Center of Integrative Physiology and Molecular Medicine

Department of Physiology

Faculty of Medicine

University of Saarland, Homburg, Germany

Fully Automated Vesicle Exocytosis Detection, Tracking and Analysis Using Artificial Intelligence

Dissertation zur Erlangung des Grades eines Doktors der Naturwissenschaften

der Medizinischen Fakultät

der UNIVERSITÄT DES SAARLANDES

2025

Vorgelegt von: Abed Alrahman H. Chouaib

Riha – Libanon

Tag der Promotion: 18.05.2026

Dekan:

Univ.-Prof. Dr. med. dent. M. Hannig

Berichterstatter:

Prof. Dr. Ute Becherer

Prof. Dr. Olga Kalinina

Anlage I

Erklärung gemäß § 7 Abs. 1 Nr. 2

Ich erkläre hiermit an Eides statt, dass ich die vorliegende Dissertation mit dem Titel „Fully Automated Granule Exocytosis Detection, Tracking and Analysis Using Artificial Intelligence“ eigenständig und ohne unzulässige Hilfe Dritter angefertigt habe.

Alle verwendeten Quellen und Hilfsmittel sind im Text sowie im Literaturverzeichnis vollständig angegeben.

Die gesamte Entwicklung der intelligent vesicle exocytosis analysis platform (IVEA & IVEA-Py), einschließlich Konzeption, Programmierung, Datenanalyse und Auswertung, wurde von mir selbst durchgeführt.

Ein Teil der Datensätze, die zur Validierung und Demonstration der Software verwendet wurden, stammt von verschiedenen Forscherinnen und Forschern, die ihre Daten unentgeltlich zur Verfügung gestellt haben.

Diese Personen waren ausschließlich an der Bereitstellung experimenteller Daten beteiligt und haben keine finanzielle Gegenleistung erhalten.

Dazu gehören unter anderem:

Hsin-Fang Chang, Santiago Echeverry, Nadia Alawar, Omnia M. Khamis, Lucie Demeersseman, Sofia Elizarova, Qinghai Tian.

Die Daten aus bereits veröffentlichten wissenschaftlichen Arbeiten sind als Quellen im Literaturverzeichnis der Dissertation angegeben; die jeweiligen Autorinnen und Autoren sind daher hier nicht erneut aufgeführt.

Weitere Personen waren an der inhaltlichen oder materiellen Erstellung der Arbeit nicht beteiligt. Insbesondere habe ich keine entgeltliche Hilfe von Vermittlungs- oder Beratungsdiensten in Anspruch genommen.

Die Dissertation wurde bisher weder im In- noch im Ausland in gleicher oder ähnlicher Form einer anderen Prüfungsbehörde vorgelegt.

Ich versichere an Eides statt, dass ich nach bestem Wissen die Wahrheit gesagt und nichts verschwiegen habe.

Die Bedeutung der eidesstattlichen Erklärung und die strafrechtlichen Folgen einer unrichtigen oder unvollständigen eidesstattlichen Erklärung sind mir bekannt.

Homburg, den 

Contents

List of Abbreviations

Abbreviation	Full Term	Abbreviation	Full Term
AI	Artificial Intelligence	JSON	JavaScript Object Notation
API	Application Programming Interface	LM	Local Maximum
CCD	Charge-Coupled Device	LoG	Laplacian of Gaussian
CNN	Convolutional Neural Network	LSTM	Long Short-Term Memory
CNN-LSTM	Convolutional + LSTM Hybrid	MAD	Median Absolute Deviation
CMOS	Complementary Metal-Oxide Semiconductor	mGFP	Monomeric Green Fluorescent Protein
CPU	Central Processing Unit	MIC	Multilayer Intensity Correction
CSV	Comma-Separated Values	MINFLUX	Minimal Photon Fluxes Localization Microscopy
CTL	Cytotoxic T Lymphocyte	MLP	Multilayer Perceptron
DART	Dopamine Recognition Tool	mNeonGreen	Monomeric NeonGreen Fluorescent Protein
DBSCAN	Density-Based Spatial Clustering of Applications with Noise	NMS	Non-Maximum Suppression
DoG	Difference of Gaussian	NPY	Neuropeptide Y
DRG	Dorsal Root Ganglion	PSF	Point Spread Function
EDT	Euclidean Distance Transform	PyTorch	Python Deep Learning Framework
EDM	Euclidean Distance Map	ReLU	Rectified Linear Unit
eViT	Encoder-Vision Transformer	RNN	Recurrent Neural Network
FWHM	Full Width at Half Maximum	ROI	Region of Interest
GeLU	Gaussian Error Linear Unit	ROI-Manager	ImageJ ROI Manager

GNMS	Gaussian Non-Maximum Suppression	ROI-Zip	ROI Archive File
GPU	Graphics Processing Unit	SAM	Segment Anything Model
GVF	Gradient Vector Flow	SIFT	Scale-Invariant Feature Transform
GUI	Graphical User Interface	SNR	Signal-to-Noise Ratio
HDF5	Hierarchical Data Format 5	STED	Stimulated Emission Depletion Microscopy
HE	Human Expert	SypHy	Synaptophysin-pHluorin Reporter
INS	Insulinoma-1	TIRF	Total Internal Reflection Fluorescence Microscopy
INT16 / FP32 / UINT8	Integer 16-bit / Float 32-bit / Unsigned 8-bit	TensorFlow	Deep Learning Framework
IoU	Intersection over Union	ViT	Vision Transformer
IVEA	Intelligent Vesicle Exocytosis Analysis	WBF	Weighted Boxes Fusion

Zusammenfassung

Die quantitative Analyse der Vesikel- und Granula-Exozytose stellt in der Lebendzellmikroskopie weiterhin eine Herausforderung dar, da die Freisetzungsereignisse transient und heterogen sind. Künstliche Intelligenz (KI) bietet inzwischen Methoden, die in der Lage sind, diese Komplexität durch adaptive und datengetriebene Erkennung zellulärer Dynamiken zu bewältigen. Zur Überwindung bestehender analytischer Einschränkungen wurde das quelloffene und plattformübergreifende Framework Intelligent Vesicle Exocytosis Analysis (IVEA) (Chouaib et al., 2025) entwickelt, das eine vollständig automatisierte und hochdurchsatzfähige Detektion und Klassifikation von Exozytoseereignissen in Fluoreszenzmikroskopie-Aufnahmen ermöglicht.

IVEA kombiniert klassische Computer-Vision-Methoden mit Deep Learning in drei komplementären Analysepfaden: einem Vision-Transformer-Encoder zur Erkennung stochastischer Burst-Ereignisse, einem multivariaten Long Short-Term Memory (LSTM)-Modell zur Analyse stationärer Freisetzungsaktivität sowie einem Hotspot-Detektionsmodul für Nanosensor-Assays wie AndromeDA (Elizarova et al., 2022). Ein Gaußsches Non-Maximum-Suppression-Verfahren, das direkt in kontinuierlichen raumzeitlichen Koordinaten (x, y, t) arbeitet, fusioniert redundante Detektionen bei diffusen und kurzlebigen Ereignissen. Das System erlernt Analyseparameter automatisch aus den Anfangsframes jeder Aufnahme, kompensiert Photobleaching dynamisch und wendet bei Bedarf ein temporales Max-Pooling an, um relevante zeitliche Informationen langsamer Signale zu bewahren und gleichzeitig die Rechenlast zu reduzieren.

