomalies. To keep consistency with previous studies in anomaly detection [121], [120], the findings are expressed through the frame-level Area Under the Curve (AUC) to facilitate a comparison of performance levels, with a higher AUC indicating better detection capabilities. Given the challenges posed by datasets where anomalies are rare, AUC is preferred over accuracy as it offers a more refined assessment [122]. Additionally, the model's ability to distinguish between different types of anomalies was assessed using confusion matrices (Figure 4.4) and the accuracy metric.

4.3.1.3 Training Settings

The optimization and fine-tuning of AI models rely on a careful selection of hyperparameters and training settings. These parameters, often selected through a combination of empirical experimentation and theoretical understanding, play a crucial role in highlighting a model's performance, accuracy, and generalization capabilities.

To achieve peak performance, we explored the optimization of critical hyperparameters. We experimented with parameters such as frame-skipping, data normalization, temporal-length, temporal-stride, and the potential benefits of randomized runtime data augmentation for temporal sequence generation. We also carefully examine the potential for enhancements within the deep network architecture itself. We trained our model with a batch size of 16, a temporal-length of 45 frames and an input image dimension of 224×224 with 3 channels.

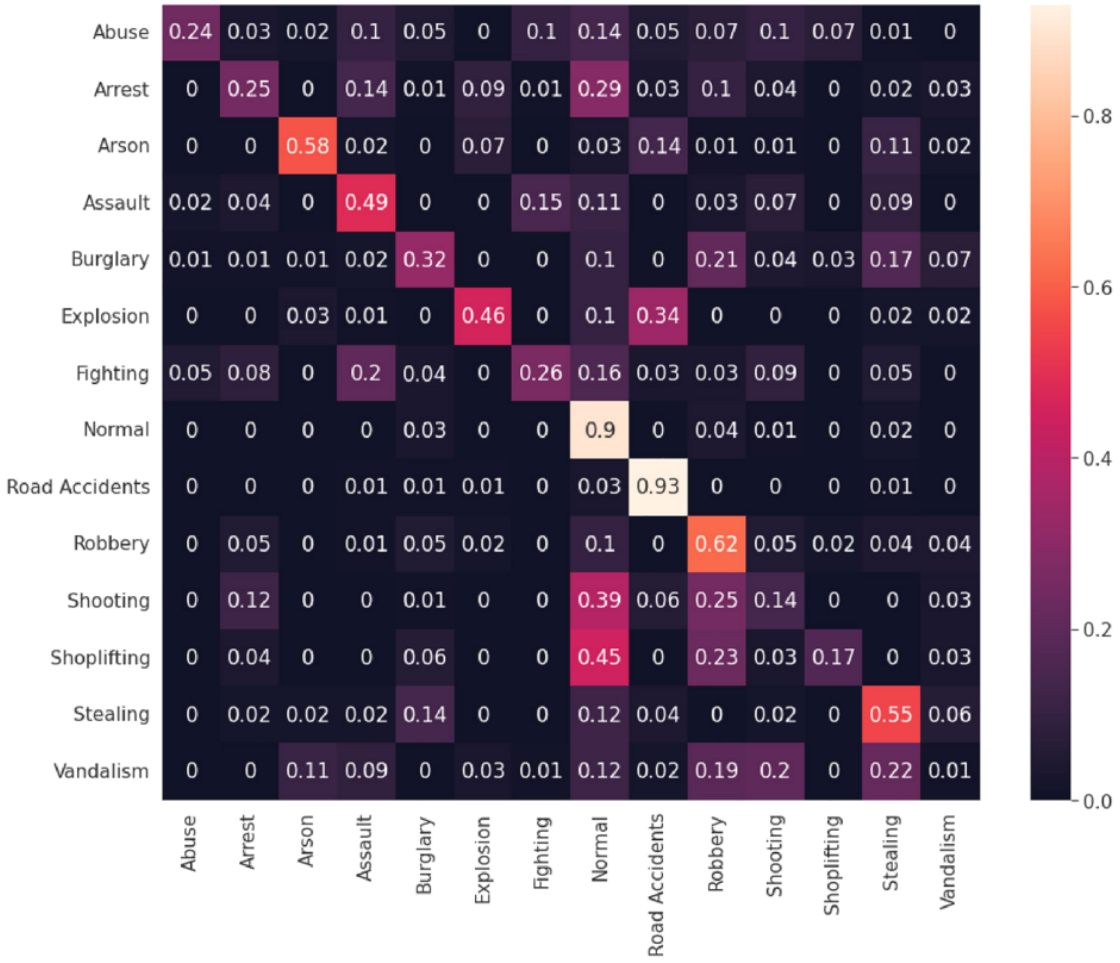

Figure 4.4: Confusion matrix across multiple categories: the diagonal represents accurate predictions and off-diagonal cells indicate false predictions. The intensity of the color corresponds to the normalized frequency of predictions, highlighting the precision and misclassification rates between different classes.

To prevent overfitting, we implement dynamic frame-skipping on input videos with a stride between 1 and 5, and random runtime augmentations are applied on image sequences with a probability of 0.3. Train, validation, and test sets are randomized to ensure a robust and unbiased evaluation.

Due to the large number of video data, the bottleneck of our training process was the data loader. Operations for decompressing videos, preprocessing, image transformations and transferring data from the CPU to GPU had a significant performance impact on the training process. In order to maximize computational efficiency and streamline the training process, we integrate NVIDIA DALI for accelerated data loading. This optimization ensures that our hardware isn't bottlenecked by data preparation. Training is conducted using the Adam optimizer with a learning rate of 0.0001. To harness the latest in computational power, this training process is powered by a system equipped with the high-end NVIDIA 4090 GTX 24GB GPU, an Intel Core i7-13700K processor, and 64 GB of RAM.

4.3.1.4 Experimental Results

In this section, results from all three modalities (RGB, depth and audio) are presented. Future research should consider the potential effects of fusing video and audio modalities. Models can be fused both on decision-level and by concatenating their respective fully connected layers. Recent studies by Kampman et al. [123] and Ortega et al. [124] have proven that using multiple modalities combined allows interaction between them in a non-trivial way and greatly outperforms the individual performance. By combining the last network layers and fine-tuning the parameters, we can take advantage of the complementary information of video and audio modalities outperforming unimodal results.

Pose Estimation

Pose estimation plays a crucial role in tracking individuals across video frames. This involves matching poses between frames, forming pose flows, and applying pose flow non-maximum suppression (NMS) to refine and link these flows. An efficient online skeleton tracking algorithm, utilizing distance-based and heuristic methods, was implemented to meet real-time performance requirements.

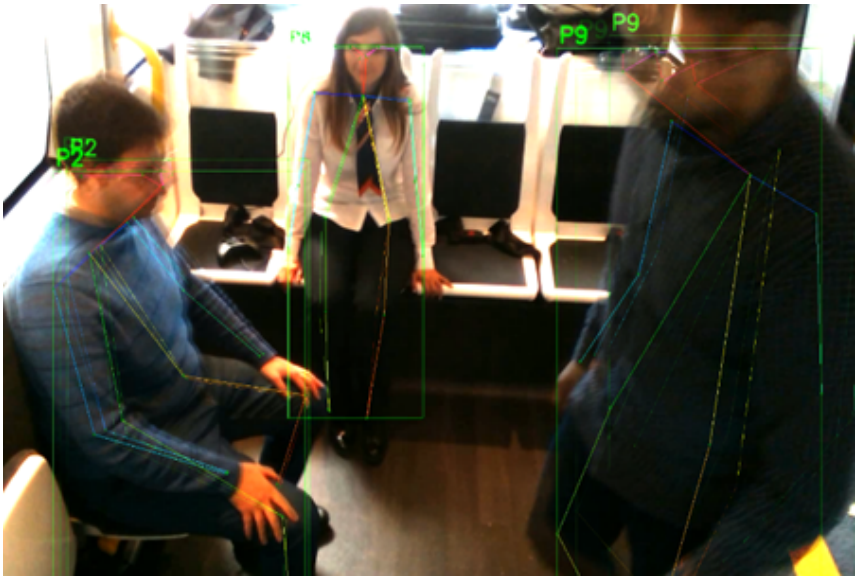

Figure 4.5: Skeleton matching across two different frames (blended). Notice that the passenger ID, highlighted in green at the left of each bounding box, is the same across the frames. This allows for the tracking of individual passengers across consecutive frames.

The first important step is to match cross-frame poses and form pose flows (tracking). Also, a novel pose flow NMS is applied to reduce redundant pose flows and re-link temporal disjoint ones. This is an important step that associates poses indicating the same person, across multiple frames. We implemented an online skeleton tracking algorithm based on distance and other heuristics, in order to meet the performance requirements of the real-time service. The algorithm is sorting the skeletons based on the distance between neck and image center, from small to large. A skeleton near the center will be processed first and be given a smaller human id. Later on, each skeleton's features will be matched between the current and the previous frame.

The distance matrix (or cost) between the skeleton joints is the main criterion for the matching function. Skeletons with the small distance are matched between the frames and are given the same id (Figure 4.5). In some cases, the skeleton detection framework might fail to detect a complete human skeleton from the image due to the restricted field of view of the camera in the autonomous vehicle, as depicted in Figure 4.6.

This may cause some blanks in the joint positions, which should be filled with some values in order to maintain a fixed-size feature vector for the following feature classification procedure. We evaluated some solutions for this issue:

- Solution 1: Discard this frame. However, the algorithm would never be able to detect the action when the person is standing sideways and not

Figure 4.6: Examples indicating the restricted field of view for the camera sensor.

facing the camera.

- Solution 2: Fill in the positions with some value outside a reasonable range. Theoretically, when the classifier is strong enough, this method could work.
- Solution 3: Fill in a joint's position based on its relative position in the previous frame with respect to the neck.

In order to solve this issue, solution 3 was implemented, but the classifier's performance was degraded in some test cases. After extensive tests, we noticed that a previous joint position might be missing too, being replaced by the estimation of our algorithm. This led to "stuck" joints across various frames and confused both the tracker and the classifier. To overcome this issue, we are using a default "idle" pose as an example for our algorithm. When a previous joint is missing, the value being replaced is relative to the default example pose (Figure 4.7). We chose a person sitting as the default, because it is the most common for the passengers in the AV.

To ensure the reliability of our feature vector for subsequent classification despite occasional missing joint data, we implemented a comprehensive strategy involving a default "idle" pose as a fallback. This approach helped mitigate the issue of "stuck" joints, but it also underscored the need for dimensionality reduction to manage the complexity of our dataset effectively. To address this, we applied a Principal Component Analysis (PCA) procedure to reduce the 314 initial features to 50 principal components. PCA is a statistical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables (entities each of which takes on various numerical values) into a set of values of linearly uncorrelated variables called principal components. This transformation is defined in such a way that the first principal

Figure 4.7: The two leg joints are being reconstructed (right) by their relative location in the previous frame (left), despite one joint being occluded by another passenger in the current frame.

Figure 4.8: Abnormal event detection showcasing two passengers involved into fighting. The red bounding boxes illustrate the abnormal event.

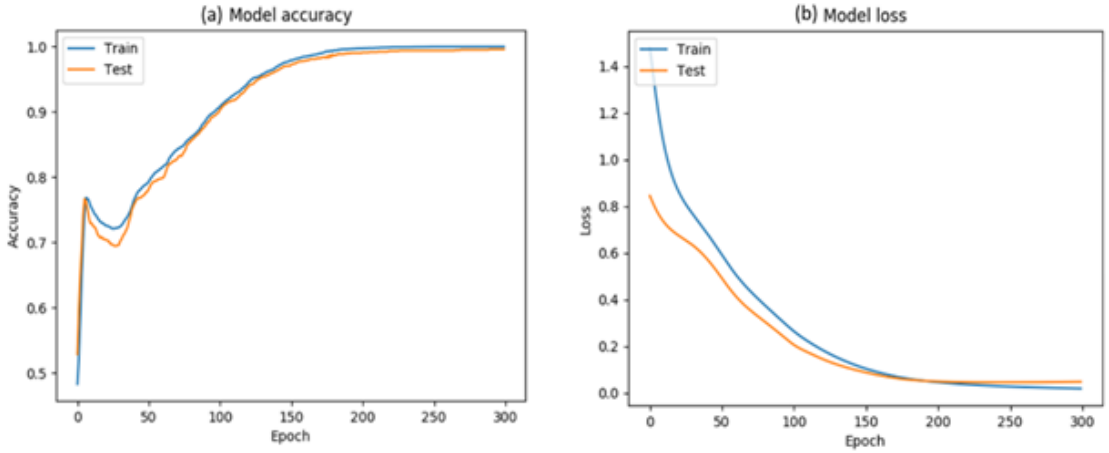

Figure 4.9: Model performance evaluation – (a) training accuracy and (b) training loss metrics.

component has the largest possible variance (that is, accounts for as much of the variability in the data as possible), and each succeeding component in turn has the highest variance possible under the constraint that it is orthogonal to the preceding components. The resulting vectors (each being a linear combination of the variables and containing n observations) are an uncorrelated orthogonal basis set. PCA is sensitive to the relative scaling of the original variables. After the PCA procedure, we achieved a sum of 0.9981% eigenvalues. The model converges faster and the accuracy improves. Due to the highly imbalanced classes, we compiled our model with a custom weighted categorical cross-entropy function for loss calculation, and a weighted categorical accuracy method in order to acquire accurate metrics.

The quantitative evaluation performance of our model is depicted below (Figure 4.9). After 50 epochs, our model achieved a 96.22% accuracy. The time cost for feature extraction and classification is less than 0.05s per frame for the classifier, since the model is relatively shallow. The model is trained end-to-end and regularized so that it distils the most compact profile of the normal patterns of training data and effectively detects abnormal events (Figure 4.17). After training the hybrid model for 20 epochs, using a 64/16/20 train/validation/test ratio the loss and the accuracy are depicted in Figure 4.16 and the final model accuracy is 98%. Finally, we evaluated our model at the NTU RGB+D dataset (Figure 4.10) and on the data captured by CERTH inside the AV's shuttle (Figure 4.11).

The qualitative results presented below highlight the robustness of our model in detecting abnormal events across various scenarios. In the images there are various debug layers enabled, such as skeleton points, lines, tracker id and

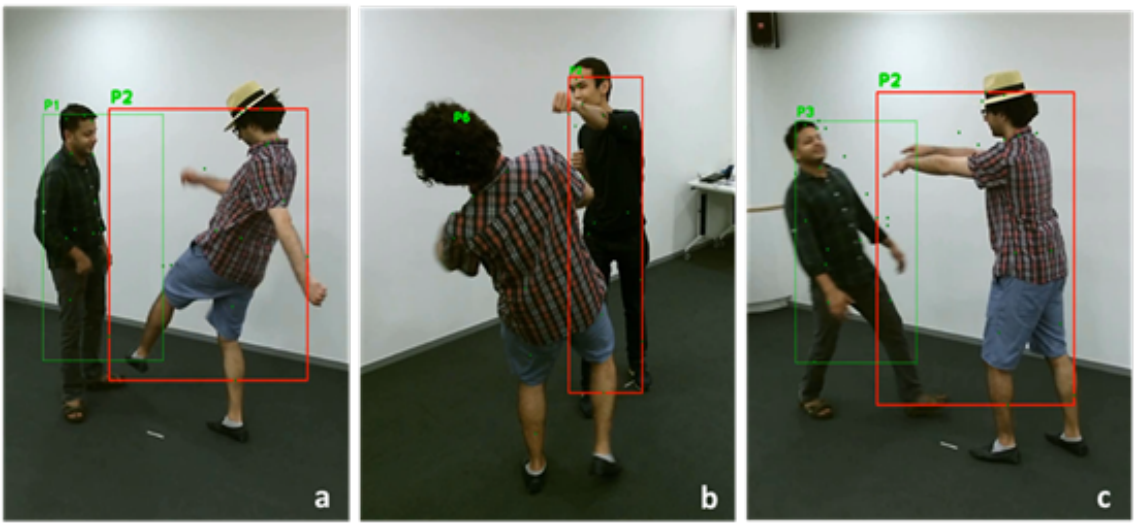

Figure 4.10: Evaluation on test data: (a), (b), (c) Abnormal event detection (violence/passengers are fighting) using different camera angles from the NTU-RGB dataset. Dots represent the detected keypoints.

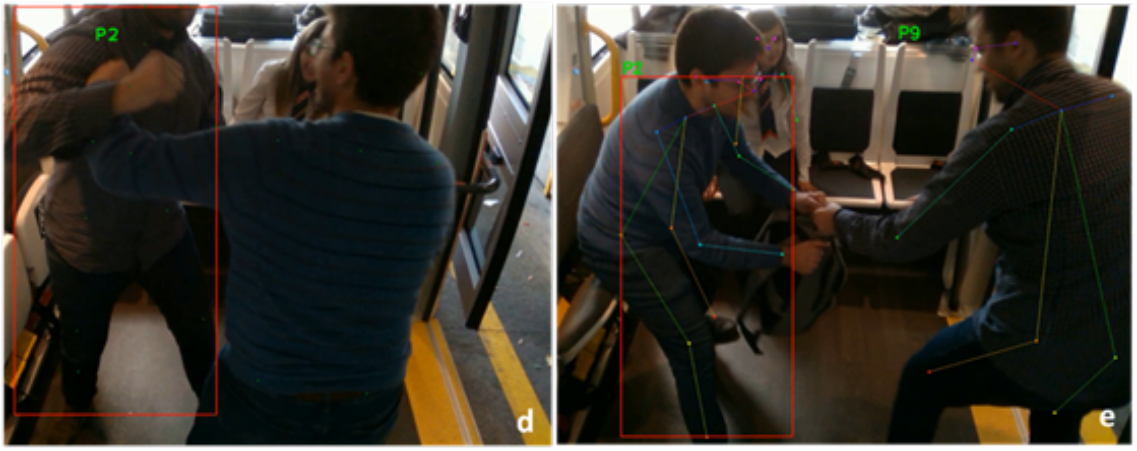

Figure 4.11: Evaluation on test data: (d), (e) Detection of fighting/bagsnatch real-world scenarios inside the shuttle.

Table 4.2: Precision, Recall and F1-score metrics for the two classes.

	Precision	Recall	F1-Score	Support
Normal	0.99	0.99	0.99	1208
Abnormal	0.93	0.95	0.94	147
Accuracy	0.99	0.99	0.99	1355
Macro Avg	0.96	0.97	0.96	1355
Weighted Avg	0.99	0.99	0.99	1355

Figure 4.12: UC01 (fighting event) – a petty crime takes place in the form of assault. Green boxes represent normal events, while red boxes (as in the picture) showcase abnormal behavior.

bounding boxes of each detection. The predicted result is marked as green, when the classifier indicates it as "normal" and red when "abnormal", correspondingly. So far, we did not include NTU dataset samples in our training set, so it is safe to assume that our model can generalize across different people and view angles. In the following Figures 4.12-4.15, we present some results on each use case. Finally, the respective classification results and metrics for each class are presented in the following Table 4.2.

The effectiveness of our model in real-world applications is demonstrated through several use case scenarios, each highlighting the system's ability to detect and respond to abnormal events with high accuracy. During a shuttle ride, if a petty crime such as assault (Figure 4.12) or attempted bag snatching (Figure 4.13) occurs, the system identifies the event and sends a security alert to the operator or security supervisor. The operator then decides on the appropriate course of action, which could involve stopping the shuttle or notifying passengers via the radio system. In another scenario, if a young person attempts vandalism on the shuttle, such as painting graffiti or smashing windows (Figure 4.14),

Figure 4.13: UC02 (bagsnatching event) – the commuter is attacked by another person who is attempting to snatch his bag. Green boxes represent normal events, while red boxes (as in the picture) showcase abnormal behavior.