Die Python-Implementierung IVEA-Py erweitert das Framework um eine Tracking- und Klassifikationspipeline, die Vesikeltrajektorien rekonstruiert. Diese basiert auf Difference-of-Gaussian-Vorverarbeitung, Integration von Gradientenvektorfeldern, Euler-Integration, DBSCAN-Clustering und Kalman-Filterung. Dadurch wird eine zuverlässige Verknüpfung der Vesikelbewegung mit Fusionsereignissen ermöglicht und eine großangelegte, reproduzierbare Analyse des Vesikelverhaltens vor der Freisetzung unterstützt.

IVEA wurde anhand von Datensätzen aus Neuronen, zytotoxischen T-Lymphozyten, Chromaffinzellen und β-Zellen mit unterschiedlichen Fluoreszenzreportern validiert. IVEA Framework erreichte eine hohe Detektionsgenauigkeit, starke Generalisierbarkeit über verschiedene Bildgebungsmodalitäten hinweg und eine bis zu sechzigfache Beschleunigung im Vergleich zur manuellen Auswertung. Benchmarking mit etablierten Tools wie ComDet, pHusion, SynActJ und ExoJ zeigte IVEA eine überlegene Sensitivität, Spezifität und Reproduzierbarkeit. Durch die Integration adaptiver Parametrisierung, tiefenzeitlicher Modellierung und Ereignis-Trajektorien-Verknüpfung demonstriert IVEA, wie KI-basierte Bildanalyse die quantitative Untersuchung schneller interzellulärer Kommunikation verbessern und ein allgemeines Framework für die automatisierte Erkennung biologischer Ereignisse in der Lebendzellmikroskopie etablieren kann.

Abstract

The quantitative analysis of vesicle and granule exocytosis remains a challenge in live-cell imaging due to the transient and heterogeneous nature of release events. Artificial intelligence (AI) now provides methods capable of addressing this complexity through adaptive, data-driven recognition of cellular dynamics. To overcome current analytical limitations, an open-source and cross-platform framework, the Intelligent Vesicle Exocytosis Analysis (IVEA) (Chouaib et al., 2025) was developed to enable fully automated and high-throughput detection and classification of exocytosis in fluorescence microscopy recordings.

IVEA combines classical computer vision and deep learning across three complementary pathways: a Vision Transformer encoder for stochastic burst events, a multivariate Long Short-Term Memory (LSTM) model for stationary release activity, and a hotspot detection module for nanosensor assays such as AndromeDA (Elizarova et al., 2022). A Gaussian non-maximum suppression algorithm operating directly in continuous spatiotemporal coordinates (x, y, t) merges redundant detections in diffuse and short-lived events. The system learns analysis parameters automatically from the initial frames of each recording, compensates for photobleaching dynamically, and applies temporal max-pooling on demand to preserve salient temporal information for slow signals while reducing computational load.

The Python implementation, IVEA-Py, extends the framework with a tracking and classification pipeline that reconstructs vesicle trajectories through Difference-of-Gaussian preprocessing, gradient vector field integration, Euler integration, DBSCAN clustering, and Kalman filtering. This enables reliable linking of vesicle motion to fusion events and supports large-scale, reproducible analysis of vesicle behavior preceding release.

IVEA was evaluated across recordings from neurons, cytotoxic T lymphocytes, chromaffin cells, and β-cells labeled with diverse reporters. The IVEA framework achieved high detection accuracy, broad generalization across imaging modalities, and up to sixty-fold faster performance compared to manual review. Benchmarking against established tools such as ComDet, pHusion, SynActJ, and ExoJ demonstrated consistently superior sensitivity, specificity, and reproducibility. By integrating adaptive parameterization, deep temporal modeling, and trajectory-event linking, IVEA demonstrates how AI-based image analysis can enhance quantitative studies of rapid intercellular communication and establish a general framework for automated biological event detection in live-cell microscopy.

1 Introduction

1.1 Artificial intelligence

The field of artificial intelligence comprises a broad collection of computational approaches aimed at enabling machines to perform tasks that generally rely on human perception, reasoning, or decision-making. Over the past few decades, AI transitioned from being a theoretical concept in literature and stories in science fiction movies to a transformative tool that is revolutionizing human society. The progress of AI has been driven by algorithmic innovations, advancements in electronic chips that increased computational power, and the availability of large datasets across virtually every domain (Meyes et al., 2019).

In the domain of AI, machine learning is denoted as the core subfield, which is further categorized into four diverse categories (**Figure 1**). First, the supervised learning method, in which algorithms are trained using labeled data. Secondly, the unsupervised learning, where patterns are extracted from unlabeled datasets. Thirdly, reinforcement learning, where models iteratively improve through feedback from their output and environment. The last one that revolutionized the field of AI is deep learning. This subfield has widened the field by introducing multi-layered neural networks capable of learning complex, hierarchical representations from raw data. Developments in deep learning were closely tied to advances in modern GPUs and specialized processors that could train large models due to their parallel architecture.

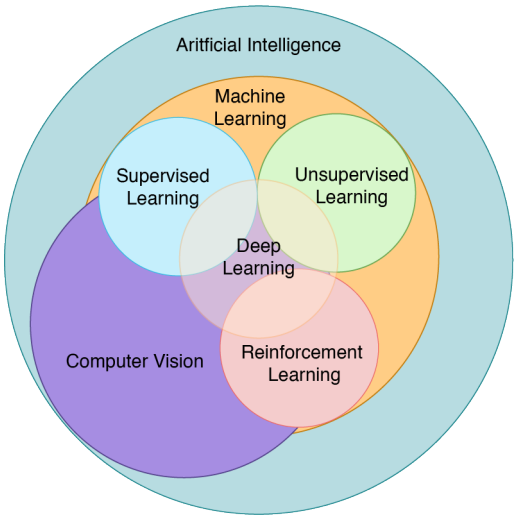


Figure 1. Venn Diagram of Learning Paradigms in Artificial Intelligence.
The outer blue circle represents the wide field of artificial intelligence. Within machine learning (orange), supervised (light blue), unsupervised (light green), and reinforcement learning (light red) are key approaches. Deep learning (transparent orange) overlaps these paradigms as a neural network-based method. Computer vision (purple) is shown as an application area that draws from multiple AI techniques. The color coding highlights the theoretical boundaries and overlaps between domains and applications.

1.2 Image processing and computer vision in microscopy

In biological and medical research, deep learning now serves as a universal framework for pattern recognition, detecting cells (Pachitariu & Stringer, 2022), classifying tissues (Archit et al., 2025), reconstructing 3D structures (Mildenhall et al., 2020a; Shaib et al., 2025), predicting protein sequence and molecular interactions (Jumper et al., 2021; Lisanza et al., 2025), and classifying biological activities (Chouaib et al., 2025). These methods have become extremely useful, allowing analysis with high accuracy and exposing subtle quantitative relationships that often remain hidden to human observers.

Image processing and computer vision are two terms that are often used interchangeably, yet they address distinct but complementary aspects of visual data analysis. Image processing focuses on pixel-level manipulation, such as filtering, denoising, correcting microscopy artifacts and laser aberrations, enhancing contrast, and performing segmentation. All these are used to improve image quality and extract measurable signals. In contrast, computer vision seeks to interpret these signals at a higher level, performing tasks such as object detection, tracking, and event recognition.

In microscopy, both fields converge to translate photon-based measurements into biologically meaningful information. As technology advanced, photon detectors and imaging techniques became more sophisticated and accurate, thereby enhancing the resolution of microscopy images (Balzarotti et al., 2017; Hensel et al., 2025). This facilitated the researchers' ability to record dynamic cellular events with nanometer precision. However, the resulting datasets often exceed terabytes, rendering manual inspection impractical. Integrating AI with computer vision provides the scalability and consistency necessary for modern microscopy, thereby enabling objective, automated analyses of extensive image collections.

The integration of AI, particularly deep learning, into computer vision has significantly advanced its capabilities. Adding models such as the convolutional neural networks (CNNs) (Ronneberger et al., 2015), recurrent neural networks (RNNs) (Rumelhart et al., 1986), and more recently, transformer-based architectures (Dosovitskiy et al., 2021; Vaswani et al., 2017) have become the cornerstones of this progress, excelling in spatial, temporal, and spatiotemporal pattern recognition.

1.3 Software ecosystem and programming environment

Developing tools for biological analysis needs programming environments that are easy to use, flexible, and fast. In this work, three major platforms were used, including Java, Python, and MATLAB. Together, these platforms cover different needs. Java offers stable tools and broad access for users. Python provides a flexible space for fast AI development. In contrast, MATLAB provides a comfortable environment for testing ideas with close control over each step.

Java is an object-oriented language that forms the backbone of the ImageJ/Fiji environment (Rueden et al., 2017; Schneider et al., 2012). ImageJ is an open-source image processing software used for scientific applications. The recent distribution of ImageJ as of the present year (2025) is ImageJ2, also known as Fiji. This distribution can be viewed as a “batteries included” version of ImageJ, as it contains a preset of plugins that are ready for use. ImageJ’s structure and object-oriented design, cross-platform portability, user-friendly GUI, and community support make it ideal for building stable and easy-to-install plugins. Java and the ImageJ libraries were used here to create the main biological analysis modules.

In contrast, Python serves as a flexible interface for advanced AI integration. Its extensive ecosystem, including TensorFlow, PyTorch, NumPy, scikit-image, OpenCV, and other packages, makes it the dominant language for deep-learning research and data analysis. Python’s readability and interoperability allow the same code to run on consumer laptops or GPU clusters. These properties ensure scalability for both small-scale experiments and large-volume microscopy datasets. In this project, the Python language was used to design and train deep neural networks, perform statistical evaluations, and run a more sophisticated version of my software.

Finally, MATLAB supports rapid testing and visual checks. Its matrix-based style and built-in plots make it useful during early development when new ideas need quick feedback. The high-level functions and graphical capabilities of MATLAB provide invaluable support during the early stages of algorithm testing and performance validation. However, MATLAB's proprietary nature and licensing costs limit its usability and adoption in open science.

Although Java, Python, and MATLAB represent the primary environments for developing scientific tools, other languages are increasingly entering the field. For example, C++ remains the best option in terms of execution speed and low-level control, making it indispensable for performance-critical applications such as image rendering, GPU computation, or hardware interfacing. But it isn't very easy and takes more effort to build and maintain. Similarly, emerging languages such as Rust are gaining attention for their combination of safety, modern syntax, and near-C++ performance. Rust is gaining interest for its safety and near-C++ speed. It could become more common in future scientific tools, but its smaller community and library base still limit wide use.

1.4 Exocytosis as a core mechanism of cellular communication

Exocytosis represents a central and evolutionarily conserved pathway for intercellular communication. Through this process, eukaryotic cells deliver a wide range of molecular cargos such as neurotransmitters, hormones, peptides, and immune effectors to the extracellular environment. Exocytosis enables rapid spatially confined signal transmission, membrane renewal, and dynamic remodeling of the cell surface through membrane fusion.

Generally, two operational modes are distinguished, namely constitutive and regulated exocytosis. Constitutive exocytosis occurs continuously, ensuring steady secretion of essential molecules such as extracellular matrix proteins and lipids. This process maintains plasma-membrane composition and cell health. In contrast, regulated exocytosis is activated only in response to specific stimuli. Typically, a rise in cytosolic calcium or receptor-mediated signaling provides precise temporal control over secretion. This regulated form is the foundation of rapid communication in neurons, hormone release from endocrine cells, and immune effector responses by cytotoxic lymphocytes.

These systems may be distinct in their biological function, but the general logic of regulated exocytosis is maintained. Vesicles originating from the Golgi or endosomal compartments are transported toward the cell periphery. These vesicles get tethered at release sites for regulated exocytosis and fused to the plasma membrane in a Ca^{2+} dependent manner. Neuronal synaptic vesicles, insulin granules of pancreatic β-cells, and the lytic granules of cytotoxic T lymphocytes (CTLs) all share the same principle. Each of these cells adapts similar molecular components to serve different physiological roles. This includes speed and precision in neurons, modulation of metabolism in endocrine cells, and precision targeting of toxicity in the immune system.

1.5 Molecular mechanisms underlying regulated exocytosis

Throughout this work, the term vesicle refers to small synaptic vesicles in neurons, whereas the larger secretory vesicles in endocrine and immune cells are referred to as granules. Both are collectively referred to as vesicular compartments when describing shared mechanisms.

At the molecular level, exocytosis proceeds through a sequence of highly coordinated stages: vesicle trafficking, tethering, docking and priming, membrane fusion, and subsequent endocytic recycling (**Figure 2**). Each step is structured by a set of conserved protein complexes that physically and temporally couple vesicle motion to signaling events at the plasma membrane (Chang et al., 2023).

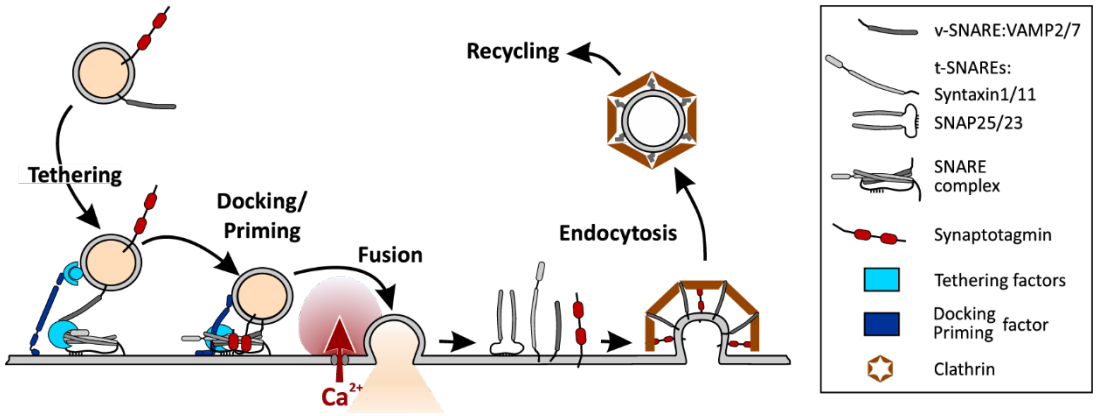


Figure 2. Model of the exocytosis machinery (adapted from Chang et al. (2023)).
This figure demonstrates the shared molecular basis of neuronal synaptic vesicle and cytotoxic granule exocytosis. The mechanism is displayed in its various stages, including tethering, docking/priming, Ca^{2+}-triggered fusion, and subsequent endocytosis and recycling. The same principle underlies secretion in pancreatic β-cells, where insulin

granules use homologous SNARE isoforms and calcium-sensing modules to achieve regulated release in response to glucose.