Figure 4.14: UC03 (vandalism event)

UC03 (vandalism event) – a person attempts to perform a vandalism action in the bus, through painting a graffiti on the windows. Green boxes represent normal events, while red boxes (as in the picture) showcase abnormal behavior.

Figure 4.15: UC04 (unaccompanied luggage monitoring) – unaccompanied luggage which remains unmoved for a long time. Green boxes represent normal events, while red boxes (as in the picture) showcase abnormal behavior.

especially during night hours, the event is detected. The system responds by either warning the person through the shuttle's radio system or alerting security personnel, who may then intervene by stopping the shuttle.

Hybrid Spatiotemporal Autoencoder

Transitioning from individual pose estimation to understanding broader scene dynamics, we now present the results of the hybrid spatiotemporal autoencoder model. This model analyzes and predicts patterns of movement and interaction within the video frames, providing a deeper understanding of the unfolding events.

The training graphs in Figure 4.16 illustrate the training/validation loss and accuracy of the hybrid classifier over 20 epochs. The left graph shows the model loss, where both the training and validation loss steadily decrease as the number of epochs increases. This indicates that the model is learning effectively, with the loss values converging towards zero. The right graph depicts the model accuracy, showing an upward trend for both training and validation accuracy over the epochs. Initially, there is a rapid increase in accuracy, which then plateaus as it approaches 1.0, showing high accuracy levels. The close alignment of training and validation curves in both graphs suggests that the model generalizes well to unseen data, without significant overfitting. The con-

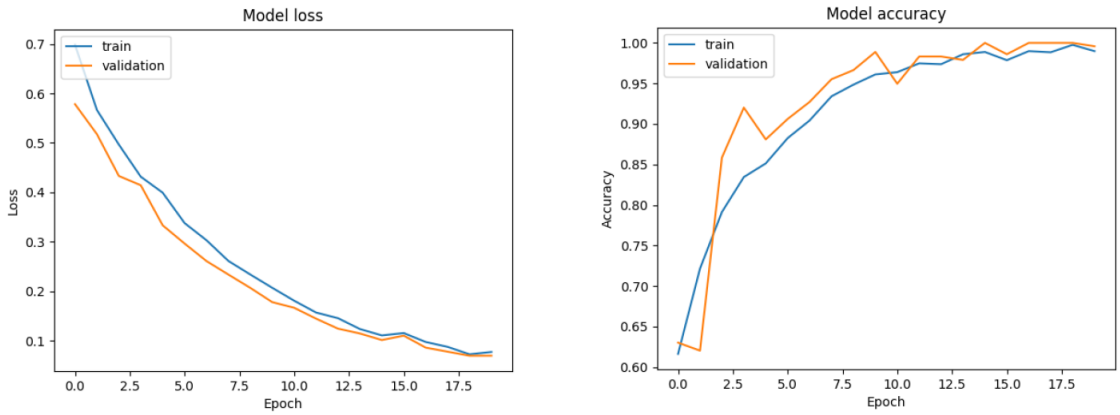

Figure 4.16: Train/Val loss and accuracy of the hybrid classifier, over 20 epochs.

Table 4.3: Classification results and metrics for each class.

Class	Precision	Recall	F1-Score	Support
Normal	0.99	0.99	0.99	6040
Abnormal	0.93	0.95	0.94	1335
Accuracy				0.99
Macro avg	0.96	0.97	0.96	7375
Weighted avg	0.99	0.99	0.99	7375

sistent improvement and convergence in these metrics demonstrate the model's robustness and effectiveness in classification tasks.

The quantitable classification results presented in Table 4.3 demonstrate the high effectiveness of our model in distinguishing between normal and abnormal events. The model achieved a precision of 0.99 for normal events, indicating that it is highly accurate in correctly identifying normal instances without falsely labeling abnormal ones as normal. Similarly, the recall for normal events is also 0.99, suggesting that almost all actual normal events are correctly recognized by the model. For abnormal events, the precision is slightly lower at 0.93, which means that there are a few instances where the model incorrectly labels normal events as abnormal. However, the recall for abnormal events is 0.95, indicating that the model successfully identifies most of the true abnormal events. The F1-score, which is the harmonic mean of precision and recall, is 0.99 for normal events and 0.94 for abnormal events, reflecting a balanced and robust performance across both classes. The overall accuracy of the model is 99%. The macro average values, which provide an average of the precision, recall, and F1-scores for both classes, are also high (0.96, 0.97, and 0.96 respectively), showing the model's ability to perform consistently well across both classes.

Table 4.4 presents an experimental comparison of various human action recognition methods based on a recent survey by Zhang et al. [125]. The perfor-

Table 4.4: Experimental comparison with human action recognition methods based on the recent survey by Zhang et al. [125]. The proposed method does not utilize depth information from the NTU-RGB-D dataset.

Method	Year	NTU-RGB-D	Custom dataset
[126]	2020	91.5%	
[127]	2020	90.3%	
[128]	2018	73.4%	
[129]	2018	30.7%	
[114]	2016	62.93%	
[130]	2016	69.2%	
[131]	2014	31.82%	
Proposed	2020	71.4%	99.6%

mance of each method is evaluated using the NTU-RGB-D dataset, a benchmark for human action recognition, and, for the proposed method, also on a custom dataset. The methods listed span from 2014 to 2020, showcasing advancements in the field over time.

The method proposed by Shi et al. in 2020 achieves the highest accuracy on the NTU-RGB-D dataset at 91.5%, closely followed by Yang et al. in the same year with 90.3%. Earlier methods such as those by Song et al. in 2018 and Liu et al. in 2016 show moderate performance with accuracies of 73.4% and 69.2%, respectively. Notably, some of the earlier approaches, such as Yan et al. in 2018 and Yang et al. in 2014, have significantly lower accuracies of 30.7% and 31.82%, indicating the rapid improvement in action recognition techniques over recent years.

The proposed method in this study, developed in 2020, does not utilize depth information from the NTU-RGB-D dataset and achieves an impressive accuracy of 71.4% on this dataset. Moreover, it achieves an accuracy of 99.6% on a custom dataset, demonstrating its effectiveness and potential for specific applications despite not using depth information, which is typically crucial for high performance on the NTU-RGB-D dataset.

Regarding the qualitative results, we provide the preprocessed input frame for the current moment at the bottom-left (Figure 4.18 and 4.19) for the visualization the predictions. The frame is resized from 64×64 and grayscale and a mask overlay are obscuring the out-of-interest areas (road). At the right next to the input frame, the resized output of our model is shown. The third mini-frame demonstrates only the significant differences between the input and the output frames with white pixels, which are then merged above the original

Figure 4.17: Outdoor group fighting scenario on a simulated bus stop. Green metrics at top-left indicate the current and the average regularity score. A lower regularity score indicates that the predicted reconstruction is not accurate, since our model did not learn such an event. Note that the current score is much lower than the average, triggering an abnormal notification.

frame for demonstration purposes. In Figure 4.20, we present some results on each use case. Each row contains two images of fighting, bagsnatch, vandalism and falldown events correspondingly.

Overhead Hybrid Convolutional Autoencoder

Shifting our focus from individual interactions to a more comprehensive understanding of scene anomalies, we now present the results of the overhead hybrid convolutional autoencoder (CAE-Hybrid) model. This model leverages a fisheye camera perspective, incorporating center-weighted loss and a hybrid approach to effectively identify and analyze unusual events within the monitored environment. As Table 4.5 demonstrates, our CAE-Hybrid method exhibits promising results in anomaly detection compared to existing approaches, particularly excelling on the CERTH-AV dataset while maintaining competitive performance on other benchmarks.

Table 4.5 presents the comparative evaluation of various anomaly detection methods based on the AUC performance across three distinct datasets: CERTH-AV, UCF-Crime, and ShanghaiTech. These datasets are benchmarks in the domain of anomaly detection within video surveillance, each presenting

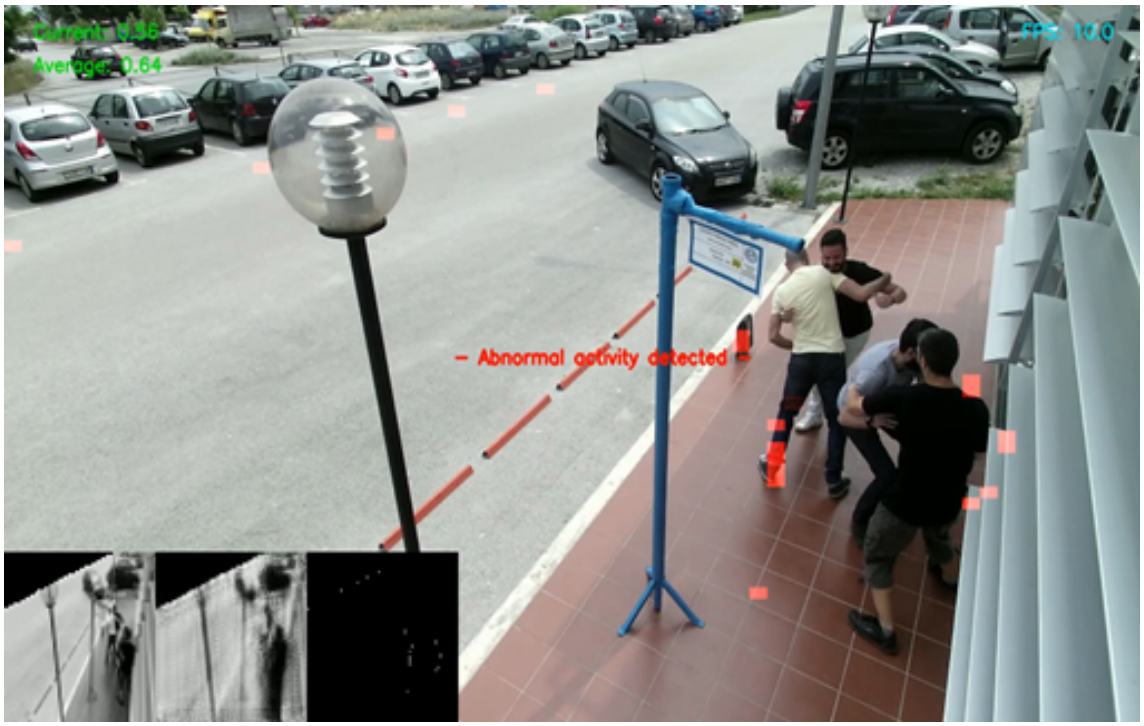

Figure 4.18: Evaluation on UC01 (fighting event) illustrating a group of four people involved into a fighting scenario. The images in bottom-left side illustrate the network's input, the reconstructed input and their absolute difference map.

Figure 4.19: Evaluation on UC01 (fighting event) illustrating a group of four people injured on the ground and running. The images in bottom-left side illustrate the network's input, the reconstructed input and their absolute difference map.

Figure 4.20: Evaluation on use cases with the SlowFast algorithm. Each row contains two images of fighting, bagsnatch, vandalism and falldown events correspondingly.

Table 4.5: AUC results evaluation of the proposed method on the datasets of CERTH-AV and benchmark datasets UCF-Crime and ShanghaiTech.

Work	**Method**	**CERTH-AV**	**UCF-crime**	**ShanghaiTech**
Sultani et al. [119]	Deep MII	0.622	0.7541	-
Liu et al. [134]	MLEP	0.651	-	0.768
Zhong et al. [135]	GCN-C3D	0.702	0.810	0.764
Zhong et al. [135]	GCN-TSNRGB	0.736	0.821	0.844
Zhong et al. [135]	GCN-TSNOptical-Flow	0.718	0.780	0.841
Gianchandani et al. [136]	Spatiotemporal	0.672	0.630	-
Hao et al. [137]	TSNRGB+Optical-Flow	0.724	0.812	0.967
Shreyas et al. [138]	AC-MIL	-	0.798	-
Zaheer et al. [139]	Binary clustering	0.721	0.782	0.841
Dubey et al. [140]	DMRMS	-	0.819	0.685
Majhi et al. [141]	Two-level attention	0.741	0.821	-
Ullah et al. [142]	CNN-BDLSTM	0.745	0.855	-
Cao et al. [132]	WAGCN	0.695	0.846	0.960
Thakare et al. [143]	TCC	0.756	0.844	-
Chen et al. [133]	MGFN	0.732	0.869	-
Proposed method	**CAE-Hybrid**	**0.878**	**0.852**	**0.927**

unique challenges and complexities. The table illustrates a notable evolution in methodological complexity and specificity, with recent methods like Cao's et al. [132] in 2022 WAGCN and Chen's et al. [133] in 2023 MGFN showcasing significant strides in performance, particularly on the UCF-Crime dataset. This reflects a trend towards more adaptive anomaly detection mechanisms, capable of handling the diverse and complex scenarios presented by these datasets. Our proposed method, CAE-Hybrid had the highest AUC of 0.878 on CERTH-AV, alongside competitive performances on UCF-Crime and ShanghaiTech. This suggests a robust and versatile approach, capable of overcoming the challenges related to the overhead fisheye camera perspective due to its incorporation of the center-weighted function loss and hybrid approach.

It is important to note that the absence of results for certain methods on our CERTH-AV dataset, is attributed to the inability to reproduce the code for the respective methods. In cases where code was not available, implementations were based on the authors' interpretations of the published methodologies, which might not fully capture the original intent of these approaches. By emphasizing the central aspects of the image more significantly than peripheral, the robust-

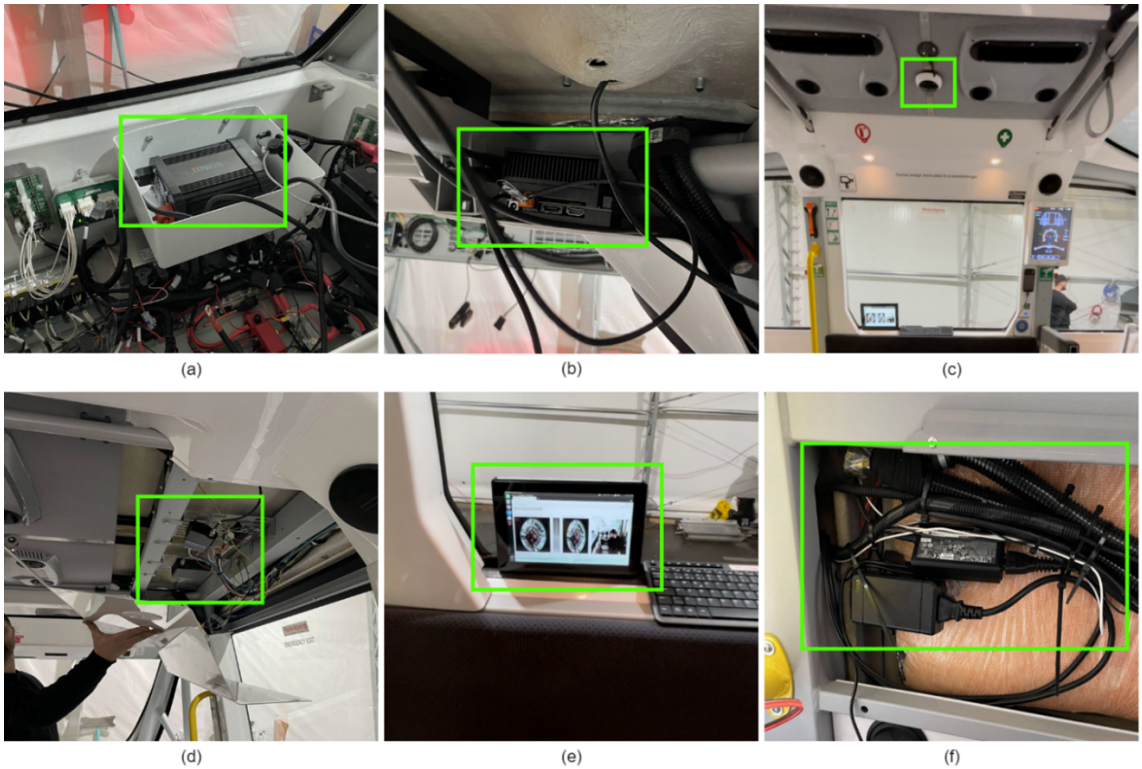

Figure 4.21: Real-world installation on NAVYA autonomous minibuses: (a) inverter, (b) installation area - panel, (c) overhead fisheye camera (D-Link DCS 4625), (d) location of the NVIDIA Jetson AGX Xavier embedded system, (e) monitoring screen for passengers, (f) power-supply cables hidden behind the vehicle's panel.

ness of this method is increased by reducing false positives and focusing on core features that are indicative of anomalous behavior, improving its sensitivity and specificity in anomaly detection task.

Regarding the system's deployment, it was evaluated in automated minibuses in Copenhagen and Geneva. The solution was installed on NAVYA AVs, featuring a NVIDIA Jetson AGX Xavier platform and a D-Link DCS-4625 fisheye camera, as presented in Figure 4.21(c) and (d). The camera was connected directly to the Jetson system via Ethernet through RTSP protocol. Both components are powered through the vehicle's batteries and Tensor-RT conversion was performed to maximize the algorithm's efficiency, reducing the power consumption to approximately 10 Watts.

The validation results by Amobility (HOLO) and Transports Publics Genevois (TPG) operators are presented in Table 4.6. As observed by the table, these results indicate the proposed semi-supervised anomaly detection algorithm is able to identify most of the perform scenarios (bagsnatch, falldown, fighting and vandalism) with an accuracy of approximately 91% on real-world conditions.

Figure 4.22 demonstrates snapshots during the real-time detection of various

Table 4.6: Number of samples validated, along with the improved accuracy and F1-Score metrics for each class.

Class	Samples	Accuracy	F1-Score
Bagsnatch	15	0.93	0.9310
Falldown	12	0.86	0.8800
Fighting	13	0.92	0.9240
Vandalism	15	0.94	0.9355

abnormal events in HOLO vehicle P109, route Slagelse in Copenhagen, Denmark. For visualization purposes, activation maps are overlaid to highlight regions of interest, denoted by red shades, where anomalous activities are detected.

Finally, as depicted in Figure 4.23, the abnormal event has also been captured in the operator's dashboard in real-time, raising appropriate alerts for the authorities and enhancing the overall safety and trust of onboard passengers.

Sound Analysis

Beyond visual analysis, our investigation now delves into the audio domain. We explore sound analysis techniques, utilizing a DenseNet-121 model to classify audio signals under varying signal-to-noise ratios (SNR). Our experiments focus on evaluating the model's performance in classifying target sounds among background noise, revealing its robustness and adaptability to different acoustic conditions.

Regarding our experiments, we used a batch size of 16 images (each image is a 3-second spectogram representation of the sound) and set the initial epochs to 200. However, we noticed that for the DenseNet-121 only eight epochs were sufficient to achieve the optimal performance. We applied an Early Stopping function [144], where we checked the validation F1 macro averaged score for an improvement in five consecutive epochs. If no improvement was detected the network stopped the training in order to avoid overfitting.

The DenseNet-121 achieved a training F1-Score of 95.92% and a validation F1-Score of 88.74% (Figure 4.24-a,c) for the case of 0 dB SNR. Regarding the case of 30 dB, the DenseNet-121 achieved a training F1-Score of 96.84% and a validation F1-Score of 91.82% (Figure 4.24-b,d). As expected, the network is able to classify the target classes with higher accuracy in the case of 30 dB SNR and we notice that the training loss starts at smaller values, compared to the

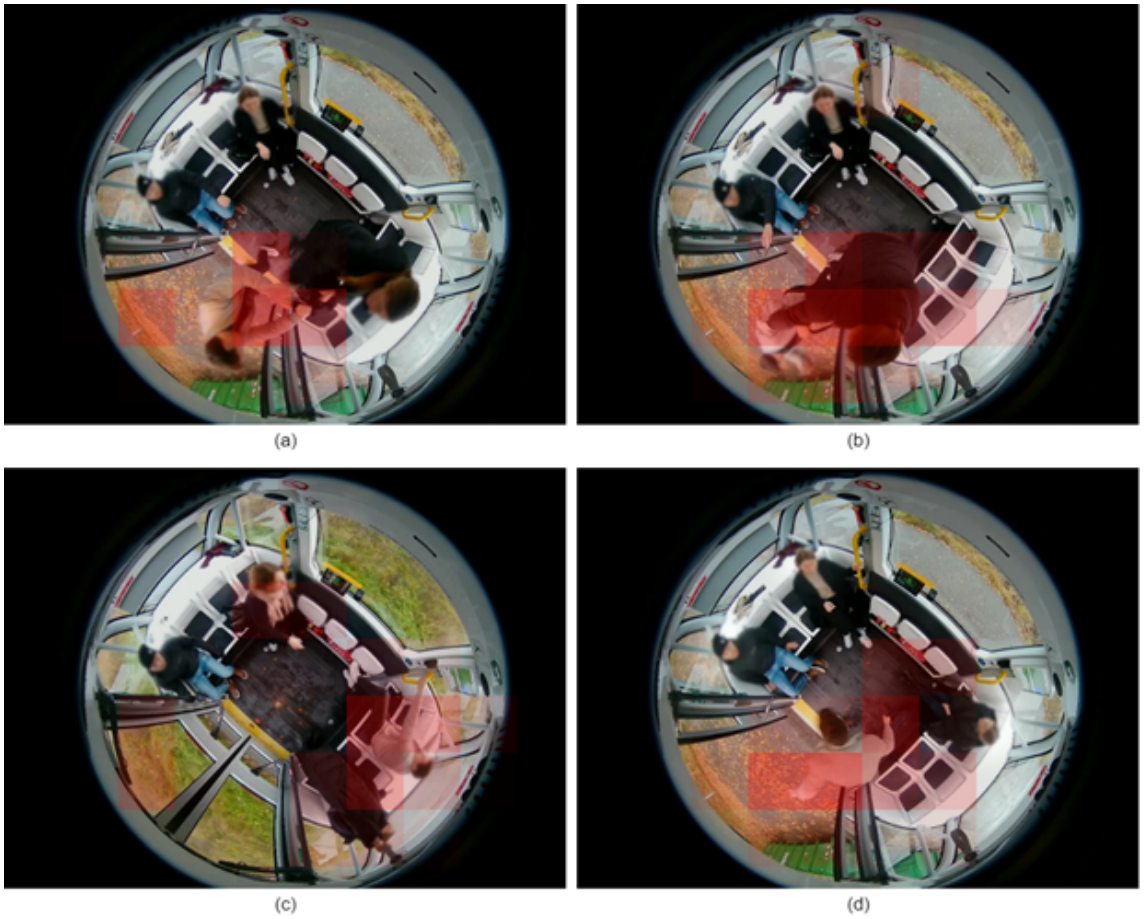

Figure 4.22: Real-time detection of abnormal events with activation maps visualization in red shades: (a) and (b) indicate a fighting event, (c) passengers falling down due to a sudden deceleration (breaking) of the vehicle, (d) bag snatching (stealing) event.

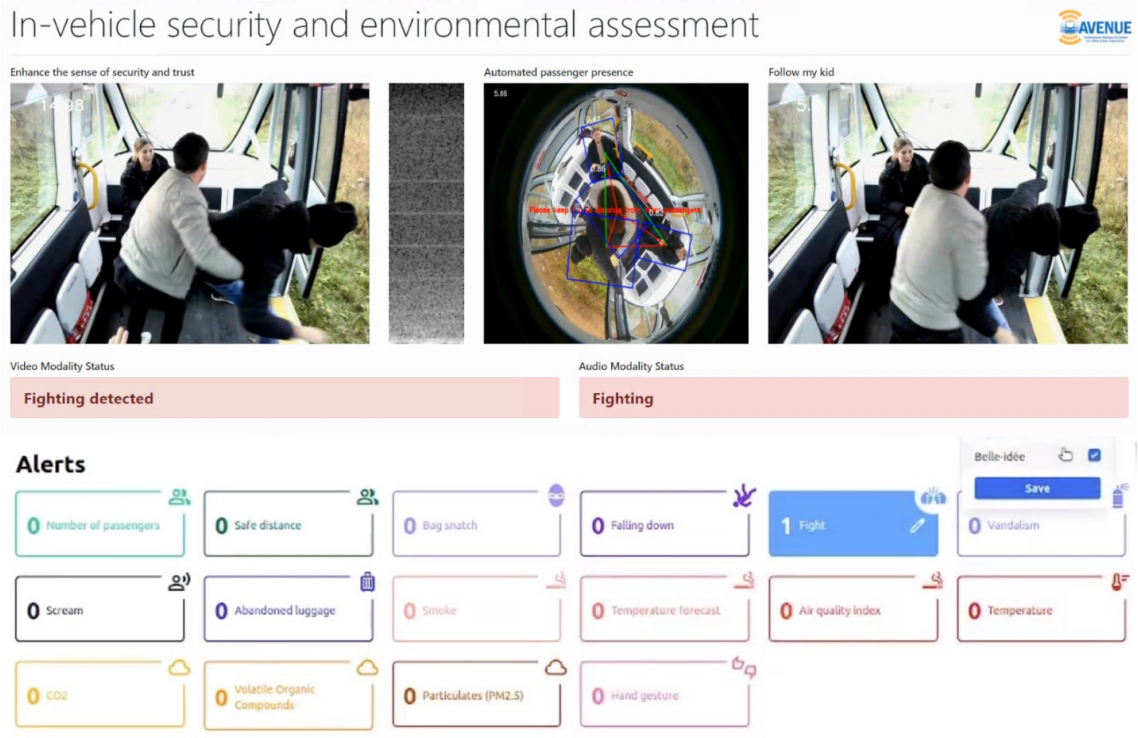

Figure 4.23: Dashboard showcasing a fighting abnormal event detection using the proposed method.

case of 0 dB SNR.

There are plenty of options to tweak in a CNN. The number of convolutional layers, max-pooling layers, filter sizes, activation functions. In our experiments, we focus on comparing the default DenseNet-121 architecture in various SNR settings.

Since the train and validation losses and F1-Scores are not sufficient for the complete evaluation of the proposed framework, we evaluated the Precision (P), Recall (R), and F1-Score (F) for each class (Table 4.7) and calculated the receiver operating characteristic (ROC) curves for each class (Figure 4.24).

From Table 4.7 we can see that while the SNR value increases, the network is able to distinguish the four classes more accurately. Specifically, for all four classes, the network achieves the highest F1-Score for SNR values higher than 15 dB. The class "gun shot" is one of the most easily distinguishable, while the "scream" class is the hardest to classify.

Regarding the multichannel spectrogram representations, a representative sample of the variation with respect to each class and SNR in the extended dataset is shown in Figure 4.25. It is evident that, as the SNR increases, the features become clearer and easier to distinguish from the background, as was

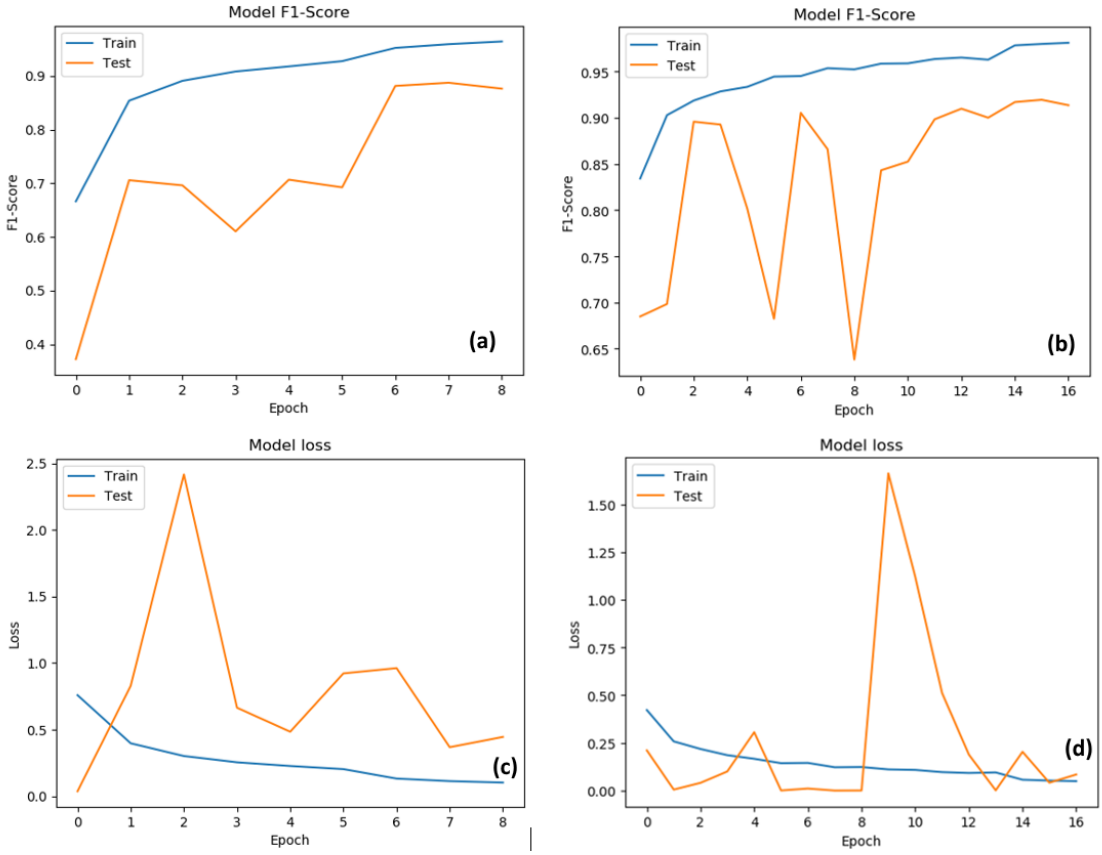

Figure 4.24: F1-Score (a) and categorical cross-entropy loss (c) for the 0 dB case of the DenseNet-121 architecture. F1-Score (b) and categorical cross-entropy loss (d) for the 30 dB case of the DenseNet-121 architecture.

Table 4.7: Precision (P), Recall (R) and F1 Score (F) metrics for the four classes at different SNR levels.

SNR (dB)	Classes											
	Background Noise			Glass Breaking			Gun Shot			Scream		
	P	R	F	P	R	F	P	R	F	P	R	F
-5	79%	86%	83%	74%	82%	78%	89%	87%	88%	85%	60%	70%
0	79%	86%	83%	74%	82%	78%	89%	87%	88%	85%	60%	70%
5	87%	90%	89%	86%	90%	88%	97%	96%	96%	88%	77%	82%
10	91%	85%	88%	80%	94%	87%	92%	94%	93%	88%	87%	87%
15	89%	91%	90%	89%	90%	90%	99%	97%	98%	88%	83%	86%
20	93%	89%	91%	88%	93%	90%	97%	98%	98%	88%	90%	89%
25	92%	90%	91%	88%	93%	90%	99%	99%	99%	89%	89%	89%
30	91%	91%	91%	87%	89%	88%	99%	99%	99%	91%	89%	90%

Table 4.8: Results (frame-by-frame) of available studies in the literature along with the results of the current work, regarding the four classes (including the background noise) of the original and the extended MIVIA Audio Events dataset.

Method Test with SNR > 0	**Representation**	**PR (%)**	**Accuracy (%)**	**MDR (%)**	**ER (%)**	**FPR (%)**
SoundNet	Gammatonegram	93.33	–	9.9	1.4	1.4
Proposed	STFT + Mel + MFCC (Stacked)	92.5	95.21	7.28	0.22	2.59

expected.

Although the focus of the present study is mainly on multichannel spectrogram representation performance (as well as studying different single channel representations) and the study of the effect of the SNR on the performance, a comparison of different studies conducted on the MIVIA Audio Events dataset is shown in Table 4.7.

The two common representations mentioned in the literature are spectrograms and gammatonegrams. The former is the traditional time-frequency visualisation, but it actually has some important differences from how sound is analysed by the ear; most significantly, the ear's frequency sub-bands get wider for higher frequencies, whereas the spectrogram has a constant bandwidth across all frequency channels. A Gammatone spectrogram or gammatonegram is a time-frequency magnitude array based on an FFT-based approximation to gammatone sub-band filters, for which the bandwidth increases with increasing central frequency.

Referring again to Table 4.8, its upper part compares the results achieved by considering the classification of positive SNR sound events only are shown. In the lower part of the table, the results achieved by including sound events with negative and null SNR to the above are exhibited. The average RRs for the three classes of interest (event-based) were 92.5% and 90.9% for the original and the extended dataset, respectively. The latter compares well with the reported value of 90.7% in [145].

Finally, in order to test the generalizability of the selected DenseNet architecture, we tested the network under three settings. The first one with the network trained on 30 dB SNR and tested on 0 dB SNR. The second one with the network trained on 0 dB SNR and tested on 30 dB SNR and finally with the network trained on 15 dB SNR and tested on 30 dB SNR. The classification reports are summarized in Figures 4.26 – 4.28.

From Figure 4.27, we notice that the network trained in an environment with the least noise cannot distinguish the classes in a noisy environment. On

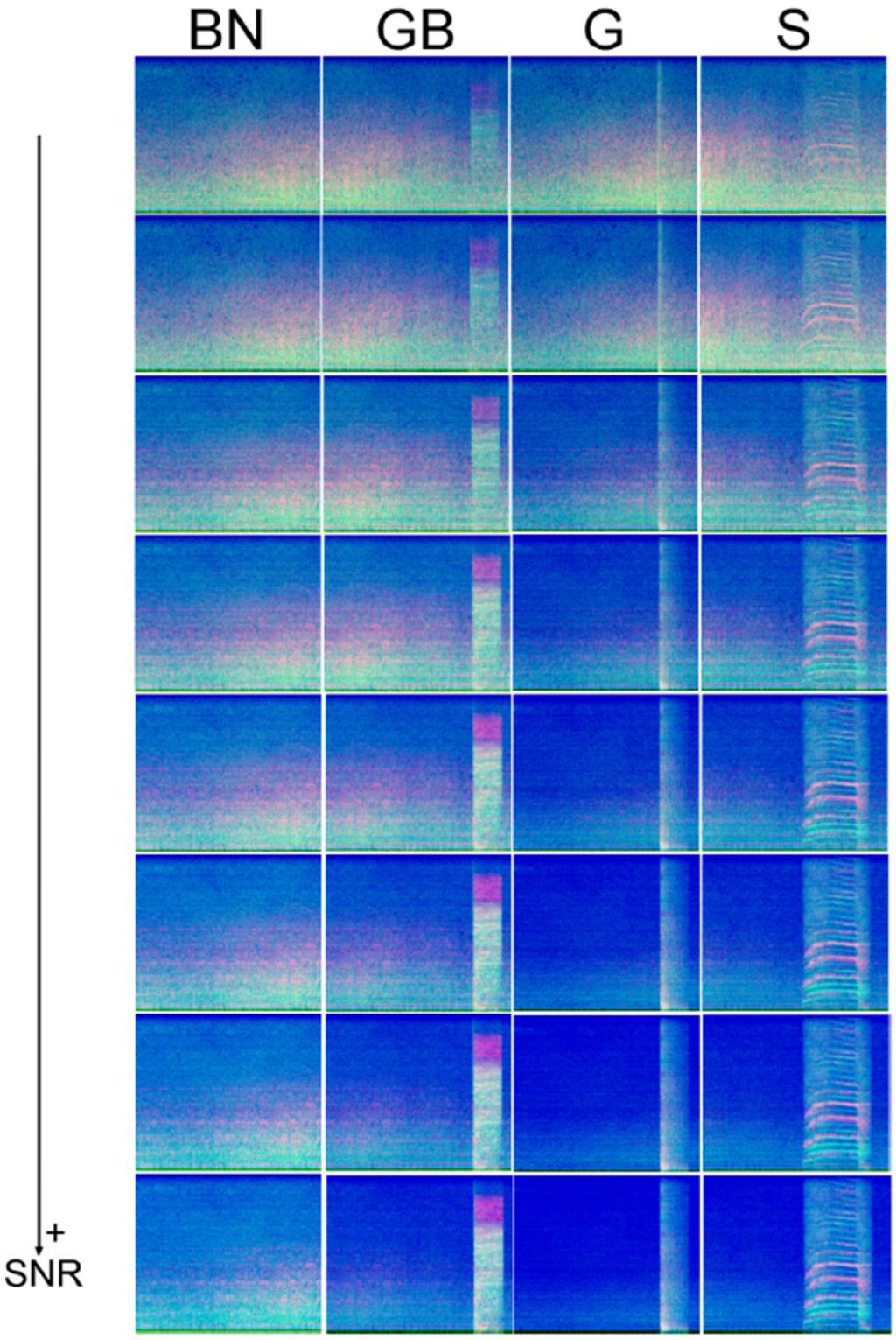

Figure 4.25: Multichannel spectrograms obtained with the stacked method: the representations of the four classes (BN for Background Noise, GB for Glass Breaking, GS for Gun Shot and S for Scream) of the extended dataset are shown, with the SNR value increasing at the direction of the arrow (left side) from -5 dB to 30 dB with a step of 5 dB.

Classification Report

	precision	recall	f1-score	support
Background Noise	0.48	0.99	0.64	2358
Glass Breaking	0.80	0.11	0.20	900
Gun Shot	1.00	0.00	0.00	900
Scream	0.82	0.04	0.07	900
accuracy			0.49	5058
macro avg	0.78	0.29	0.23	5058
weighted avg	0.69	0.49	0.35	5058

Figure 4.26: Classification report for DenseNet-121 trained on 30 dB and tested on 0 dB SNR.

Classification Report

	precision	recall	f1-score	support
Background Noise	0.71	0.84	0.77	2358
Glass Breaking	0.80	0.96	0.88	900
Gun Shot	0.97	0.28	0.44	900
Scream	0.83	0.86	0.84	900
accuracy			0.77	5058
macro avg	0.83	0.74	0.73	5058
weighted avg	0.79	0.77	0.74	5058

Figure 4.27: Classification report for DenseNet-121 trained on 0 dB and tested on 30 dB SNR.

the other hand, the network trained on a noisier environment (Figure 4.28) can distinguish the classes in the quietest environment settings almost as well as the network trained on the same environmental settings. Therefore, we notice that our network can generalize well in clean environments when trained in noisy ones.