The key catalytic engine of membrane fusion is the SNARE complex, a group of soluble N-ethylmaleimide-sensitive factor attachment protein receptors. Vesicular SNARE (v-SNARE; typically VAMP2 or VAMP7) pairs with two target-membrane SNAREs (t-SNAREs; syntaxin 1/11 and SNAP-25/23) to form a parallel four-helix bundle that pulls the opposing membranes into proximity. This progressive "zippering" releases sufficient energy to overcome the hydration barrier and drive bilayer fusion (Chang et al., 2023).

SNARE function is regulated by a collection of assistant proteins that ensure the constancy and timing of secretion. Munc18 binds to syntaxin and organizes the assembly of SNARE, while Munc13 converts syntaxin conformation from closed to open, enabling priming to occur. Complexin stabilizes the partially assembled SNARE bundle to prevent premature fusion, and synaptotagmin, the principal Ca^{2+} sensor on secretory vesicles, triggers the final fusion event when intracellular Ca^{2+} rises. Additional modulator proteins exist, such as Rab GTPases, RIM, CAPS, and DOC2. These modulators coordinate vesicle positioning and coupling to calcium channels, which link biochemical readiness with spatial precision at the release site.

Once fusion occurs, the ATPase NSF disassembles the SNARE complex, thus enabling the components to be recycled and maintaining the balance between exocytosis and endocytosis. This recycling step keeps exocytosis and endocytosis balanced, allowing neurons to signal rapidly and enabling endocrine and immune cells to sustain repeated secretion.

Understanding this universal machinery provides the foundation for exploring how different cell types tune the dynamics of secretion. These conserved mechanisms form the molecular basis for neuronal communication, endocrine regulation, and immune cytotoxicity. This connection establishes the foundation for the subsequent sections, which investigate the specific adaptation of SNARE proteins in each of these distinct biological systems.

1.6 Cellular models of regulated exocytosis

Regulated exocytosis follows a conserved sequence of molecular events, but the dynamics and organization of exocytosis are quite variable, given the spectrum of specialized cell types. Neurons, endocrine cells, and CTLs provide three representative models that illustrate how a universal mechanism is adapted to distinct physiological needs.

In neurons, exocytosis occurs at synapses with remarkable speed and precision. Synaptic vesicles are clustered at an active zone on the presynaptic membrane. These vesicles undergo cycles of tethering, priming, and fusion, releasing neurotransmitters within milliseconds. This rapid release is triggered by voltage-activated calcium entry, which follows an action potential. The high temporal resolution of vesicle fusion is made possible by the tight coupling of calcium sensors, such as synaptotagmin, to the

SNARE complex. These components are essential for synchronous vesicle fusion. The spatial arrangement of these molecular players, organized at the nanometer scale in microdomains, ensures both reliability and speed in synaptic transmission. Because of this, studies on neuronal exocytosis have provided a conceptual framework for understanding vesicle fusion in different cell types.

Endocrine and neuroendocrine cells, such as pancreatic β-cells and chromaffin cells, represent a slower but equally precise variant of regulated secretion. In this case, larger, dense-core vesicles containing hormones or peptides, such as insulin, glucagon, or catecholamines, also undergo regulated exocytosis. Intracellular calcium elevation following metabolic or electrical stimulation triggers their exocytosis. Compared to synaptic vesicles, dense-core granules are substantially larger (60–300 nm in diameter) and less densely packed near the plasma membrane. Unlike synaptic vesicles, which fuse within milliseconds, lytic granule exocytosis in immune cells shows much slower and more variable timing, ranging from rapid sub-second events to delays of many minutes. For example, insulin secretion by β-cells is essential for maintaining glucose balance in the body. When this process doesn't work properly, it can lead to metabolic diseases like diabetes. Despite their slower kinetics, the same SNARE-based machinery and calcium-dependent triggering principles govern granule release in endocrine systems.

A third model of regulated exocytosis is found in CTLs. These cells execute highly specialized exocytosis at the immunological synapse, a structured contact area created between the CTL and its target cell. Upon recognizing an antigen via the T cell receptor, lytic granules are transported along microtubules toward the synapse. These granules must also tether, dock, and undergo calcium-dependent fusion to release perforin and granzymes. This form of targeted secretion enables CTLs to induce apoptosis specifically in infected or malignant cells without damaging surrounding tissue. Even the molecular apparatus in this type of secretion can be compared to that of neuronal synapses: v-SNAREs like VAMP7 or VAMP8; t-SNAREs like syntaxin-11 or SNAP-23; and regulatory factors like Munc13-4, Munc18-2, and synaptotagmin-7 all work in similar ways. Additionally, mutation of these proteins results in severe immunodeficiencies, or familial hemophagocytic lymphohistiocytosis (FHL), pointing to their essential role in immune function.

Secretory vesicles vary substantially in density and size across cell types. Neuronal synapses contain over 100 vesicles per μm², β-cells around 18, and CTLs approximately 5–7 granules per μm², each 60–300 nm in diameter. These differences in vesicle crowding and mobility strongly influence the detection of exocytosis using fluorescent live cell imaging. This generates a variety of fluorescence profiles, necessitating separate detection modules.

1.7 Random burst events module

In most secretory cells, vesicles exhibit dynamic movement before exocytosis, traveling along cytoskeletal tracks to reach the plasma membrane (**Figure 3a**). These vesicles fuse at variable locations at the cell surface rather than at fixed pre-defined sites. Each fusion event is typically independent,

spatially dispersed, and transient. Such activity patterns are characteristic of CTLs, chromaffin cells, and endocrine cells such as pancreatic β-cells, where individual vesicles undergo stochastic fusion events in response to a trigger. These events are termed random bursts because they occur at apparently random positions and times relative to each other.

In TIRFM recordings, random burst events can manifest as localized spikes in fluorescence intensity that appear and vanish within a few frames. When vesicles are labeled with pH-sensitive fluorophores, such as pHuji or pHluorin, fusion produces a rapid fluorescence increase followed by decay (**Figure 3c**). This is because the fluorophore is first unquenched as it goes from an acidic compartment (the granule) to a slightly basic compartment (the extracellular medium) and then diffuses away either in the extracellular medium or into the plasma membrane as the vesicle membrane merges with the plasma membrane (**Figure 3c**). In contrast, pH-insensitive labels or dyes reveal these events as sudden disappearances, since the fluorescent cargo dissipates into the extracellular space (**Figure 3b**). Both cases generate sharp temporal changes that serve as visual indicators of fusion but vary substantially depending on labeling strategy, fluorophore sensitivity, and imaging conditions. The dynamic nature of these vesicles presents significant challenges for automated analysis. Their motion across frames complicates spatial alignment, while differences in intensity profiles between pH-sensitive and pH-insensitive probes hinder classical rule-based detection.