Despite the classification reports and ROC curves, we used the t-distributed Stochastic Neighbor Embedding (t-SNE) plots to further visualize the automatic features that were learnt by the proposed 2D CNN architecture. The advantage

Classification Report

	precision	recall	f1-score	support
Background Noise	0.95	0.84	0.89	2358
Glass Breaking	0.82	0.98	0.89	900
Gun Shot	0.98	0.99	0.99	900
Scream	0.84	0.92	0.88	900
accuracy			0.91	5058
macro avg	0.90	0.93	0.91	5058
weighted avg	0.91	0.91	0.91	5058

Figure 4.28: Classification report for DenseNet-121 trained on 15 dB and tested on 30 dB.

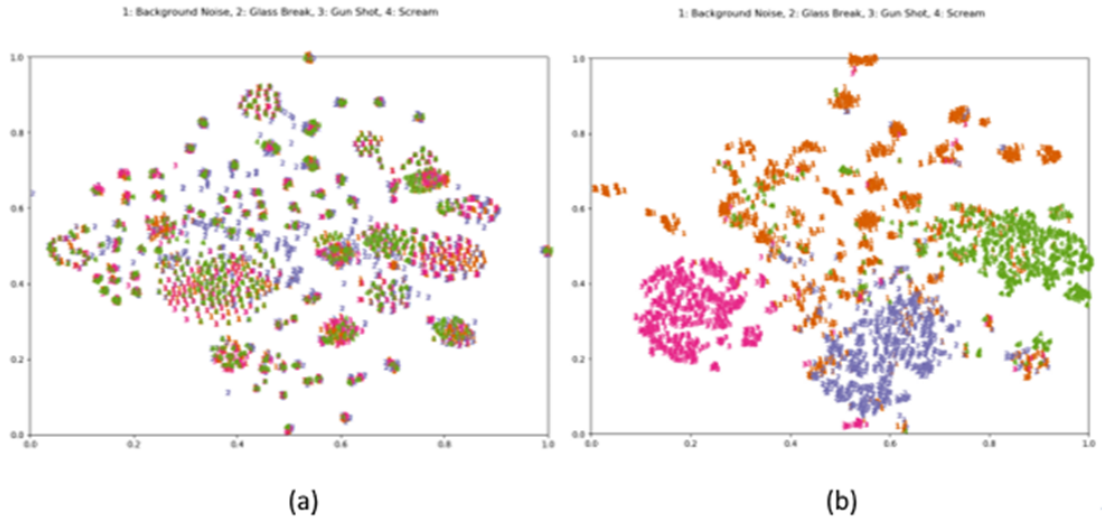

Figure 4.29: t-SNE results at 0dB.

of the t-SNE visualization, compared to a PCA, is that it uses the local relationships between points to create a low-dimensional mapping. This allows the t-SNE to capture non-linear structure of the given dataset (raw data and learnt features), since the neural network is learning non-linear representations of the dataset.

Figure 4.29 shows the t-SNE visualization at 0 dB (noisy environment) and Figure 4.30 shows the t-SNE visualization at 30 dB (quiet environment).

From the above figures we notice the randomness in the 2D space of the raw features, in both environments and on the right part of the figures the ability of the proposed 2D CNN architecture to distinguish between the target classes and create clear clusters for each class.

Since there is not a public dataset with similar sound events in a shuttle environment for in-cabin monitoring services, the proposed architecture had to be evaluated on data collected in the HOLO test track. A small dataset, as an initial phase, consisting of 200 3-second clips with ambient noise and 75 3-second events with female and male screams was recorded. All 275 sound clips were successfully classified and were not confused with glass breaking or gun shots. Collecting real data from the latter classes was not possible in this project. Additional details, with unseen during the training datasets, can be found in the paper by Papadimitriou et al. [146].

Microphone distance, ambient noise and SNR are well-known challenges in audio analysis and classification; they are factors that differentiate the latter from fields that have proven more straightforward for image analysis. The present

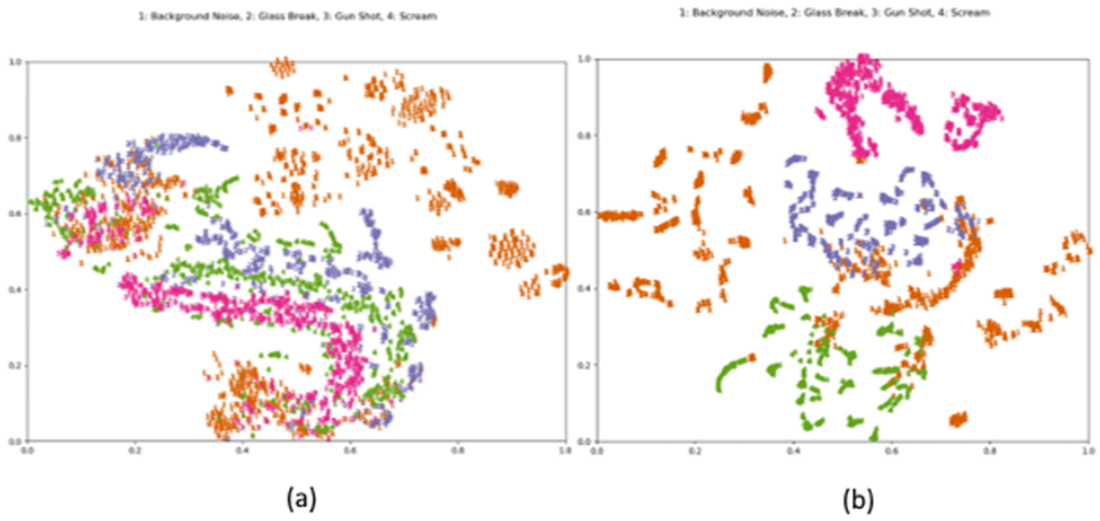

Figure 4.30: t-SNE results at 30 dB.

analysis aim was to tackle the aforementioned issues and to provide a form of analysis that generalizes well even when background noise is high and/or the signal of the event of interest is weak and the SNR drops to the negative territory.

4.3.2 Automated Passenger Counting

Moving beyond anomaly detection, we now shift into the field of object detection and more specifically in automated passenger counting (APC). APC is a crucial component for the successful integration of AVs in public transportation. APC systems enable public transport operators (PTOs) to accurately monitor the number of passengers on board in real time, which is instrumental in optimizing resource allocation, such as the allocation of vehicles to routes with higher passenger demand, and helping to ensure that AVs are efficiently utilized, reducing unnecessary energy consumption and operational costs.

Moreover, knowing the exact number of passengers onboard is crucial for maintaining safety standards, as overcrowding can lead to safety hazards and discomfort for passengers. By employing APC, AVs can prevent overloading and ensure that passengers travel comfortably and securely. Additionally, the data collected by APC systems provide valuable insights into passenger behavior and preferences, offering PTOs the ability to make decisions about route planning, scheduling, and service adjustments, improving the overall quality of public

transportation services. Finally, AVs equipped with APC systems can provide real-time information about passenger occupancy through onboard displays or mobile apps, also allowing passengers to make informed decisions about their travel plans, reducing wait times and enhancing their overall experience.

4.3.2.1 Datasets

To facilitate the development and evaluation of our APC system, we introduce the CERTH-AVenue Overhead Fisheye (C-AVOF) dataset. This novel dataset has been collected and labelled, containing frames and human objects in a simulated shuttle environment, and also including challenging scenarios such as crowded rooms, various body poses, and various-light conditions. The camera used for the data capture process is the D-Link DCS-4625 at 1080p resolution output. During the annotation of this dataset, we used DeepSort [147] for tracking the individual passengers in the shuttle across multiple frames and thus our dataset can be also used for additional vision tasks using overhead, fisheye images, such as video-object tracking and human re-identification. Also, for evaluation purposes, some cherry-picked samples from the BOSS dataset were included, especially the scenarios from camera 5 and 7 with the top-down overhead perspective.

4.3.2.2 Performance Metrics

To assess the effectiveness of our APC system, we employ a tailored set of performance metrics. Following the MS COCO challenge [5], average precision (AP) was commonly utilized as one of the evaluation criteria, especially the area under the Precision-Recall curve. However, due to the inherent ambiguity in ground truth annotations, we focus on AP at IOU = 0.5 (AP_{50}), as even with a perfect algorithm the IOU might be relatively low.

The reason behind this is the multiple bounding boxes that can exist at various angles for the same individual and a single choice of human annotator as the ground truth. Apart from AP, F-measure was also incorporated at a constant confidence threshold ($\hat{b}_{\text{conf}} = 0.3$) as another performance indicator, which, for a specific $\hat{b}_{\text{conf}}$ value, corresponds to a particular point on the Precision-Recall curve.

Table 4.9: Statistics of the new AVOF dataset in comparison with existing overhead fisheye image datasets. The dataset contains challenging scenarios in the vehicle's cabin, such as crowded conditions, occlusion scenarios, light variations and low-light conditions.

Dataset	Resolution	Segs	# avg/max	Frames	FPS
HABBOF	2048	4	3.5/5	5.837	30
CEPDOF	1080-2048	8	6.8/13	25.504	1-10
AVOF	2048	32	3.8/9	14.400	15

Scenarios	AVOF				
Stationary	2048	7	6.0	3.150	15
Moving	2048	11	6.0	4.950	15
Crowded	2048	6	10.8	2.700	15
Edge Cases	2048	4	5.5	1.800	15
Night (IR)	2048	5	6.8	2.250	15

The choice of AP_{50} was made to favor detections that are acceptable in a practical APC application, without demanding perfect alignment. F-measure at 0.3 provides an acceptable trade-off between precision and recall, avoiding false negatives and minimizing false positives. Combined, these metrics offer a robust evaluation of the model's ability to accurately detect passengers under the specific challenges presented by the overhead fisheye perspective.

4.3.2.3 Experimental Results

Building upon our methodology foundation, we now present the experimental results of our APC system. The training process was initiated on MS COCO 2017 training images [5] for 120,000 iterations, followed by fine-tuning the network on single or multiple datasets from Table 4.9 for 10,000 iterations, with each iteration comprised of 112 images.

During training on COCO images, the network weights are updated using Stochastic Gradient Descent (SGD) with a step size of 0.0005, a momentum of 0.9 and a 0.0003 weight decay. The learning rate is adjusted via a decay mechanism, reduced by a factor of 10 after every 30,000 iterations without improvement in validation loss for optimal convergence.

For the datasets listed in Table 4.9, the standard SGD was utilized with a step size of 0.0001, while rotation, flipping, shifting, resizing, and color augmentation techniques were also applied during both training stages. All results presented

here are based on a single run of training and inference. The training was conducted on a system with an Intel Core i9-9900K CPU @ 3.60GHz, 64 GB of system RAM and a single NVIDIA RTX 4090 GPU with 24GB of VRAM.

Table 4.10 provides a comparative analysis of our method with other competing algorithms. To evaluate AA and AB algorithms from Li et al. [148], we utilize the authors' publicly-available implementation. Furthermore, given the absence of a predefined train-test split in these three datasets, a cross-validation of our method was conducted, highlighting the use of two datasets for training and the remaining one for testing, repeated so each dataset is included once as the test set.

For instance, our method is trained on HABBOF and AVOF and tested on CEPDOF, and vice versa for other transformations. As neither approach from Li et al. [148] nor Tamura et al. [149] is designed to be trained on rotated bounding boxes, for the purposes of this work, they are both trained solely on the COCO dataset, as described in their respective papers. Moreover, Tamura et al. employed a top-view standard-lens image dataset called DPI-T [150] for training, in addition to the COCO dataset; however, this dataset is currently inaccessible and thus cannot be used in this study.

Table 4.10 provides a detailed performance comparison of various methods evaluated on three different datasets: HABBOF, CEPDOF, and AVOF. The evaluation is conducted on an NVIDIA RTX 4090 graphics card, which is a high-end hardware platform for DL and computer vision tasks. The performance metrics include the Average Precision at an IOU threshold of 50% (AP50), Precision (P), Recall (R), and F1-Score (F), alongside the frame rate measured in frames per second (FPS), which indicates the inference speed of each method. These metrics collectively offer insights into the accuracy, efficiency, and speed of the evaluated methods under different resolution settings, denoted in parentheses next to each method's name, indicating the input resolution scaled by a power of two. A confidence threshold of 0.3 is used for all methods to calculate Precision, Recall, and F-measure, with test results demonstrating that our method achieves the best performance in CEPDOF and AVOF and the fastest execution speed at a resolution of 1024×1024 among all tested methods. The RAPiD (608) method achieves the highest FPS of 52.5 at a lower resolution of 608, making it an attractive option for real-time applications. On the other hand, the RAPiD (1024) method showcases the best AP50 performance on the HABBOF dataset with a score of 98.1% at a lower frame rate of 27.7 FPS. This trade-off between accuracy and speed is a common challenge in the design and implementation

Table 4.10: Performance comparison of various methods on the HABBOF, CEPDOF and AVOF datasets on RTX 4090. Numbers in parentheses indicate the input resolution (multiplied by a power of two).

Method	**FPS**	**Classes**											
		HABBOF				**CEPDOF**				**AVOF**			
		AP_{50}	P	R	F	AP_{50}	P	R	F	AP_{50}	P	R	F
Tamura et al. (608) [149]	51.2	87.3	0.970	0.827	0.892	61.0	0.969	0.526	0.634	62.5	0.945	0.810	0.873
Li et al. AA (1024) [148]	3.2	87.7	0.922	0.867	0.892	73.9	0.896	0.638	0.683	75.0	0.950	0.820	0.881
Li et al. AB (1024) [148]	1.7	93.7	0.881	0.935	0.907	76.9	0.884	0.694	0.743	77.5	0.955	0.825	0.886
RAPiD (608) [104]	**52.5**	97.3	0.984	0.935	0.958	82.4	0.970	0.827	0.892	85.0	0.968	0.850	0.906
RAPiD (1024) [104]	27.7	**98.1**	0.975	0.963	0.969	85.8	0.970	0.827	0.892	87.0	0.972	0.855	0.911
Proposed (1024)	29.1	97.9	0.960	0.931	0.951	**86.1**	0.978	0.965	0.971	**92.3**	0.960	0.940	0.949

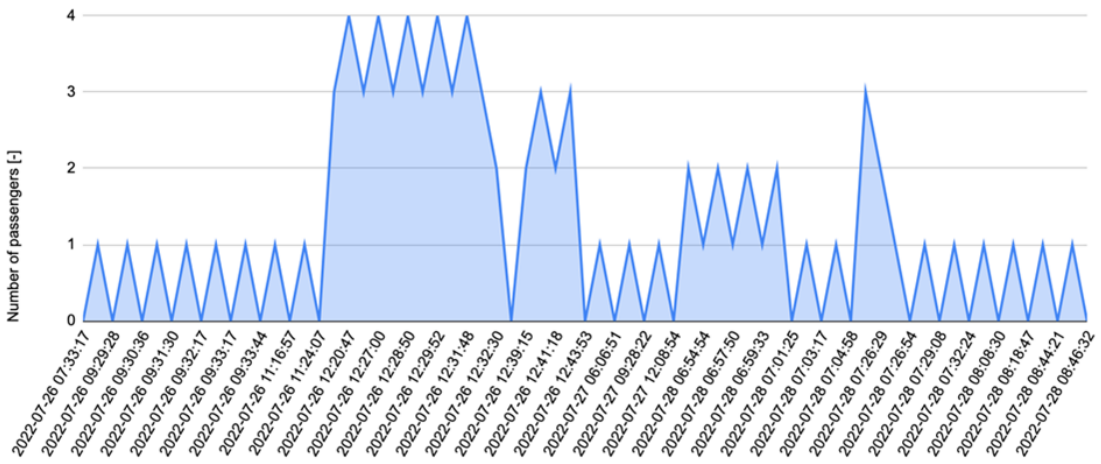

Figure 4.31: Passenger count over time received from the automated data stream.

of object detection systems. The proposed method, exhibits a balanced performance across all datasets, achieving a nearly top AP50 on HABBOF (97.9%) and the highest scores on CEPDOF (86.1% AP50) and AVOF (92.3% AP50). Notably, our method's performance at HABBOF is slightly lower than RAPiD (1024), where human objects appear in an upright pose (movement), a significant observation since people walking or standing typically exhibit radial orientation in overhead fisheye images. Our method also demonstrates a HFR of 29.1 FPS, indicating its efficiency and suitability for applications requiring both high accuracy and real-time processing.

The system was evaluated on automated minibuses in Copenhagen and Geneva. The solution was installed on NAVYA AVs, featuring a NVIDIA Jetson AGX Xavier platform and a D-Link DCS-4625 fisheye camera. The camera was connected directly to the Jetson system via Ethernet through the RTSP protocol. Both components are powered by the vehicle's batteries and Tensor-RT conversion was performed to maximize the algorithm's efficiency, reducing the power consumption to approximately 10 Watts.

The in-shuttle operator on site has already manually been counting passengers using the operator app; hence, validating the APC has been done through comparison with data received from the operator's phone data stream (Figure 4.31 and Figure 4.32).

In Figure 4.33, the manual passenger count is seen to the left, where each person getting on the shuttle is entered as 1s in the data stream. Within the same minute as the operator manually counts the 4 entries, the data stream

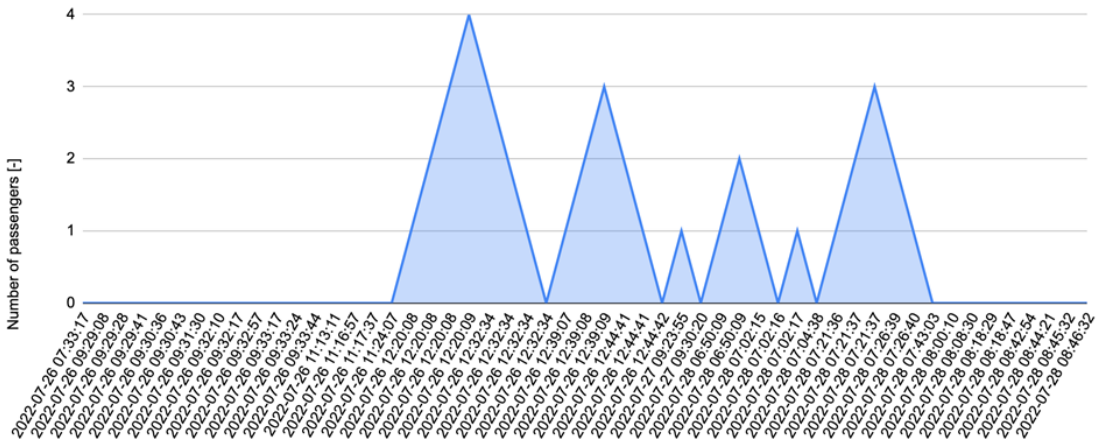

Figure 4.32: Passenger count over time received from the driver's (manual counting) data stream.

received from the Jetson increases to a count of 5 passengers (4 passengers and the operator).

After comparing all data points received within a real-world operation, the accuracy of the count was further investigated. Only the times where data from our automated approach showed more than 1 person (other than the driver) in the shuttle were extracted and compared. The timestamp is given in UTC time, meaning the actual time was (Copenhagen summer time) 2 hours ahead. By visually comparing the two data streams in Figure 4.31 and Figure 4.32, we could see that patterns indicate a similar count of passengers. Initially, an appropriate algorithm to determine the total number of passengers on board

Row	timestamp	qty
11	2022-07-26 12:20:09.609000 U...	1
12	2022-07-26 12:20:08.858000 U...	1
13	2022-07-26 12:20:08.487000 U...	1
14	2022-07-26 12:20:08.251000 U...	1

Row	timestamp	passengers...
68	2022-07-26 12:22:39.253144 U...	5
69	2022-07-26 12:21:39.427374 U...	5
70	2022-07-26 12:20:47.357969 U...	5
71	2022-07-26 12:20:15.349922 U...	4
72	2022-07-26 12:19:45.832723 U...	1
73	2022-07-26 12:18:58.261609 U...	1
74	2022-07-26 12:18:51.749958 U...	1
75	2022-07-26 12:18:19.611659 U...	1

Figure 4.33: Data from BigQuery: four passengers are getting on the shuttle. In the right-side the manual count from the operator's app is depicted with data points received when state changes (button is pushed in the shuttle). In the left side the data points are received continuously from our automated method. The count includes the on-board operator.