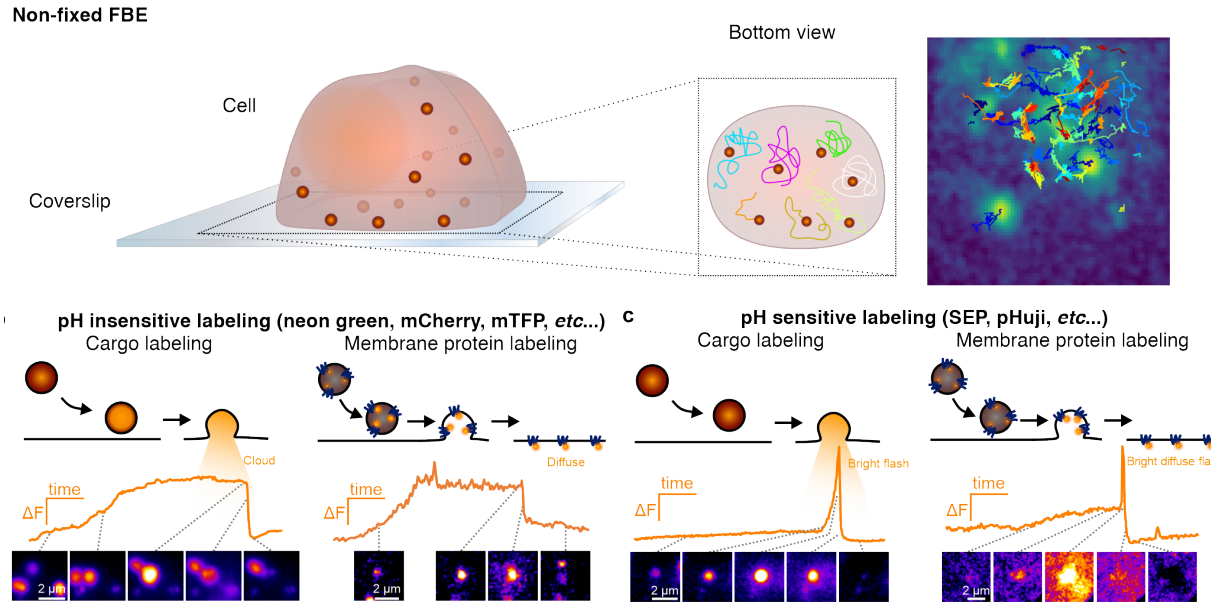


Figure 3. Granule exocytosis characteristics using different labeling techniques (adapted from (Chouaib et al., 2025)).

a. Illustration of a cytotoxic T-cell showing the overall distribution of fluorescently labeled secretory (lytic) granules. These granules occupy the cell interior (left) and display continuous intracellular motion, indicated in the magnified region or in the example TIRF recording with real-time tracks overlaid (right). **b-c**. Top row schema: simplified diagrams summarizing the steps that lead to granule fusion in CTLs. Middle row schema: fluorescence time-courses for individual granules, aligned with the image snapshots shown beneath each graph. The stippled lines on the intensity graphs above highlight the time points of the snapshots. **b.** Examples of granule exocytosis using pH-insensitive fluorescent protein labels. These proteins are bound either to a cargo protein (e.g., granzyme

B, left) or to a membrane protein (e.g., synaptobrevin2, right). **c**. Examples of granule exocytosis using pH-sensitive fluorescent protein labels, including variants like super-ecliptic pHluorin (SEP). These labels are similarly bound either to a cargo protein (e.g., granzyme B, left) or to a membrane protein (e.g., synaptobrevin2, right). Notably, upon fusion, the vesicle lumen becomes neutralized, producing a sharp and rapid rise in fluorescence..

Fluorescent labeling plays a crucial role in visualizing vesicle fusion and interpreting fluorescence dynamics. In this work, the pH-sensitive and pH-insensitive probes were utilized. pH-sensitive fluorophores including pHuji and super ecliptic pHluorin (SEP) were fused to label vesicular membrane proteins (e.g., synaptobrevin2 or synaptophysin) or luminal cargoes (e.g., granzyme B), allowing clear and easy visualization of fusion events due to the rapid fluorescence spikes occurring upon exocytosis. In parallel, pH-insensitive fluorophores such as mCherry and mNeonGreen were employed for stable tracking of granule membranes or cargo proteins, providing continuous localization independent of vesicular pH. Endogenous knock-in labeling of target proteins further minimized overexpression artifacts and maintained native trafficking kinetics. These complementary labeling strategies enabled IVEA to handle both fluorescence appearance and disappearance patterns during event detection.

1.8 Stationary burst events module

In contrast to random exocytosis, stationary burst events occur at fixed membrane sites, where vesicle fusion is repeated or sustained over time. This behavior is typical of neuronal synapses, where multiple vesicles fuse sequentially at specialized release zones, producing recurrent or prolonged bursts of fluorescence at the same spatial coordinates (**Figure 4a**). During evoked or spontaneous activity, synaptic vesicles fuse and recycle within milliseconds, resulting in fluorescence changes confined to static puncta (**Figure 4b**). These events thus represent high-frequency, spatially localized bursts of exocytosis with characteristic temporal profiles.

From an analytical perspective, stationary burst events differ profoundly from random ones. Since fluorescence changes occur at fixed positions, the analysis can disregard vesicle movement and focus on the temporal evolution of intensity within defined regions. However, these recordings often span thousands of frames and encompass a large number of active sites, making frame-by-frame inspection impractical. Furthermore, signal kinetics vary depending on stimulation paradigms and acquisition rates, requiring an adaptable model capable of handling extended time sequences (**Figure 4c**).

Stationary burst events were primarily studied using neurons expressing SypHy (synaptophysin–SEP), a pH-sensitive reporter that increases fluorescence upon vesicle fusion and reacidifies during endocytosis. This probe provided high temporal resolution for detecting both evoked and spontaneous exocytosis events.

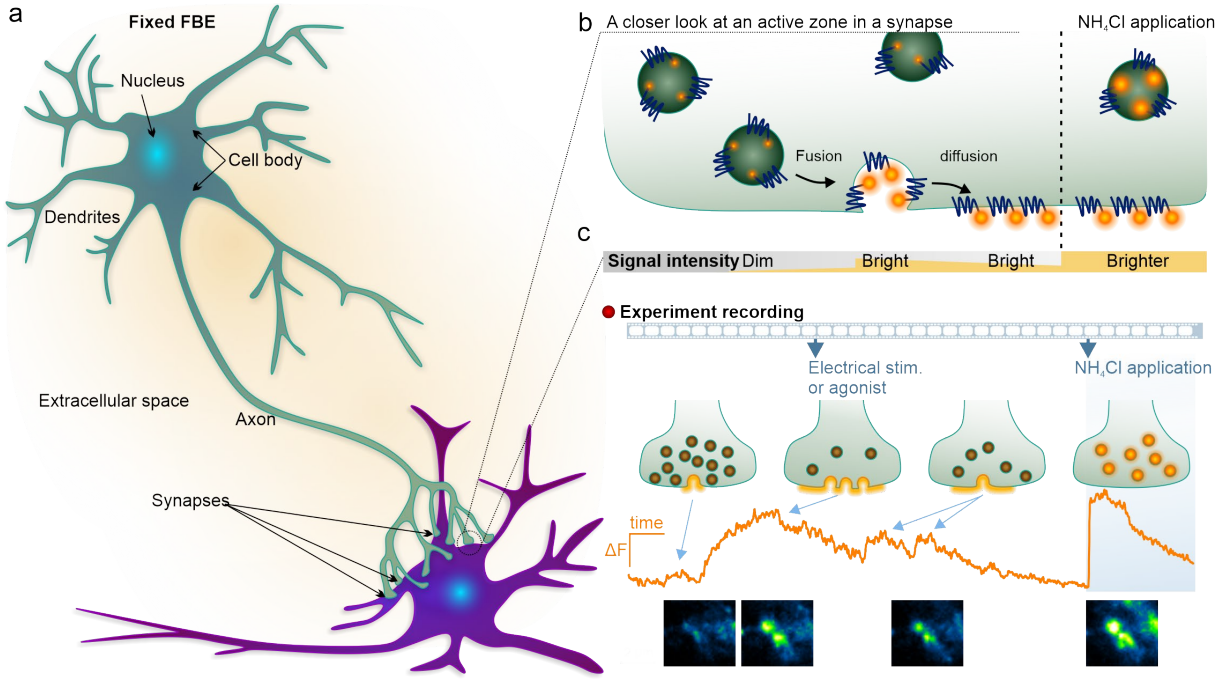


Figure 4. Synaptic granule exocytosis stages (adapted from (Chouaib et al., 2025)).
a. Schematic representation of neurons with synaptic transmission connectivity. **b.** Schematic representation of synaptic activity monitoring with synaptophysin-SEP (SypHy). As SEP is quenched in the acidic interior of synaptic vesicles, low fluorescence is detected when at rest. Upon fusion with the plasma membrane, SEP is exposed to the neutral extracellular environment, leading to a marked increase in signal. Applying $NH4^+$ to the medium equilibrates the vesicle lumen to a neutral pH, allowing all vesicles to appear fluorescent. **c.** Schematic illustration of SypHy-based recordings with real fluorescence intensity profiles from DRG neurons. The upper row displays different types of release behavior that can occur at individual synapses, with corresponding fluorescence traces shown in the middle row. The images in the bottom row display the synapse at the indicated moments. In asynchronous release, only short-lived fluorescence increases occur because a limited number of vesicles undergo fusion. Applying electrical or chemical stimulation can trigger coordinated release of many vesicles, producing a more persistent rise in fluorescence intensity. This occurs because SypHy remains in the membrane before retrieval. As endocytosis and lumen reacidification progress, the signal diminishes again. $NH4^+$ treatment yields the highest fluorescence levels by forcing all vesicles into a neutral state. Importantly, these fluorescence fluctuations arise from highly localized regions representing individual synapses.

1.9 Hotspot area extraction module

Beyond conventional fluorescent protein reporters, nanosensor technologies have recently opened new possibilities for visualizing neurotransmitter release (**Figure** 5**a-b**). One such approach, termed AndromeDA, uses single-walled carbon nanotubes coated with dopamine-sensitive polymers to create a thin sensing layer over neuronal processes (Elizarova et al., 2022). Upon dopamine release, the nanosensor layer produces rapid and localized changes in near-infrared fluorescence (**Figure** 5**b**). This allows the detection of individual release sites with sub-second temporal precision and micrometer-scale spatial resolution. This method avoids the need to genetically tag vesicle cargo and instead directly measures the substance released through exocytosis. Notably, AndromeDA revealed that neighboring varicosities of the same axon can differ significantly in their probability of dopamine release, underscoring the heterogeneity of synaptic signaling.

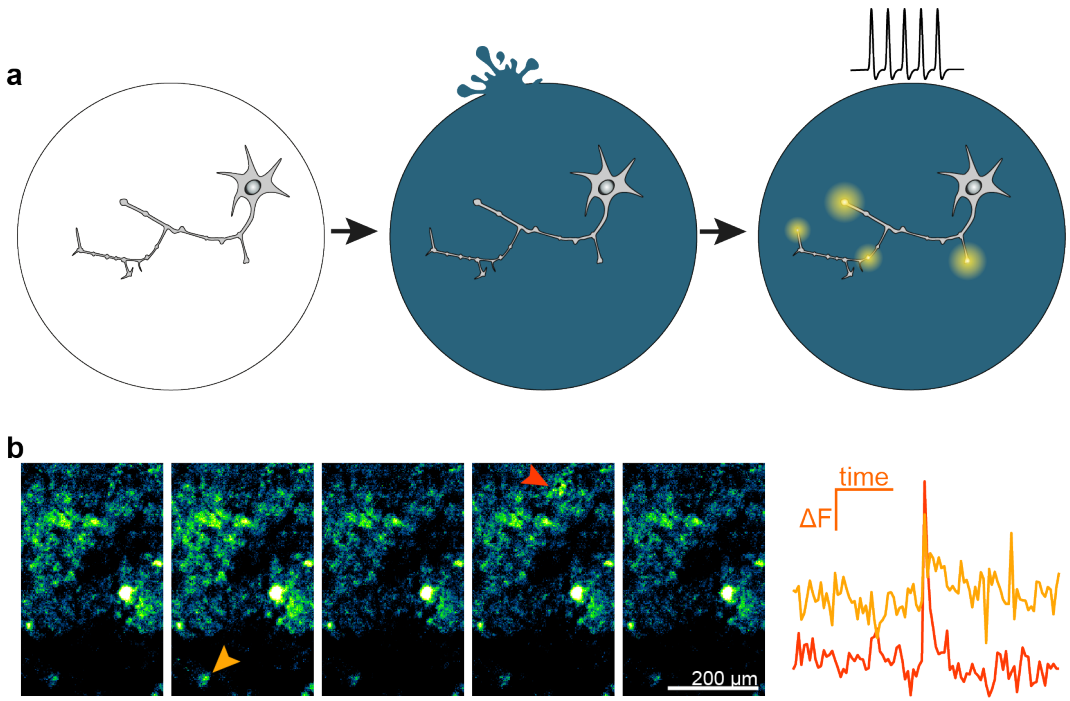


Figure 5. AndromeDA infrared nanosensor paint illustration (adapted from (Chouaib et al., 2025)).
a. Schematic representation of a dopaminergic neuron culture coated with fluorescent dopamine nanosensor-paint ("AndromeDA"). The first schematic shows a neuron without a nanosensor coating, followed by the application of the nanosensor paint, and the final schema illustrates the dopamine response upon stimulation**. b**. Cropped time-series data from a dopaminergic neuron culture. The overlay arrows point to the dopamine release sites (orange and red), corresponding to local nanosensor activation upon DA binding. The sequence of images (bottom left) shows individual frames from the recording, in which release sites brighten momentarily. On the right, the corresponding fluorescence ΔF traces over time display the two transient fluorescent events, illustrating how distinct release events produce sharp, time-locked increases in signal.

The utilization of AI and fully automated computation and analysis facilitates the processing of large datasets and the identification of latent patterns that may be overlooked by manual analysis, thereby reducing the potential for human bias. By integrating the disciplines of artificial intelligence and biological sciences, our objective is to develop innovative tools and methodologies for investigating cellular communication and functionality.