Figure 4.34: Results from the proposed method (left side) and [104] (right side). The proposed method detects correctly the two passengers that are partially occluded.

was developed that counted only the increases in passenger numbers. However, the oscillating nature of the automated counting data results in a higher total passenger count, combined with the occasional inaccuracies in the exact number of onboard passengers, which pose an ongoing challenge. A post-processing filtering method would help mitigate the error.

Based on all of the aforementioned information, the main findings of this study suggest that our approach can be successfully implemented across both simple and challenging tasks, while simultaneously maintaining high computational efficiency. Furthermore, it was found that the network's performance is enhanced when the input image resolution is increased to 1024×1024, at the expense, however, of doubled inference time. Sample results can also be seen for the three datasets in Figure 4.34, demonstrating nearly flawless detection across various scenarios, including diverse body poses, orientations, and backgrounds.

However, certain scenarios, such as images of people on a projection screen, low-light conditions, and hard shadows, continue to pose challenges. Our study encounters the challenge of false positives as Figure 4.35 illustrates. The algorithm, while adept at identifying individuals with a high degree of accuracy, as denoted by the green bounding box, also exhibits some erroneous classifications.

An indicative example is the detection of a non-human object - specifically, a piece of cloth - as a person, highlighted by the red bounding box. Such false positives are not only statistical outliers, but also highlight the complexities that these algorithms must navigate. The discriminative power of the algorithm could be tuned to differentiate between human figures and objects similar in shape or size to mitigate the incidence of false positives and enhance the robustness of the detection system in diverse operational environments.

Figure 4.35: Illustration of a false positive: the green bounding box represents a correctly identified individual, whereas the red bounding box indicates a false positive detection by the algorithm, misidentifying a piece of cloth as a person. This example highlights the challenges in discriminating between actual human figures and objects with similar form factors in complex visual scenes.

Figure 4.34 illustrates a real-time testing scenario, showcasing a comparison between the proposed method (left side) and [104] (right side) in a crowded scenario, as captured through the fisheye-lens camera. The proposed method correctly detects the two passengers that are partially occluded.

On the other hand, Figure 4.35 illustrates both a correct and a false positive scenario, where the green bounding box successfully identifies a real person, accurately detecting their presence as they stand near the entry point, whereas the jacket included in the red box is falsely identified as a passenger. This misidentification underscores a significant challenge faced by detection algorithms for distinguishing between humans and inanimate objects with similar size or shape, especially when viewed from unconventional angles in complex visual scenes.

Finally, Figure 4.36 illustrates a sequence of extreme and rapid lighting variations caused by shadows and the vehicle's motion. This is a perfect example of a delayed exposure adaptation from the camera sensor, despite featuring WDR. The sudden change in slide 3 results in blown highlights in the image, blending the person's appearance with the vehicle color. This loss of detail results in a failure in the detection of the passenger.

In Figure 4.37 and Figure 4.38, we can see additional visualized results on unseen scenarios from the BOSS dataset, representing a safe distance between onboard passengers in the shuttle. The blue bounding boxes indicate the de-

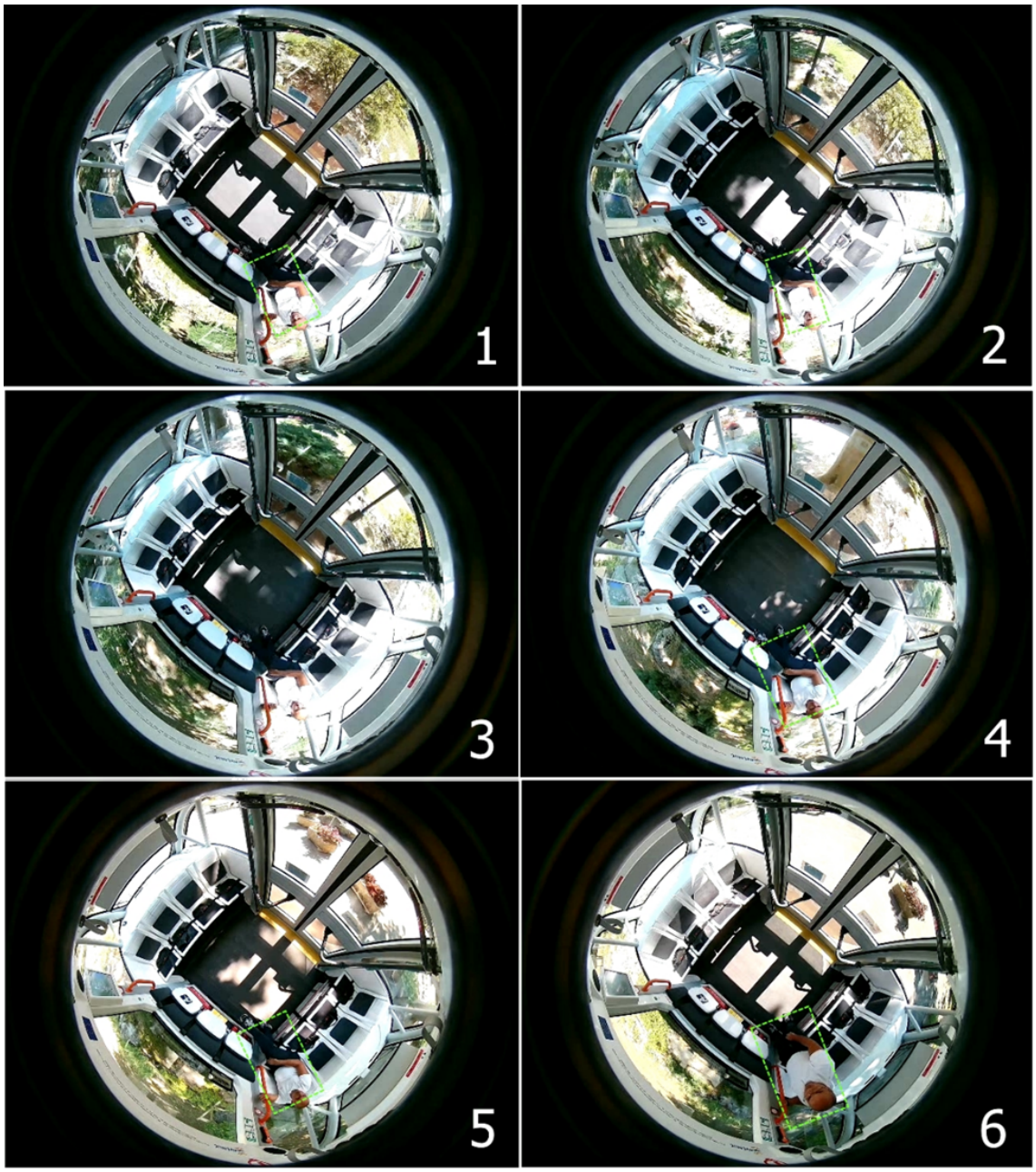

Figure 4.36: Light variations can cause loss of detail in the camera stream, especially without WDR capable sensors.

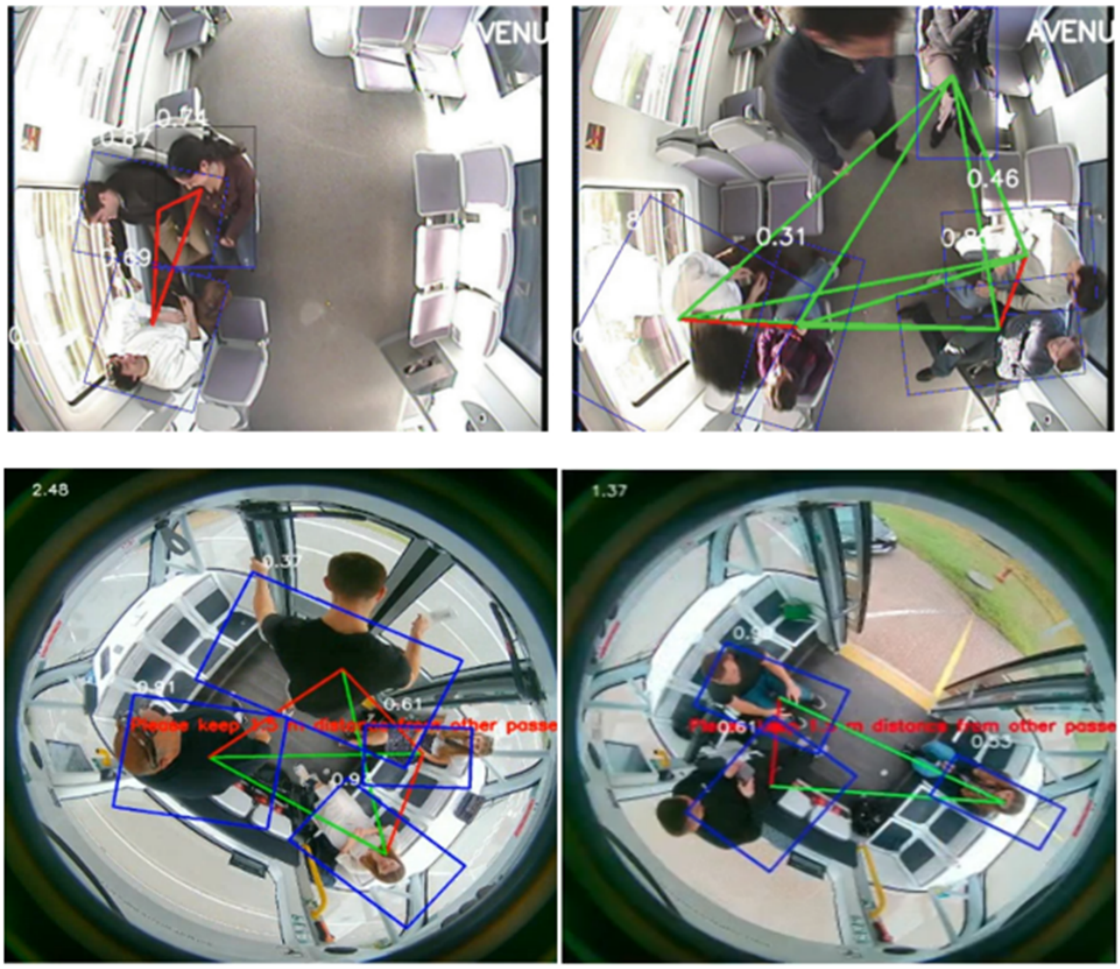

Figure 4.37: Results on unseen scenarios from the BOSS and HOLO+TPG datasets. Green lines represent a safe distance while red lines an unsafe one.

tections, and the numbers represent the confidence of each detection. Red lines denote an unsafe distance while lines in green a sufficient distance.

4.3.3 Face Identification

Transitioning from APC and object detection to individual identification, we now focus on the development and implementation of a face identification service. In the following sections, the research, involving algorithms and experiments conducted, is presented for the face identification service. As depicted in Figure 4.39, the first layer of sensors connects to the Hardware Abstraction Layer (HAL). The HAL implements the IP and the USB protocol supporting IP and USB cameras respectively, but also can request raw data by the API endpoints in

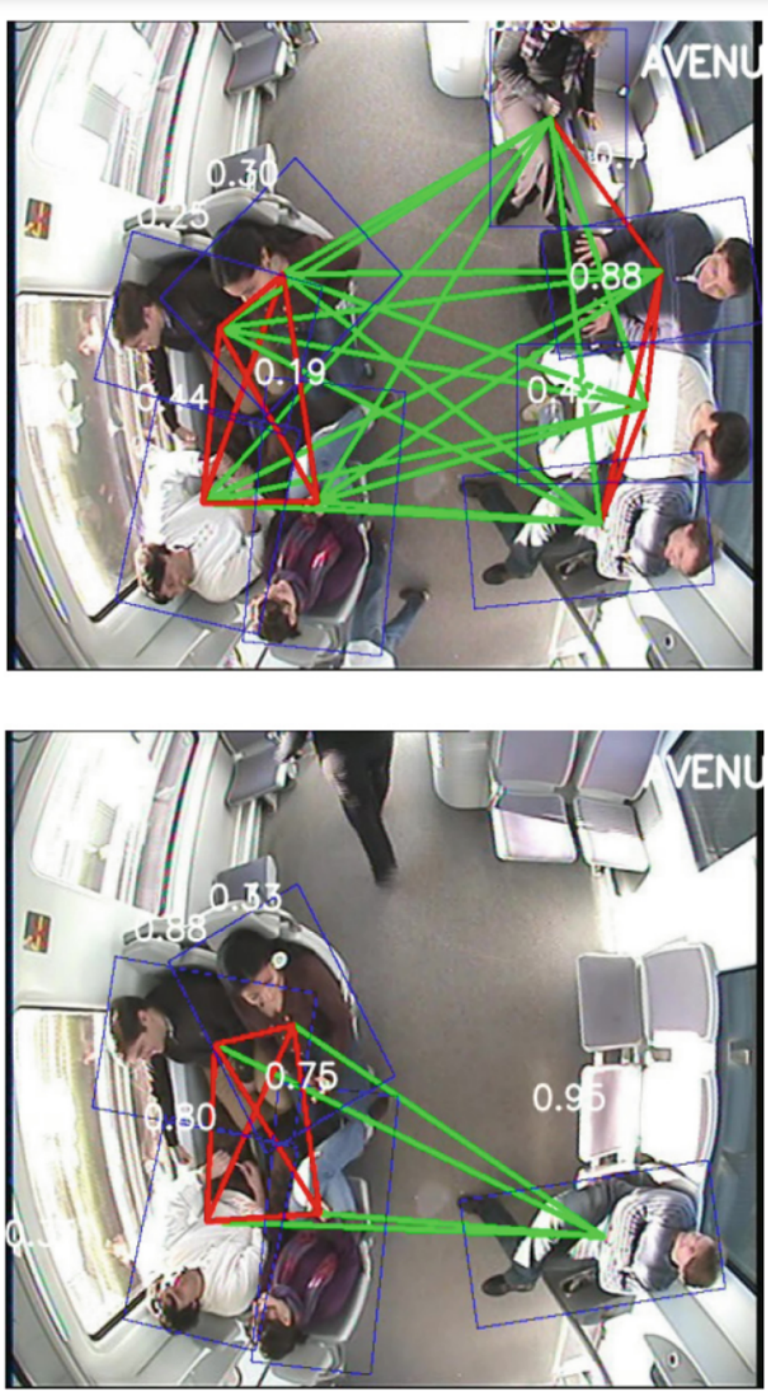

Figure 4.38: Results on an unseen crowded scenario from the BOSS dataset, with overlapping detections.

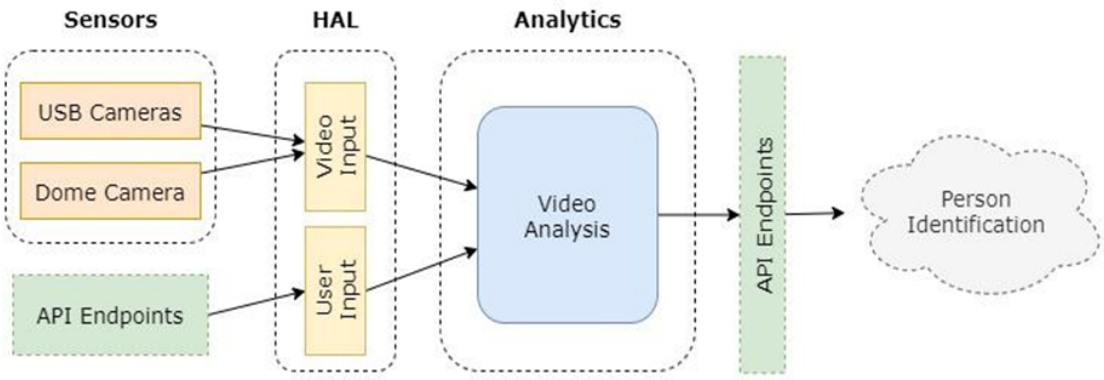

Figure 4.39: The diagram illustrates a video analysis and person identification system architecture. The system consists of three main components: Sensors, a HAL, and Analytics. The Sensors component captures video data from USB and dome cameras, while also receiving input from API endpoints. The HAL manages video and user input, passing the data to the Analytics component, which performs video analysis and sends the results to API endpoints for person identification.

order to perform face recognition. The input data is converted and transformed in a compatible format and passed into the analytics algorithms. The prediction is then transferred via the API endpoints into the cloud. The user has access to the data and acts accordingly.

Given the dynamic nature of passengers in a public transportation setting, where new faces are encountered continuously, the ability to quickly and accurately identify individuals from limited samples becomes crucial. This is where the concept of one-shot learning plays a crucial role in our face identification system.

One-shot learning is a classification task where one, or a few, examples are used to classify many new examples in the future. This characterizes tasks seen in the field of face recognition, such as face identification and face verification, where people must be classified correctly with different facial expressions, lighting conditions, accessories, and hairstyles given one or a few template photos. Modern face recognition systems approach the problem of one-shot learning via face recognition by learning a rich low-dimensional feature representation, called a face embedding, that can be calculated for faces easily and compared for verification and identification tasks. Historically, embeddings were learned for one-shot learning problems using a Siamese network (Figure 4.40). The training of Siamese networks with comparative loss functions resulted in better performance, later leading to the triplet loss function used in the FaceNet [75] system by Google that achieved state-of-the-art results on benchmark face recognition tasks.

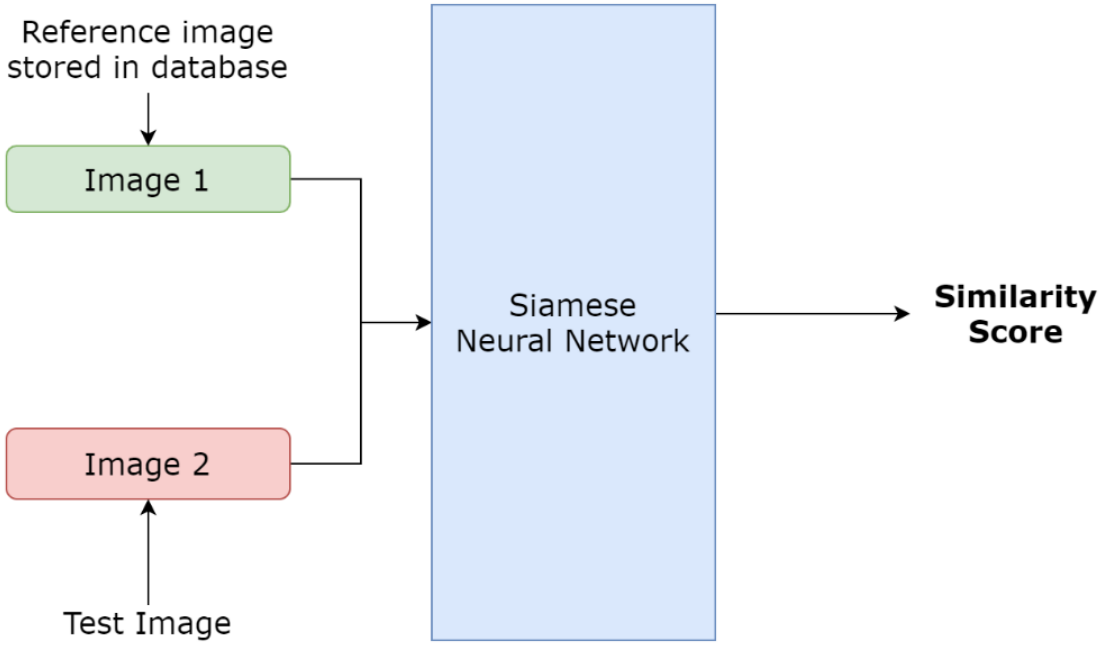

Figure 4.40: Diagram of a facial verification process using a Siamese Neural Network. The process involves comparing a reference image (Image 1) stored in a database with a test image (Image 2) to determine if they belong to the same individual. The Siamese network takes both images as input and produces a similarity score, indicating the likelihood of a match.

Instead of directly classifying an input (test) image to one of the 10 people in the shuttle, this network instead takes an extra reference image of the person as input and will produce a similarity score denoting the chances that the two input images belong to the same person. Typically, the similarity score is squished between 0 and 1 using a sigmoid function; wherein 0 denotes no similarity and 1 denotes full similarity. Any number between 0 and 1 is interpreted accordingly. Notice that this network is not learning to classify an image directly to any of the output classes. Rather, it is learning a similarity function, which takes two images as input and expresses how similar they are. A new passenger can be enrolled to the face recognition service using a single image of his face which will be stored in a database. Using this as the reference image, the network will calculate the similarity for any new instance presented to it. Thus, we conclude that the network can predict the score in one shot.

4.3.3.1 Datasets

To train and evaluate our one-shot learning model for face identification, we utilize a combination of existing and synthetic datasets. We used the MS1M-

Table 4.11: Accuracy comparison between methods and different datasets indicate a significant improvement of about 6.2% in our augmented datasets that contain face masks.

Method	Training dataset	Testing dataset	Masks	Accuracy
ArcFace	MS1M-ArcFace	LFW	No	99.83%
ArcFace-xCos	MS1M-ArcFace	LFW	No	99.35%
ArcFace+M	MS1M-ArcFace+M	LFW+M	Yes	68.33%
Ours, ArcFace-xCos+M	MS1M-ArcFace+M	LFW+M	Yes	74.52%

ArcFace [151] dataset for training our network and the LFW [74] for the testing. The two datasets were augmented via pre-processing to include face masks using the MaskTheFace tool [152]. For the sake of simplicity, we name the synthetic datasets MS1M-ArcFace+M and LFW+M, respectively.

4.3.3.2 Result Analysis

Following the technical details and experimental setup, we now present a comprehensive analysis of our face identification system's performance. This analysis involves a multi-faceted approach, encompassing real-world evaluations in operational settings, cross-validation with manual identification, and comparisons with existing mobile applications. More specifically, Table 4.11 shows the testing accuracy on the original MS1M-ArcFace, LFW and the artificially created MS1M-ArcFace+M and LFW+M, which contain additional masked samples. As we can see, despite the minimal loss caused by the explainability module on the original datasets, a significant improvement of about 6.2% has been accomplished in our augmented datasets which contain face masks.

Figure 4.41 also highlights the improvements made over the original ArcFace+M. The maps are generated (a) by the original ArcFace+M while (b) by our improved model. The cosine similarity between the two faces is negative both in (a) and (b) on the bottom half of the faces as the mask portrays different characteristics such as shape and color. However, the attention map in our improved (b) model indicated that the network focuses more on the upper half characteristics around the eyes and the nose.

Figure 4.43 showcases a real-world scenario on an autonomous shuttle, running on the NVIDIA Jetson AGX Xavier. The proposed system is able to correctly identify the two passengers among a database of 20 people.

Figure 4.44 shows additional results of our improved model. The attention

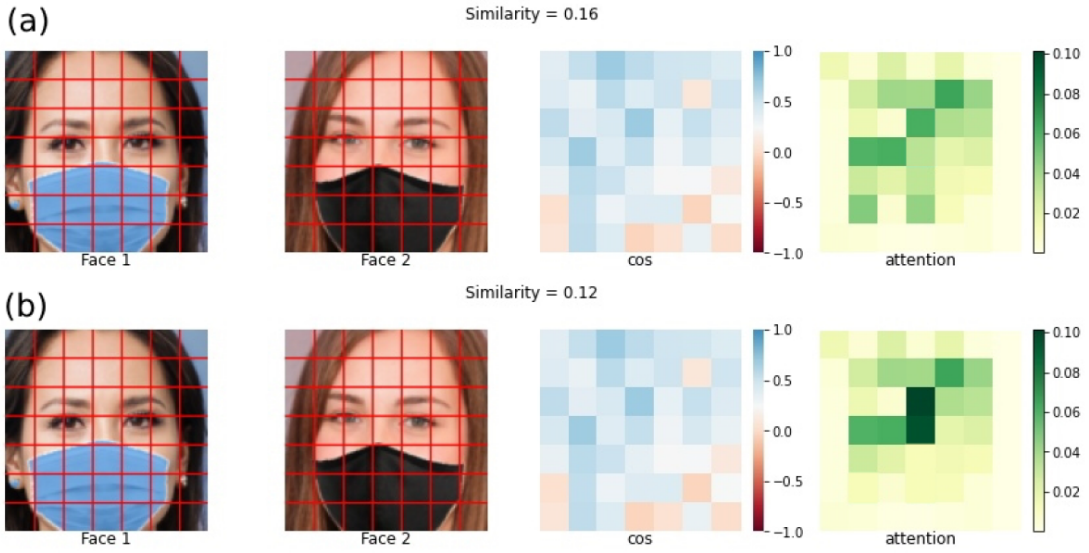

Figure 4.41: Input and maps generated by (a) the original ArcFace+M and (b) our improved model. Although the cosine (cos) similarity is negative both in (a) and (b) on the bottom half of the faces, the attention map in our improved (b) model indicated that the network focuses more on the upper half characteristics around the eyes and the nose, ignoring the region covered by the mask.

Figure 4.42: An experimental real-world demonstration running on the NVIDIA Jetson AGX Xavier. The proposed system is able to correctly identify the passengers of the autonomous shuttle even with their face masks on, among a database of 10 people.

Figure 4.43: An experimental real-world demonstration running on the NVIDIA Jetson AGX Xavier. The proposed system is able to correctly identify the two passengers of the autonomous shuttle, among a database of 20 people.

maps verify that the bottom parts of the face are efficiently ignored while the upper face characteristics have the most significant impact on calculating the similarity score.

The system was designed as a flexible and end-to-end service accessible by an Android mobile application. In Figure 4.45, the architecture of the proposed system is depicted. A new passenger can be enrolled to the service via the mobile application, using a single image of his face. The image will be processed and stored temporarily in a database on the cloud platform. By using this as the reference image, the network calculates the similarity for any new instance presented to it from the shuttle. The network can predict a similarity score in one shot and inform the client via an API call and relevant notification through the mobile phone. The proposed approach can be further extended to support homeland security surveillance infrastructures in order to mitigate domestic security risks. In this context, it is envisioned to establish a highly adaptable security framework capable of leveraging the capabilities of autonomous public transport operators as well as law enforcement agencies.

4.3.4 Computational Efficiency

Having explored the results of various anomaly detection modalities, we now shift our focus to the computational efficiency results of our system.

Based on our methodology foundations, the proposed multimodal detection system is designed to operate in real time within the constraints of an embedded system, ensuring prompt detection of critical events in autonomous shuttles. To optimize computational efficiency, we employ several strategies. Firstly, the CAE architecture utilizes depth-wise separable convolutions, which significantly reduce parameter size compared to standard convolutions. Moreover, network optimizations via tools like TensorRT enable the model to run efficiently on the NVIDIA Jetson platform, which possesses limited computational resources when compared to desktop GPUs.

Experimental results demonstrate that our approach achieves a processing speed of approximately 37 frames per second (FPS) using the Performance (MAX-N) mode on the NVIDIA Jetson AGX Xavier embedded system. However, we are restricting the processing framerate to 15 FPS accumulating a sliding buffer of 3 seconds. The use of TensorRT optimizations (layer fusion, kernel optimizations 4.2, quantization 4.1, and dynamic tensor memory optimization)

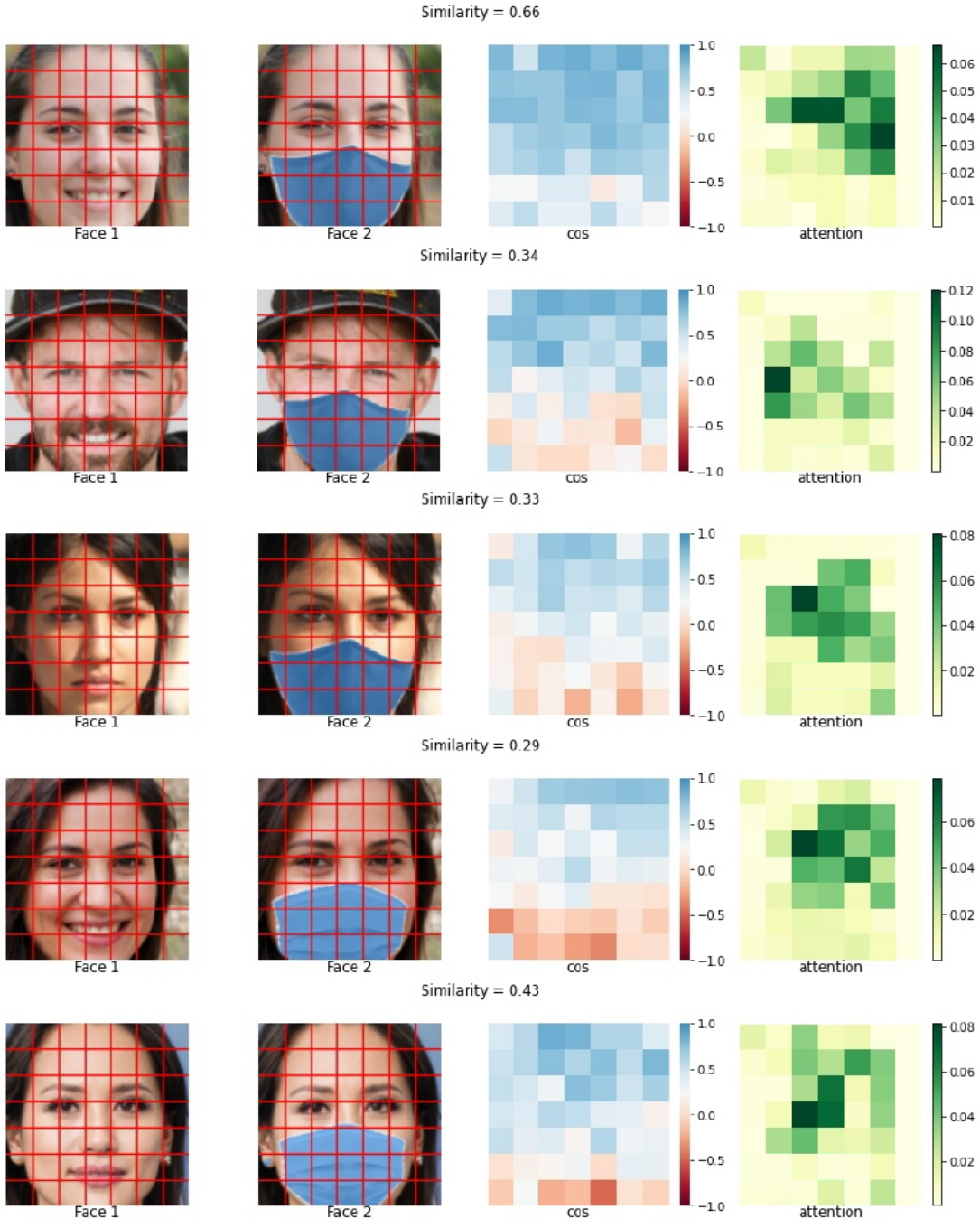

Figure 4.44: Additional results of our improved model. Facial images are artificially generated using StyleGAN [153] and post-processed to include masks using the MaskTheFace [152].

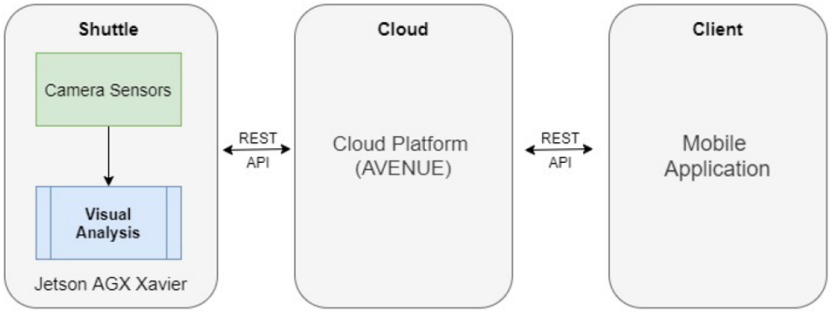

Figure 4.45: The architecture of the proposed system. A new passenger is enrolled by the client. A single image of his face will be processed and stored temporarily in a database and the network will calculate similarities with current passengers. The client is informed via an API call and relevant notification through the mobile phone.

along with a reduced processing frame rate has reduced the power consumption to an acceptable level for the AV's battery.

Listing 4.1: Enabling Kernel Auto-Tuning in TensorRT

```
def build_engine_with_auto_tuning(model_path):
     trt.OnnxParser(network, TRT_LOGGER) as parser:
        with open(model_path, 'rb') as model:
            parser.parse(model.read())
        builder.max_workspace_size = 1 << 30  # 1 GB
        builder.int8_mode = True  # Enable INT8 mode
        builder.int8_calibrator = trt.Int8EntropyCalibrator2(
            calibration_dataset)
        engine = builder.build_cuda_engine(network)
        return engine
```

Listing 4.2: Enabling Kernel Auto-Tuning in TensorRT

```
def optimize_kernels(network):
    for layer in network:
        if layer.type == trt.LayerType.CONVOLUTION:
            layer.precision = trt.DataType.FLOAT16  # Use FP16
                precision
            layer.set_output_type(0, trt.DataType.FLOAT16)
        if layer.type == trt.LayerType.FULLY_CONNECTED:
            layer.precision = trt.DataType.INT8  # Use INT8
                precision
            layer.set_output_type(0, trt.DataType.INT8)
    return network
```

The NVIDIA Jetson AGX Xavier platform, running our framework, operates at an average power of approximately 10 watts, as measured by the tegrastats utility. The D-Link DCS-4625 fisheye camera has an average power consump-

tion of 3.5 watts, based on the manufacturer's specifications. The consumption is considered as low and sufficient for real-time deployment within power-constrained autonomous shuttles.

4.4 Chapter Summary

This chapter presented the real-world deployment and thorough examination of the proposed multimodal framework in the field of transportation and more specifically AVs. For this purpose, the primary safety concerns associated with AVs have been presented, including psychological discomfort due to the lack of human supervision and physical safety challenges inside the autonomous environment, ultimately highlighting the importance of the proposed framework at augmenting passenger safety and trust in fully autonomous public transportation systems.

In this chapter, an AI framework based our methodology foundation detects and responds to a variety of safety-related incidents within AVs, showcasing the use of advanced safety monitoring systems equipped with an array of sensors and cameras to manage both external driving conditions and internal passenger safety effectively. Practical applications and case studies were also presented to illustrate how the proposed AI framework can be utilized in real-world scenarios, showcasing examples including passenger counting, abnormal event detection (such as aggression or health emergencies), as well as adherence to health regulations.

5

Discussion, Future Work and Conclusions

In this chapter, a comprehensive discussion on the key findings of this dissertation, as well as its main contributions in the field of AI and edge computing are offered. This section also presents some key extensions to enhance the capabilities and impact of the proposed accelerated multimodal AI framework for edge computing.

5.1 Discussion

This dissertation aims to address the edge computing challenges by proposing a novel design methodology for edge processing, leading to an accelerated multimodal framework tailored for these environments. This framework enables the execution of complex and in-depth data processing directly at the source, leveraging novel AI models and optimizations for various applications, such as abnormal event detection, object recognition, proximity assessment, and facial recognition.

One of the key findings of this research is the successful integration and fusion of multiple data modalities within the framework. By combining vi-

sual, audio, and sensor data, the framework achieved a more comprehensive understanding of complex scenarios, leading to significant improvements in the accuracy and robustness of event detection and recognition tasks compared to single-modality systems. The emphasis on hardware acceleration and software optimization techniques effectively mitigated the resource constraints in embedded devices.

These approaches demonstrate significant improvements in decision-making capabilities, response times, and overall system performance, even with the limited resources of embedded systems. To mitigate the computational overhead of data preprocessing, we have implemented various computational processing designs in hardware, taking advantage of the parallelism of FPGAs. We introduced a VHDL design for color transformation and Sobel edge detection, further improved using HLS for increased efficiency. We also explored an accelerated noise reduction technique based on image stacking, highlighting the potential of hardware acceleration for edge processing.

The findings of this dissertation hold significant implications for the future of AI and edge computing. By enabling efficient and versatile multimodal processing on embedded devices, the proposed framework builds the foundation for a new generation of intelligent systems capable of operating autonomously and adapting to dynamic environments. This advancement promises to revolutionize various industries, from transportation and healthcare to manufacturing and smart cities.

In conclusion, this dissertation has made a substantial contribution to the field of AI and edge computing. As AI technology continues to evolve, further research and development in this area will lead to even more innovative and impactful applications, shaping a future where intelligent systems seamlessly integrate into our lives, enhancing safety, efficiency, and overall well-being.

5.2 Future Work

Building upon the foundation established in this dissertation, several extensions as future work emerge, promising to further enhance the capabilities and impact of the proposed accelerated multimodal AI framework for edge computing.

One key area of future research lies in exploring advanced privacy-preserving techniques. As AI systems become increasingly integrated into our daily lives, concerns regarding data privacy and security are raised. Investigating meth-

ods such as federated learning, where models are trained collaboratively across multiple devices without sharing raw data, can ensure user privacy while still enabling effective model training and performance improvement. Additionally, exploring techniques like differential privacy and homomorphic encryption can further enhance data security, protecting sensitive information during processing and analysis.

Another promising direction is the development of more robust and adaptable AI models capable of handling even greater data diversity and complexity. The current framework has demonstrated its effectiveness across various modalities, but as technology evolves, AI systems will encounter increasingly challenging data streams and scenarios. Exploring advanced DL architectures, such as transformers and graph neural networks, can render the framework capable for learning more complex relationships and patterns within data. Furthermore, investigating techniques like continual learning and meta-learning can enable the framework to adapt to new data types and tasks with minimal additional training, enhancing its versatility and long-term viability.

To address the challenge of scalability in distributed edge computing environments, future work can focus on implementing distributed training and inference mechanisms. Exploring technologies like Kubernetes and Docker Swarm can facilitate the efficient deployment and management of AI models across a network of edge devices, enabling seamless scaling and resource allocation. Additionally, investigating distributed learning paradigms, such as model parallelism and data parallelism, can further enhance the framework's ability to handle large-scale data processing tasks efficiently.