1.10 Motivation and aim

Manual analysis of live-cell imaging remains a major bottleneck in biological research, specifically when analyzing highly dynamic processes such as vesicle exocytosis. Detecting these transient fluorescence events requires screening thousands of image frames searching for fluorescence intensity fluctuations, such as intensity flash or sudden vesicle disappearance. Furthermore, the fluorescence signal often differs across fluorophores, cell types, and imaging conditions, which increases the complexity of finding and analyzing such events. This labor-intensive and subjective task limits both reproducibility and throughput, constraining the scale of quantitative biological studies.

To overcome these challenges, this project aims to develop a unified, automated framework that integrates artificial intelligence and computer vision for the reliable detection and analysis of vesicle

exocytosis. This thesis presents a novel software designed as a fully open-source, cross-platform ImageJ plugin called Intelligent Vesicle Exocytosis Analysis (IVEA). This software combines accessibility for biologists with the integration of parameter automation and deep learning for the highest reproducibility, accuracy, and precision. An extension of IVEA, IVEA-Py, is designed to integrate event detection, granule tracking, and quantitative analysis into a single unified platform. This platform facilitates the automated identification of fusion events and the direct association of these events with their corresponding vesicle trajectories. IVEA-Py is a powerful tool that combines traditional frame-by-frame inspection and modern AI-driven automation, providing a scalable and reproducible tool for large datasets.

The main objective of this research is to develop a novel system that is automatic, flexible, accurate, and extensible. This framework is designed to be adaptable to a variety of transient cellular phenomena, including calcium signaling and organelle trafficking. Additionally, we intend to maintain characteristics including performance, robustness, accessibility, and interoperability across different imaging modalities and biological systems.

1.11 Conceptual framework

IVEA integrates classical computer vision methods with deep learning architectures to capture the full range of granule-fusion behaviors. Within this framework, three complementary recognition strategies have been implemented, tailored to the heterogeneity of biological signals and imaging modalities. Random burst events are analyzed with an encoder-Vision Transformer (eViT) that models spatiotemporal patches (**Figure** 6**a**, **b**) (Chouaib et al., 2025) and generalizes across both pH-sensitive and pH-insensitive fluorescent reporters. Stationary burst events are processed as a multivariate time series (**Figure** 6**d**) and classified using an LSTM (Hochreiter & Schmidhuber, 1997) (**Figure** 4a-c), which is well-suited for recurrent release at fixed sites (**Figure** 6**a**, **c**). IVEA was extended to support nanosensor-based assays through a hotspot area extraction analysis module, first developed in the context of the DART/AndromeDA project (Elizarova et al., 2022). This module detects spreading surface signals generated in nanosensor films and classifies them alongside more conventional vesicle fusion events. By combining vesicle-focused imaging (in CTLs and neurons) with nanosensor-based readouts, IVEA offers a unified framework that connects molecular cargo release with extracellular neurotransmitter dynamics, broadening the scope of biological processes accessible to automated, AI-driven analysis.

Integrating these three strategies into IVEA demonstrates how different computational paradigms, vision-based ROI detection, sequence modeling, and spatiotemporal deep classification can be combined to achieve both batch-level automation and highly accurate recognition. This hybrid design ensures adaptability across cell types, labeling methods, and imaging platforms, while linking naturally to IVEA's broader framework of automated parameter estimation, trajectory-event association, and

transfer learning. Taken together, these features establish IVEA as a flexible and extensible platform for activity recognition in live-cell imaging (Chouaib et al., 2025).

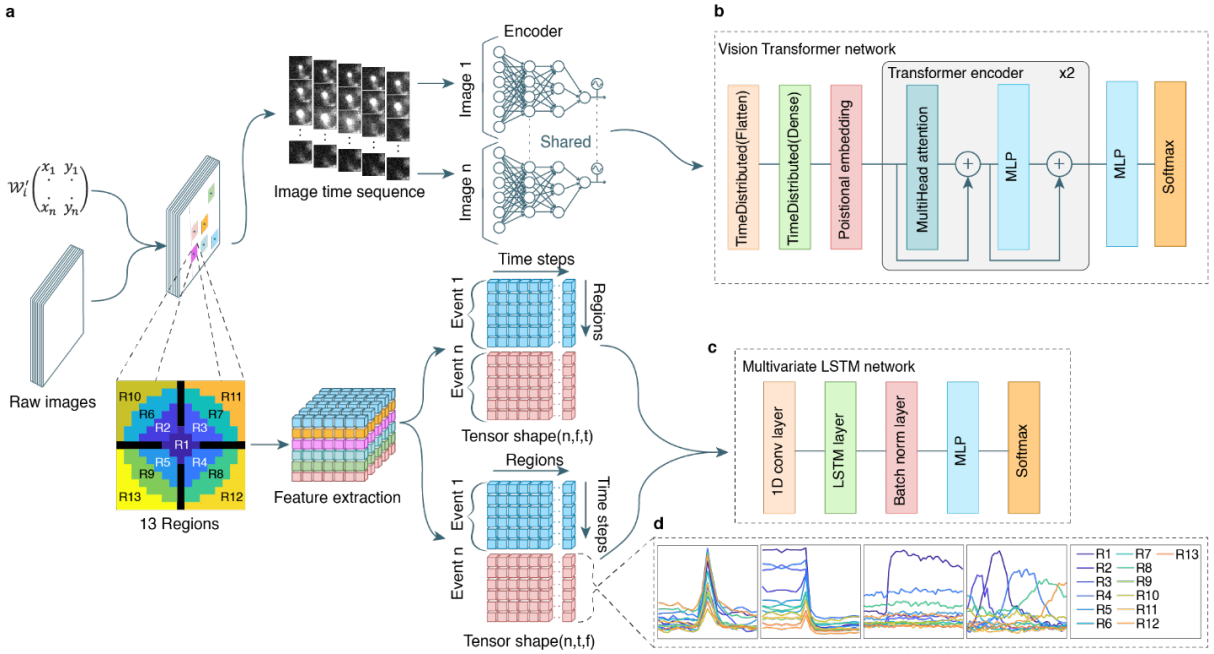


Figure 6. Overview, our neural networks and the feature extraction process (adapted from (Chouaib et al., 2025)).

a. Overview of the two distinct data preparation methodologies used prior to classification. For random burst events module, each candidate region is represented as an LM coordinate matrix $\mathcal{W}_i' \in \mathbb{N}^{2\times d}$. From every selected region, 26 patches of size 32×32 pixels are sampled and passed into a shared convolutional encoder composed of three 3D convolutional layers. For the second module, the stationary burst events, features are derived from 13 small regions positioned around the local maximum of the event (R_1, R_2…R_{13}). In the feature-extraction scheme, black tiles denote separating areas excluded from analysis. The resulting feature sets are indexed according to event number, spatial region, and temporal sequence. **b**. Diagram of the Vision Transformer-based encoder used for random burst classification. **c.** Diagram of the multivariate LSTM network used for analyzing stationary burst activity. **d**. The LSTM model treats each input as a collection of 13 temporal intensity traces, one per region, representing the normalized change in mean fluorescence over time. This model can also be applied to random burst events. Example patterns include: (1) a single T-cell granule fusion with a pH-sensitive fluorescent cargo (left), (2) a pH-insensitive fluorescent cargo (middle left), (3) synaptic vesicle fusion in neurons labeled via a pH-sensitive membrane probe (middle right), and (4) a fusion event from a granule that undergoes lateral movement prior to release (right).