Finally, the integration of the proposed framework with emerging technologies, such as AR and virtual reality (VR), presents exciting possibilities. By combining the framework's ability to interpret real-world data with the immersive experiences offered by AR/VR, novel applications in areas like healthcare training, remote assistance, and interactive education can be developed. This convergence of AI, edge computing, and immersive technologies promises to unlock new levels of human-computer interaction and create transformative experiences across various domains.

In summary, the research presented in this dissertation serves as a base for further exploration and development in the dynamic field of AI and edge computing. By pursuing the aforementioned future directions, the proposed framework can evolve into an even more powerful and versatile tool, contributing significantly to the advancement of intelligent systems and their integration

into our lives, ultimately shaping a future where technology seamlessly interacts with and enhances the human experience.

5.3 Conclusions

This dissertation addresses edge computing challenges by presenting a novel design methodology for edge processing. This methodology resulted in an accelerated multimodal framework tailored for edge computing, enabling complex data processing directly at the source. The framework leverages innovative artificial intelligence models and optimizations, tailored for applications like action recognition, abnormal event detection, object detection, proximity assessment, and facial recognition.

This research focuses into the core of this design methodology, detailing the specific techniques and approaches that led to the development of the accelerated multimodal framework. It provides in-depth explanation of the AI models and optimizations, showcasing how they were integrated to enhance processing capabilities at the edge. The dissertation explores the various stages of the methodology, elucidating the decision-making processes and trade-offs involved in shaping the framework.

In this dissertation, we proved the significant advancement of edge computing facilitated by the utilization of complex multimodal sensor data directly at the source, as detailed in Chapter 4. We presented efficient AI models and optimizations in Chapter 3, Section 3.1, and validated their effectiveness in various applications: abnormal event detection [154], [99], [155], overhead object detection [156], proximity detection [157], and facial recognition [158]. These approaches conclusively demonstrated that accelerated edge processing, even within the resource constraints of edge devices, leads to significantly enhanced decision-making, faster response times, and improved overall system performance.

Furthermore, to address preprocessing overheads, we implemented various computational processing designs in hardware, leveraging the parallelism of FPGAs, as discussed in Chapter 3, Section 3.2. We introduced a VHDL design for color transformation and Sobel edge detection on the Altera DE2-115 [91], and further optimized this design using HLS on the Xilinx Pynq-Z1 [159], enabling direct comparison of both techniques. Additionally, we investigated an accelerated noise reduction approach employing image stacking with HLS [160].

These contributions collectively highlight the potential of accelerated multimodal frameworks in revolutionizing edge computing, enabling real-time, intelligent processing of complex data streams within resource-constrained environments.

The implications of this research extend beyond the specific applications explored in this dissertation. As edge computing and AI technologies continue to evolve, the demand for efficient and versatile multimodal processing will only grow. The framework developed here serves as a foundation for a new generation of intelligent systems capable of operating autonomously, adapting to dynamic environments, and ultimately enhancing human lives. Looking ahead, the future of AI and edge computing is continuously pushing the boundaries of research and development. By exploring novel algorithms, and embracing emerging technologies, we can create a future where intelligent systems are seamlessly integrated into our world, empowering individuals, transforming industries, and shaping a more efficient, sustainable, and equitable society. The journey towards this future begins with the work presented in this dissertation, an evidence of the transformative power of AI and its potential to revolutionize the way we live, work, and interact with the world around us.

Bibliography

[1] D. O'Keeffe, T. Salonidis, and P. Pietzuch, "Frontier: Resilient edge processing for the internet of things", *Proceedings of the VLDB Endowment*, vol. 11, no. 10, pp. 1178–1191, 2018.

[2] C.-H. Lu and X.-Z. Lin, "Toward direct edge-to-edge transfer learning for iot-enabled edge cameras", *IEEE Internet of Things Journal*, vol. 8, no. 6, pp. 4931–4943, 2020.

[3] S. Liu, L. Liu, J. Tang, B. Yu, Y. Wang, and W. Shi, "Edge computing for autonomous driving: Opportunities and challenges", *Proceedings of the IEEE*, vol. 107, no. 8, pp. 1697–1716, 2019.

[4] R. A. Cooke and S. A. Fahmy, "Quantifying the latency benefits of near-edge and in-network fpga acceleration", in *Proceedings of the Third ACM International Workshop on Edge Systems, Analytics and Networking*, 2020, pp. 7–12.

[5] T.-Y. Lin, M. Maire, S. Belongie, *et al.*, "Microsoft coco: Common objects in context", in *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, Springer, 2014, pp. 740–755.

[6] J. Kim, S. Chang, and N. Kwak, "Pqk: Model compression via pruning, quantization, and knowledge distillation", *arXiv preprint arXiv:2106.14681*, 2021.

[7] J. Zhang, B. Chen, Y. Zhao, X. Cheng, and F. Hu, "Data security and privacy-preserving in edge computing paradigm: Survey and open issues", *IEEE access*, vol. 6, pp. 18 209–18 237, 2018.

[8] J. Gu, Z. Wang, J. Kuen, *et al.*, "Recent advances in convolutional neural networks", *Pattern recognition*, vol. 77, pp. 354–377, 2018.

[9] V. Antoniadou, "Collecting, organizing and analyzing multimodal data sets: The contributions of caqdas.", *Research-publishing. net*, 2017.

[10] J. Gao, P. Li, Z. Chen, and J. Zhang, "A survey on deep learning for multimodal data fusion", *Neural Computation*, vol. 32, no. 5, pp. 829–864, 2020.

[11] J. Summaira, X. Li, A. M. Shoib, S. Li, and J. Abdul, "Recent advances and trends in multimodal deep learning: A review", *arXiv preprint arXiv:2105.11087*, 2021.

[12] D. Lahat, T. Adalỳ, and C. Jutten, "Challenges in multimodal data fusion", in *2014 22nd European Signal Processing Conference (EUSIPCO)*, IEEE, 2014, pp. 101–105.

[13] A. Rodríguez, J. Valverde, J. Portilla, A. Otero, T. Riesgo, and E. De la Torre, "Fpga-based high-performance embedded systems for adaptive edge computing in cyber-physical systems: The artico3 framework", *Sensors*, vol. 18, no. 6, p. 1877, 2018.

[14] K. Sharma and M. Giannakos, "Multimodal data capabilities for learning: What can multimodal data tell us about learning?", *British Journal of Educational Technology*, vol. 51, no. 5, pp. 1450–1484, 2020.

[15] V. Vielzeuf, S. Pateux, and F. Jurie, "Temporal multimodal fusion for video emotion classification in the wild", in *Proceedings of the 19th ACM International Conference on Multimodal Interaction*, 2017, pp. 569–576.

[16] M. M. H. Shuvo, S. K. Islam, J. Cheng, and B. I. Morshed, "Efficient acceleration of deep learning inference on resource-constrained edge devices: A review", *Proceedings of the IEEE*, vol. 111, no. 1, pp. 42–91, 2022.

[17] S. Deng, Z. Xiang, J. Taheri, *et al.*, "Optimal application deployment in resource constrained distributed edges", *IEEE transactions on mobile computing*, vol. 20, no. 5, pp. 1907–1923, 2020.

[18] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, "Arcface: Additive angular margin loss for deep face recognition", in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 4690–4699.

[19] Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh, "Realtime multi-person 2d pose estimation using part affinity fields", in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 7291–7299.

[20] R. Poppe, "A survey on vision-based human action recognition", *Image and vision computing*, vol. 28, no. 6, pp. 976–990, 2010.

[21] S. Yin, Y. Wang, and Y.-H. Yang, "Attentive u-recurrent encoder-decoder network for image dehazing", *Neurocomputing*, vol. 437, pp. 143–156, 2021.

[22] Z.-Q. Zhao, P. Zheng, S.-t. Xu, and X. Wu, "Object detection with deep learning: A review", *IEEE transactions on neural networks and learning systems*, vol. 30, no. 11, pp. 3212–3232, 2019.

[23] B. Koonce and B. Koonce, "Vgg network", *Convolutional Neural Networks with Swift for Tensorflow: Image Recognition and Dataset Categorization*, pp. 35–50, 2021.

[24] M. Ferguson, R. Ak, Y.-T. T. Lee, and K. H. Law, "Automatic localization of casting defects with convolutional neural networks", in *2017 IEEE international conference on big data (big data)*, IEEE, 2017, pp. 1726–1735.

[25] W. Zaremba, I. Sutskever, and O. Vinyals, "Recurrent neural network regularization", *arXiv preprint arXiv:1409.2329*, 2014.

[26] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, `http://www.deeplearningbook.org`.

[27] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling", *arXiv preprint arXiv:1412.3555*, 2014.

[28] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3d convolutional networks", in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 4489–4497.

[29] N. Sarafianos, B. Boteanu, B. Ionescu, and I. A. Kakadiaris, "3d human pose estimation: A review of the literature and analysis of covariates", *Computer Vision and Image Understanding*, vol. 152, pp. 1–20, 2016.

[30] A. Toshev and C. Szegedy, "Deeppose: Human pose estimation via deep neural networks", in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 1653–1660.

[31] V. Lepetit, F. Moreno-Noguer, and P. Fua, "Ep n p: An accurate o (n) solution to the p n p problem", *International journal of computer vision*, vol. 81, pp. 155–166, 2009.

[32] P. F. Felzenszwalb and D. P. Huttenlocher, "Pictorial structures for object recognition", *International journal of computer vision*, vol. 61, pp. 55–79, 2005.

[33] A. Newell, K. Yang, and J. Deng, "Stacked hourglass networks for human pose estimation", in *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part VIII 14*, Springer, 2016, pp. 483–499.

[34] C. Ionescu, D. Papava, V. Olaru, and C. Sminchisescu, "Human3. 6m: Large scale datasets and predictive methods for 3d human sensing in natural environments", *IEEE transactions on pattern analysis and machine intelligence*, vol. 36, no. 7, pp. 1325–1339, 2013.

[35] L. Pishchulin, M. Andriluka, P. Gehler, and B. Schiele, "Poselet conditioned pictorial structures", in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2013, pp. 588–595.

[36] G. H. Martınez, "Openpose: Whole-body pose estimation", Ph.D. dissertation, Carnegie Mellon University Pittsburgh, PA, USA, 2019.

[37] H.-S. Fang, J. Li, H. Tang, *et al.*, "Alphapose: Whole-body regional multi-person pose estimation and tracking in real-time", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[38] J. Martinez, R. Hossain, J. Romero, and J. J. Little, "A simple yet effective baseline for 3d human pose estimation", in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2640–2649.

[39] H.-S. Fang, S. Xie, Y.-W. Tai, and C. Lu, "Rmpe: Regional multi-person pose estimation", in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2334–2343.

[40] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement", *arXiv preprint arXiv:1804.02767*, 2018.

[41] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks", *Advances in neural information processing systems*, vol. 28, 2015.

[42] M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu, "Spatial transformer networks", in *Advances in Neural Information Processing Systems (NeurIPS)*, 2015, pp. 2017–2025.

[43] B. Xiao, H. Wu, and Y. Wei, "Simple baselines for human pose estimation and tracking", in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 466–481.

[44] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition", in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[45] S.-E. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh, "Convolutional pose machines", in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2016, pp. 4724–4732.

[46] J. K. Aggarwal and M. S. Ryoo, "Human activity analysis: A review", *Acm Computing Surveys (Csur)*, vol. 43, no. 3, pp. 1–43, 2011.

[47] A. F. Bobick and J. W. Davis, "The recognition of human movement using temporal templates", *IEEE Transactions on pattern analysis and machine intelligence*, vol. 23, no. 3, pp. 257–267, 2001.

[48] L. Wang, Y. Qiao, and X. Tang, "Action recognition with trajectory-pooled deep-convolutional descriptors", in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 4305–4314.

[49] T. Lindeberg, "Scale invariant feature transform", 2012.

[50] C. Tomasi, "Histograms of oriented gradients", *Computer Vision Sampler*, pp. 1–6, 2012.

[51] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection", in *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, Ieee, vol. 1, 2005, pp. 886–893.

[52] S. Ji, W. Xu, M. Yang, and K. Yu, "3d convolutional neural networks for human action recognition", *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 1, pp. 221–231, 2012.

[53] L. Medsker and L. C. Jain, *Recurrent neural networks: design and applications*. CRC press, 1999.

[54] J. Donahue, L. Anne Hendricks, S. Guadarrama, *et al.*, "Long-term recurrent convolutional networks for visual recognition and description", in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 2625–2634.

[55] C. R. De Souza, *Action Recognition in Videos: Data-efficient approaches for supervised learning of human action classification models for video*. 2018.

[56] L. Wang, Y. Xiong, Z. Wang, *et al.*, "Temporal segment networks for action recognition in videos", *IEEE transactions on pattern analysis and machine intelligence*, vol. 41, no. 11, pp. 2740–2755, 2018.

[57] J. Lin, C. Gan, and S. Han, "Tsm: Temporal shift module for efficient video understanding", in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 7083–7093.

[58] A. G. Howard, M. Zhu, B. Chen, *et al.*, "Mobilenets: Efficient convolutional neural networks for mobile vision applications", in *arXiv preprint arXiv:1704.04861*, 2017.

[59] C. Feichtenhofer, H. Fan, J. Malik, and K. He, "Slowfast networks for video recognition", in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 6202–6211.

[60] C. Feichtenhofer, "X3d: Expanding architectures for efficient video recognition", in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 203–213.

[61] C. Wang and J. Yan, "A comprehensive survey of rgb-based and skeleton-based human action recognition", *IEEE Access*, 2023.

[62] D. Kondratyuk, L. Yuan, Y. Li, *et al.*, "Movinets: Mobile video networks for efficient video recognition", in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 16 020–16 030.

[63] A. Jain, R. Bolle, and S. Pankanti, *Introduction to biometrics*. Springer, 1996.

[64] W. W. Bledsoe, "The model method in facial recognition", *Panoramic Research Inc., Palo Alto, CA, Rep. PR1*, vol. 15, no. 47, p. 2, 1966.

[65] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, "Deepface: Closing the gap to human-level performance in face verification", in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 1701–1708.

[66] P. Theerthagiri and G. basha Nagaladinne, "Deepfake face detection using deep inceptionnet learning algorithm", in *2023 IEEE International Students' Conference on Electrical, Electronics and Computer Science (SCEECS)*, IEEE, 2023, pp. 1–6.

[67] P. Grother, P. Grother, M. Ngan, and K. Hanaoka, *Face recognition vendor test (frvt) part 2: Identification*, 2019.

[68] C. Garvie, *The perpetual line-up: Unregulated police face recognition in America*. Georgetown Law, Center on Privacy & Technology, 2016.

[69] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features", in *Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition. CVPR 2001*, Ieee, vol. 1, 2001, pp. I–I.

[70] M. Pietikäinen, "Local binary patterns", *Scholarpedia*, vol. 5, no. 3, p. 9775, 2010.

[71] M. A. Turk and A. P. Pentland, "Face recognition using eigenfaces", in *Proceedings. 1991 IEEE computer society conference on computer vision and pattern recognition*, IEEE Computer Society, 1991, pp. 586–587.

[72] P. N. Belhumeur, J. P. Hespanha, and D. J. Kriegman, "Eigenfaces vs. fisherfaces: Recognition using class specific linear projection", *IEEE Transactions on pattern analysis and machine intelligence*, vol. 19, no. 7, pp. 711–720, 1997.

[73] G. Koch, R. Zemel, R. Salakhutdinov, *et al.*, "Siamese neural networks for one-shot image recognition", in *ICML deep learning workshop*, Lille, vol. 2, 2015.

[74] G. B. Huang, M. Mattar, T. Berg, and E. Learned-Miller, "Labeled faces in the wild: A database forstudying face recognition in unconstrained environments", in *Workshop on faces in'Real-Life'Images: detection, alignment, and recognition*, 2008.

[75] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering", in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 815–823.

[76] D. G. Lowe, "Distinctive image features from scale-invariant keypoints", *International journal of computer vision*, vol. 60, pp. 91–110, 2004.

[77] C. Cortes and V. Vapnik, "Support-vector networks", *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995. DOI: 10.1007/BF00994018.

[78] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks", *Advances in neural information processing systems*, vol. 25, 2012.

[79] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation", in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580–587.

[80] W. Liu, D. Anguelov, D. Erhan, *et al.*, "Ssd: Single shot multibox detector", in *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*, Springer, 2016, pp. 21–37.

[81] D. G. Lowe, "Object recognition from local scale-invariant features", in *Proceedings of the seventh IEEE international conference on computer vision*, Ieee, vol. 2, 1999, pp. 1150–1157.

[82] R. Girshick, "Fast r-cnn", in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 1440–1448. DOI: 10.1109/ICCV.2015.169.

[83] W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges", *IEEE internet of things journal*, vol. 3, no. 5, pp. 637–646, 2016.

[84] T. Baltrušaitis, C. Ahuja, and L.-P. Morency, "Multimodal machine learning: A survey and taxonomy", *IEEE transactions on pattern analysis and machine intelligence*, vol. 41, no. 2, pp. 423–443, 2018.

[85] V. Sze, Y.-H. Chen, T.-J. Yang, and J. S. Emer, "Efficient processing of deep neural networks: A tutorial and survey", *Proceedings of the IEEE*, vol. 105, no. 12, pp. 2295–2329, 2017.

[86] D. Mourtzis, E. Vlachou, and N. Milas, "Industrial big data as a result of iot adoption in manufacturing", *Procedia cirp*, vol. 55, pp. 290–295, 2016.

[87] C. Perera, C. H. Liu, S. Jayawardena, and M. Chen, "A survey on internet of things from industrial market perspective", *IEEE Access*, vol. 2, pp. 1660–1679, 2014.

[88] M. Xu, S. Yoon, A. Fuentes, and D. S. Park, "A comprehensive survey of image augmentation techniques for deep learning", *Pattern Recognition*, vol. 137, p. 109 347, 2023.

[89] A. Mikołajczyk and M. Grochowski, "Data augmentation for improving deep learning in image classification problem", in *2018 international interdisciplinary PhD workshop (IIPhDW)*, IEEE, 2018, pp. 117–122.

[90] S. Yang, W. Xiao, M. Zhang, S. Guo, J. Zhao, and F. Shen, "Image data augmentation for deep learning: A survey", *arXiv preprint arXiv:2204.08610*, 2022.

[91] D. Tsiktsiris, D. Ziouzios, and M. Dasygenis, "A portable image processing accelerator using fpga", in *2018 7th International Conference on Modern Circuits and Systems Technologies (MOCAST)*, IEEE, 2018, pp. 1–4.

[92] R. R. Sector, "Recommendation itu-r bt. 500-15", 2023.

[93] I. Sobel, "A 3x3 isotropic gradient operator for image processing", in *Machine Vision for Three-Dimensional Scenes*, Academic Press, 2014, pp. 376–379.

[94] Knittel, "A pci-compatible fpga-coprocessor for 2d/3d image processing", in *1996 Proceedings IEEE Symposium on FPGAs for Custom Computing Machines*, IEEE, 1996, pp. 136–145.

[95] L. L. Biro, J. J. Grodstein, J.-W. Pan, and N. L. Rethman, *Static timing verification in the presence of logically false paths*, US Patent 5,648,909, Jul. 1997.

[96] B. Vaidya, M. Surti, P. Vaghasiya, J. Bordiya, and J. Jain, "Hardware acceleration of image processing algorithms using vivado high level synthesis tool", in *2017 International Conference on Intelligent Computing and Control Systems (ICICCS)*, IEEE, 2017, pp. 29–34.

[97] Q. Huynh-Thu and M. Ghanbari, "Scope of validity of psnr in image/video quality assessment", *Electronics letters*, vol. 44, no. 13, pp. 800–801, 2008.

[98] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition", *arXiv preprint arXiv:1409.1556*, 2014.

[99] D. Tsiktsiris, N. Dimitriou, A. Lalas, M. Dasygenis, K. Votis, and D. Tzovaras, "Real-time abnormal event detection for enhanced security in autonomous shuttles mobility infrastructures", *Sensors*, vol. 20, no. 17, p. 4943, 2020.

[100] Y. Lu, K. M. Kumar, S. shahabeddin Nabavi, and Y. Wang, "Future frame prediction using convolutional vrnn for anomaly detection", in *2019 16th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, IEEE, 2019, pp. 1–8.

[101] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-c. Woo, "Convolutional lstm network: A machine learning approach for precipitation nowcasting", *Advances in neural information processing systems*, vol. 28, 2015.

[102] V. Patraucean, A. Handa, and R. Cipolla, "Spatio-temporal video autoencoder with differentiable memory", *arXiv preprint arXiv:1511.06309*, 2015.

[103] M. Hasan, J. Choi, J. Neumann, A. K. Roy-Chowdhury, and L. S. Davis, "Learning temporal regularity in video sequences", in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 733–742.

[104] Z. Duan, O. Tezcan, H. Nakamura, P. Ishwar, and J. Konrad, "Rapid: Rotation-aware people detection in overhead fisheye images", in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2020, pp. 636–637.

[105] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks", in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.

[106] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "2012 alexnet", *Adv. Neural Inf. Process. Syst*, pp. 1–9, 2012.

[107] Umlaeute, *V4l2loopback*, `https://github.com/umlaeute/v4l2loopback`, Accessed 11 June 2024, 2024.

[108] T. Litman, "Autonomous vehicle implementation predictions: Implications for transport planning", 2020.

[109] D. J. Fagnant and K. Kockelman, "Preparing a nation for autonomous vehicles: Opportunities, barriers and policy recommendations", *Transportation Research Part A: Policy and Practice*, vol. 77, pp. 167–181, 2015.

[110] S. A. Bagloee, M. Tavana, M. Asadi, and T. Oliver, "Autonomous vehicles: Challenges, opportunities, and future implications for transportation policies", *Journal of modern transportation*, vol. 24, pp. 284–303, 2016.

[111] V. R. Kumar, C. Eising, C. Witt, and S. K. Yogamani, "Surround-view fisheye camera perception for automated driving: Overview, survey & challenges", *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 4, pp. 3638–3659, 2023.

[112] Y. Zhao, B. Deng, C. Shen, Y. Liu, H. Lu, and X.-S. Hua, "Spatio-temporal autoencoder for video anomaly detection", in *Proceedings of the 25th ACM international conference on Multimedia*, 2017, pp. 1933–1941.

[113] C. Zhou and R. C. Paffenroth, "Anomaly detection with robust deep autoencoders", in *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, 2017, pp. 665–674.

[114] A. Shahroudy, J. Liu, T.-T. Ng, and G. Wang, "Ntu rgb+ d: A large scale dataset for 3d human activity analysis", in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 1010–1019.

[115] N. Dimitriou, G. Kioumourtzis, A. Sideris, *et al.*, "An integrated framework for the timely detection of petty crimes", in *2017 European Intelligence and Security Informatics Conference (EISIC)*, IEEE, 2017, pp. 24–31.

[116] W. Li, V. Mahadevan, and N. Vasconcelos, "Anomaly detection and localization in crowded scenes", *IEEE transactions on pattern analysis and machine intelligence*, vol. 36, no. 1, pp. 18–32, 2013.

[117] A. Adam, E. Rivlin, I. Shimshoni, and D. Reinitz, "Robust real-time unusual event detection using multiple fixed-location monitors", *IEEE transactions on pattern analysis and machine intelligence*, vol. 30, no. 3, pp. 555–560, 2008.

[118] C. Lu, J. Shi, and J. Jia, "Abnormal event detection at 150 fps in matlab", in *Proceedings of the IEEE international conference on computer vision*, 2013, pp. 2720–2727.

[119] W. Sultani, C. Chen, and M. Shah, "Real-world anomaly detection in surveillance videos", in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 6479–6488.

[120] W. Luo, W. Liu, and S. Gao, "A revisit of sparse coding based anomaly detection in stacked rnn framework", in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 341–349.

[121] B. Wan, W. Jiang, Y. Fang, Z. Luo, and G. Ding, "Anomaly detection in video sequences: A benchmark and computational model", *IET Image Processing*, vol. 15, no. 14, pp. 3454–3465, 2021.

[122] C. X. Ling, J. Huang, and H. Zhang, "Auc: A better measure than accuracy in comparing learning algorithms", in *Advances in Artificial Intelligence: 16th Conference of the Canadian Society for Computational Studies of Intelligence, AI 2003, Halifax, Canada, June 11–13, 2003, Proceedings 16*, Springer, 2003, pp. 329–341.

[123] O. Kampman, E. J. Barezi, D. Bertero, and P. Fung, "Investigating audio, visual, and text fusion methods for end-to-end automatic personality prediction", *arXiv preprint arXiv:1805.00705*, 2018.

[124] J. D. Ortega, M. Senoussaoui, E. Granger, M. Pedersoli, P. Cardinal, and A. L. Koerich, "Multimodal fusion with deep neural networks for audio-video emotion recognition", *arXiv preprint arXiv:1907.03196*, 2019.

[125] H.-B. Zhang, Y.-X. Zhang, B. Zhong, *et al.*, "A comprehensive survey of vision-based human action recognition methods", *Sensors*, vol. 19, no. 5, pp. 1005, 2019.

[126] L. Shi, Y. Zhang, J. Cheng, and H. Lu, "Decoupled spatial-temporal attention network for skeleton-based action recognition", *arXiv preprint arXiv:2007.03263*, 2020.

[127] D. Yang, M. M. Li, H. Fu, J. Fan, and H. Leung, "Centrality graph convolutional networks for skeleton-based action recognition", *arXiv preprint arXiv:2003.03007*, 2020.

[128] S. Song, C. Lan, J. Xing, W. Zeng, and J. Liu, "Spatio-temporal attention-based lstm networks for 3d action recognition and detection", *IEEE Transactions on image processing*, vol. 27, no. 7, pp. 3459–3471, 2018.

[129] S. Yan, Y. Xiong, and D. Lin, "Spatial temporal graph convolutional networks for skeleton-based action recognition", in *Thirty-second AAAI conference on artificial intelligence*, 2018.

[130] J. Liu, A. Shahroudy, D. Xu, and G. Wang, "Spatio-temporal lstm with trust gates for 3d human action recognition", in *European conference on computer vision*, Springer, 2016, pp. 816–833.

[131] X. Yang and Y. Tian, "Super normal vector for activity recognition using depth sequences", in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 804–811.

[132] C. Cao, X. Zhang, S. Zhang, P. Wang, and Y. Zhang, "Adaptive graph convolutional networks for weakly supervised anomaly detection in videos", *IEEE Signal Processing Letters*, vol. 29, pp. 2497–2501, 2022.

[133] Y. Chen, Z. Liu, B. Zhang, W. Fok, X. Qi, and Y.-C. Wu, "Mgfn: Magnitude-contrastive glance-and-focus network for weakly-supervised video anomaly detection", in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, 2023, pp. 387–395.

[134] W. Liu, W. Luo, Z. Li, P. Zhao, S. Gao, *et al.*, "Margin learning embedded prediction for video anomaly detection with a few anomalies.", in *IJCAI*, 2019, pp. 3023–3030.

[135] J.-X. Zhong, N. Li, W. Kong, S. Liu, T. H. Li, and G. Li, "Graph convolutional label noise cleaner: Train a plug-and-play action classifier for anomaly detection", in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 1237–1246.

[136] U. Gianchandani, P. Tirupattur, and M. Shah, "Weakly-supervised spatiotemporal anomaly detection", *University of Central Florida Center for Research in Computer Vision REU*, 2019.

[137] W. Hao, R. Zhang, S. Li, *et al.*, "Anomaly event detection in security surveillance using two-stream based model", *Security and Communication Networks*, vol. 2020, 2020.

[138] D. Shreyas, S. Raksha, and B. Prasad, "Implementation of an anomalous human activity recognition system", *SN Computer Science*, vol. 1, pp. 1–10, 2020.

[139] M. Z. Zaheer, J.-h. Lee, M. Astrid, A. Mahmood, and S.-I. Lee, "Cleaning label noise with clusters for minimally supervised anomaly detection", *arXiv preprint arXiv:2104.14770*, 2021.

[140] S. Dubey, A. Boragule, J. Gwak, and M. Jeon, "Anomalous event recognition in videos based on joint learning of motion and appearance with multiple ranking measures", *Applied Sciences*, vol. 11, no. 3, p. 1344, 2021.

[141] S. Majhi, S. Das, F. Brémond, R. Dash, and P. K. Sa, "Weakly-supervised joint anomaly detection and classification", in *2021 16th IEEE International Conference on Automatic Face and Gesture Recognition (FG 2021)*, IEEE, 2021, pp. 1–7.

[142] W. Ullah, A. Ullah, I. U. Haq, K. Muhammad, M. Sajjad, and S. W. Baik, "Cnn features with bi-directional lstm for real-time anomaly detection in surveillance networks", *Multimedia tools and applications*, vol. 80, pp. 16 979–16 995, 2021.

[143] K. V. Thakare, N. Sharma, D. P. Dogra, H. Choi, and I.-J. Kim, "A multi-stream deep neural network with late fuzzy fusion for real-world anomaly detection", *Expert Systems with Applications*, vol. 201, p. 117 030, 2022.

[144] L. Prechelt, "Automatic early stopping using cross validation: Quantifying the criteria", *Neural networks*, vol. 11, no. 4, pp. 761–767, 1998.

[145] N. Strisciuglio, M. Vento, and N. Petkov, "Learning representations of sound using trainable cope feature extractors", *Pattern recognition*, vol. 92, pp. 25–36, 2019.

[146] I. Papadimitriou, A. Vafeiadis, A. Lalas, K. Votis, and D. Tzovaras, "Audio-based event detection at different snr settings using two-dimensional spectrogram magnitude representations", *Electronics*, vol. 9, no. 10, p. 1593, 2020.

[147] N. Wojke, A. Bewley, and D. Paulus, "Simple online and realtime tracking with a deep association metric", in *2017 IEEE international conference on image processing (ICIP)*, IEEE, 2017, pp. 3645–3649.

[148] S. Li, M. O. Tezcan, P. Ishwar, and J. Konrad, "Supervised people counting using an overhead fisheye camera", in *2019 16th IEEE international conference on advanced video and signal based surveillance (AVSS)*, IEEE, 2019, pp. 1–8.

[149] M. Tamura, S. Horiguchi, and T. Murakami, "Omnidirectional pedestrian detection by rotation invariant training", in *2019 IEEE winter conference on applications of computer vision (WACV)*, IEEE, 2019, pp. 1989–1998.

[150] A. Haque, A. Alahi, and L. Fei-Fei, "Recurrent attention models for depth-based person identification", in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 1229–1238.

[151] Y. Guo, L. Zhang, Y. Hu, X. He, and J. Gao, "Ms-celeb-1m: A dataset and benchmark for large-scale face recognition", in *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part III 14*, Springer, 2016, pp. 87–102.

[152] A. Anwar and A. Raychowdhury, "Masked face recognition for secure authentication", *arXiv preprint arXiv:2008.11104*, 2020.

[153] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, "Analyzing and improving the image quality of stylegan", in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 8110–8119.

[154] D. Tsiktsiris, A. Lalas, M. Dasygenis, and K. Votis, "Multimodal abnormal event detection in public transportation", *IEEE Access*, 2024, (under review).

[155] D. Tsiktsiris, A. Vafeiadis, A. Lalas, M. Dasygenis, K. Votis, and D. Tzovaras, "A novel image and audio-based artificial intelligence service for security applications in autonomous vehicles", *Transportation Research Procedia*, vol. 62, pp. 294–301, 2022.

[156] D. Tsiktsiris, A. Lalas, M. Dasygenis, and K. Votis, "Improving passenger detection with overhead fisheye imaging", *IEEE Access*, 2024.

[157] D. Tsiktsiris, A. Lalas, M. Dasygenis, K. Votis, and D. Tzovaras, "An efficient method for addressing covid-19 proximity related issues in autonomous shuttles public transportation", in *IFIP International Conference on Artificial Intelligence Applications and Innovations*, Springer, 2022, pp. 170–179.

[158] D. Tsiktsiris, A. Lalas, M. Dasygenis, K. Votis, and D. Tzovaras, "Enhanced security framework for enabling facial recognition in autonomous shuttles public transportation during covid-19", in *IFIP International Conference on Artificial Intelligence Applications and Innovations*, Springer, 2021, pp. 145–154.

[159] D. Tsiktsiris, D. Ziouzios, and M. Dasygenis, "A high-level synthesis implementation and evaluation of an image processing accelerator", *Technologies*, vol. 7, no. 1, p. 4, 2018.

[160] D. Tsiktsiris, D. Ziouzios, and M. Dasygenis, "HLS accelerated noise reduction approach using image stacking on xilinx PYNQ", in *2019 8th International Conference on Modern Circuits and Systems Technologies (MOCAST)*, IEEE, 2019, pp. 1–4.

Appendix A

Publications

During the doctoral thesis, the following works were published in Journals, Conferences and Book Chapters, respectively.

A.1 Journals

[**J1**] **D. Tsiktsiris**, A. Lalas, M. Dasygenis and K. Votis, "Enhancing the Safety of Autonomous Vehicles: Semi-Supervised Anomaly Detection with Overhead Fisheye Perspective", IEEE Access, 2024 (Q1).

[**J2**] **D. Tsiktsiris**, A. Lalas, M. Dasygenis and K. Votis, "Improving Passenger Detection with Overhead Fisheye Imaging", IEEE Access, 2024 (Q1).

[**J3**] T. Sanida, **D. Tsiktsiris**, A. Sideris, and M. Dasygenis. "A heterogeneous implementation for plant disease identification using deep learning". In Multimedia Tools and Applications, vol. 81, p. 15041–15059, 2022 (Q1).

[**J4**] T. Sanida, A. Sideris, **D. Tsiktsiris**, and M. Dasygenis. "Lightweight neural network for COVID-19 detection from chest X-ray images implemented on an embedded system". In Technologies, vol. 10, p. 37, 2022 (Q1).

[**J5**] **D. Tsiktsiris**, N. Dimitriou, A. Lalas, M. Dasygenis, K. Votis, and D. Tzovaras. "Real-time abnormal event detection for enhanced security in autonomous shuttles mobility infrastructures". In Sensors, vol. 20, p. 4943, 2020 (Q1).

[**J6**] D. Ziouzios, **D. Tsiktsiris**, N. Baras, and M. Dasygenis. "A distributed architecture for smart recycling using machine learning". In Future Internet,

vol. 12, p. 141, 2020 (Q2).

[J7] **D. Tsiktsiris**, D. Ziouzios, and M. Dasygenis. "A High-Level synthesis implementation and evaluation of an image processing accelerator". In Technologies, vol. 7, p. 4, 2018 (Q1).

A.2 Conferences

[C1] **D. Tsiktsiris**, N. Dimitriou, Z. Kolias, S. Skourti, P. Girssas, A. Lalas, K. Votis, and D. Tzovaras. "A Framework for Contextual Recommendations Using Instance Segmentation". In International Conference on Human-Computer Interaction, July, 2023, pp. 395-408, Cham: Springer Nature Switzerland.

[C2] **D. Tsiktsiris**, A. Vafeiadis, A. Lalas, M. Dasygenis, K. Votis, and D. Tzovaras. "A novel image and audio-based artificial intelligence service for security applications in autonomous vehicles". In Transportation Research Procedia, vol. 62, p. 294–301, 2022.

[C3] **D. Tsiktsiris**, A. Lalas, M. Dasygenis, K. Votis, and D. Tzovaras. "An Efficient Method for Addressing COVID-19 Proximity Related Issues in Autonomous Shuttles Public Transportation". In Artificial Intelligence Applications and Innovations: 18th IFIP WG 12.5 International Conference, AIAI 2022, Hersonissos, Crete, Greece, June 17–20, 2022, Proceedings, Part I, 2022.

[C4] A. Sideris, T. Sanida, **D. Tsiktsiris**, and M. Dasygenis. "Image Hashing Based on SHA-3 Implemented on FPGA". In Recent Advances in Manufacturing Modelling and Optimization: Select Proceedings of RAM 2021, Springer, 2022, p. 521–530.

[C5] **D. Tsiktsiris**, T. Sanida, A. Sideris, and M. Dasygenis. "Accelerated Defective Product Inspection on the Edge Using Deep Learning". In Recent Advances in Manufacturing Modelling and Optimization: Select Proceedings of RAM 2021, Springer, 2022, p. 185–191.

[C6] T. Sanida, **D. Tsiktsiris**, A. Sideris, and M. Dasygenis. "A Heterogeneous Lightweight Network for Plant Disease Classification". In 2021 10th

International Conference on Modern Circuits and Systems Technologies (MOCAST), 2021.

[C7] **D. Tsiktsiris**, A. Lalas, M. Dasygenis. K. Votis, and D. Tzovaras. "Enhanced Security Framework for Enabling Facial Recognition in Autonomous Shuttles Public Transportation During COVID-19". In Artificial Intelligence Applications and Innovations: 17th IFIP WG 12.5 International Conference, AIAI 2021, Hersonissos, Crete, Greece, June 25–27, 2021, Proceedings, 2021.

[C8] D. Ziouzios, **D. Tsiktsiris**, N. Baras, S. Bibi, and M. Dasygenis. "A generator tool for Carry Look-ahead Adders (CLA)". In SHS Web of Conferences, 2021.

[C9] **D. Tsiktsiris**, K. Kechagias, M. Dasygenis, and P. Angelidis. "Accelerated seven segment optical character recognition algorithm". In 2019 Panhellenic Conference on Electronics & Telecommunications (PACET), 2019.

[C10] **D. Tsiktsiris**, D. Ziouzios, and M. Dasygenis. "HLS Accelerated Noise Reduction Approach Using Image Stacking on Xilinx PYNQ". In 2019 8th International Conference on Modern Circuits and Systems Technologies (MOCAST), 2019.

[C11] **D. Tsiktsiris**, D. Ziouzios, and M. Dasygenis. "A portable image processing accelerator using FPGA". In 2018 7th International Conference on Modern Circuits and Systems Technologies (MOCAST), 2018.

[C12] A. Sideris, T. Sanida, **D. Tsiktsiris**, and M. Dasygenis. "Acceleration of Image Processing with SHA-3 (Keccak) Algorithm using FPGA".

A.3 Book Chapters

[BC1] A. Sideris, **D. Tsiktsiris**, D. Ziouzios, and M. Dasygenis. "Smart grid hardware security". In IoT for Smart Grids: Design Challenges and Paradigms, p. 85–113, 2019.

[BC2] D. Ziouzios, A. Sideris, **D. Tsiktsiris**, and M. Dasygenis. "Smart-Grid Modelling and Simulation". In IoT for Smart Grids: Design Challenges and Paradigms, p. 43–54, 2019.