1.12 Overview of the IVEA platform

A central design choice in IVEA is automation for batch analysis. When detection parameters are left at their default zero values, the software infers them automatically from the first four frames of each recording. During this initialization, the system estimates thresholds, sensitivity, and employs a ratio-based linear temporal normalization on-the-fly to compensate for gradual photobleaching (without modifying the raw data). To preserve salient temporal cues while reducing sequence length, IVEA can apply temporal max pooling instead of simple downsampling. This can also accelerate processing by evaluating frames using a configurable sampling window when appropriate.

Event analysis proceeds as a connected pipeline. First, local maxima are identified and linked in space and time to define candidate regions of interest. Detection parameters are estimated from the initial

frames, providing baseline thresholds that limit unnecessary computation by pre-selecting plausible regions for later classification. Second, spatiotemporal data centered on each candidate is extracted and provided to the appropriate neural network (eViT for random burst events; LSTM for stationary burst events (Chouaib et al., 2025)) for activity classification, effectively replacing human triage with a reproducible, data-driven decision. Third, a novel technique is introduced using Gaussian non-maximum suppression to eliminate duplicates. The spatiotemporal Gaussian formulation enables suppression of overlapping detections based on both spatial proximity and temporal persistence, providing a continuous alternative to fixed-radius search and discrete box-based non-maximum suppression. Fourth, in IVEA-Py, vesicles are tracked and traced, and trajectories are linked to the predicted fusion events, enabling inspection of pre-fusion motion and site usage. Finally, quantitative measurements are produced, including fluorescence traces, fusion metrics, and positional readouts suitable for downstream statistical analysis.

Beyond analysis, IVEA includes a training and refinement environment that supports transfer learning, allowing users to adapt the pretrained models to new labels, cell types, or related phenomena with minimal data, to only twenty events or maybe less. To lower the barrier for adoption and reproducibility, the platform is open-source and ships with expert-labeled datasets for exocytosis, providing a high-quality starting point for model reuse and community extension. To our best knowledge, this constitutes the first deep-learning-based, end-to-end software specifically oriented toward exocytosis analysis that couples automated batch parameterization, spatiotemporal deep recognition, and custom non-maximum suppression in (x, y, t), trajectory linking, and integrated retraining in a cross-platform, open-source plugin (**appx. Table 10**).

2 Mathematical Foundations and Methods

2.1 Digital images

A digital image is composed of a matrix of numbers, known as pixels, which represent visual information that computers can process. Digital images are created by capturing light with cameras equipped with electronic sensors. Common sensors are charge-coupled device (CCD) and complementary metal-oxide-semiconductor (CMOS), each with its own advantages and disadvantages. In these sensors, photons are first converted into electrical charge by photodiodes, while transistors within the pixel circuitry handle amplification, resetting, and readout. For laser scanning microscopes, the emitted photons are collected by photomultipliers. The captured signals are then digitized into numerical values that represent the intensity (grayscale), or the spectral composition (RGB) of light at each pixel. This digital information is organized as a stream of binary data (bitstream, zeroes and ones) that holds visual information. The information is stored in a format that can be displayed, edited, or analyzed by computers, such as TIFF, LSM, CZI, etc... In digital images, each pixel encodes the detected photon signal as a numerical value produced by the sensor. These values are digitized and stored according to the image's bit depth, which determines the available intensity range (e.g., 8, 12, 16, 24, or 32 bits). Each pixel can be regarded as a compact package of information containing the recorded signal (**Figure 7**).

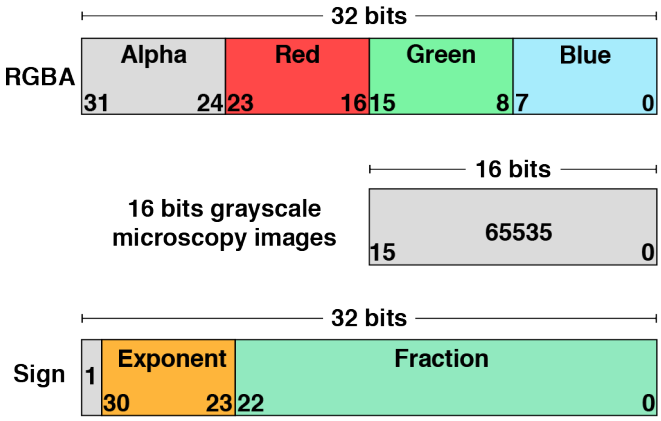


Figure 7. One pixel data package.
The first package is a 32-bit Alpha RGB pixel, where 8 bits are assigned for each channel. Alpha (transparency, usually used with the PNG file format), Red, Green, and Blue, corresponding to bit ranges 31–24, 23–16, 15–8, and 7–0, respectively. If the image is only RGB, then the total bits package is 24 bits. The second package displays a 16-bit microscopy grayscale pixel, where all 16 bits (15–0) encode a single intensity value in the range 0–65535. These are unsigned integers and do not represent negative values or decimals, allowing high dynamic range for fluorescence TIRFM or confocal imaging. The last package is a 32-bit floating-point microscopy pixel, represented in IEEE-754 single-precision format with one sign bit, eight exponent bits, and 23 fraction bits. This encoding allows storage of both negative and positive values, as well as decimal values, making it suitable for processed microscopy images that require extended numerical precision for analysis.

When reading digital images, the first step is to determine how the data are stored in memory, which is defined by the file format and associated metadata. Metadata typically specifies the image dimensions

as well as acquisition details such as resolution and laser wavelength (e.g., width, height, number of channels, number of slices, number of frames, time stamps, etc.). Correct interpretation of these dimensions is crucial, since images are stored as linear arrays in computer memory. The linear storage of n-dimensional data of shape $(D_1, D_2, \ldots, D_n)$ is expressed as:

$$i = \sum_{k}^{n} d_k \cdot \prod_{j=1}^{k-1} D_j$$

Here, $d_k$is the index at D_k dimension. In the simplest case of a 2D image with dimensions X×Y, the pixel at index (x, y) (zero-based indexing) is stored using the previous equation as:

$$i_{2D} = x \cdot (1) + \mathrm{X} \cdot y$$

$$= x + \mathrm{X} \cdot y$$

Here, X is the width of the image, which is the stride X reflecting how many elements must be skipped when moving one step along the y-axis. This concept holds true for higher dimensions, like adding a Z dimension at the z-axis as XYZ, the index will be expressed as:

$$i_{3D} = x(1) + \mathrm{X} \cdot y + \mathrm{X} \cdot (\mathrm{Y} \cdot z)$$

$$= x + \mathrm{X} \cdot (y + \mathrm{Y} \cdot z)$$

And if we add more dimensions like channel C and time T dimensions ordered as XYCZT, the index is expressed as:

$$i_{5D} = x + \mathrm{X} \cdot \Big(y + \mathrm{Y} \cdot \big(c + \mathrm{C} \cdot (z + \mathrm{Z} \cdot t)\big)\Big)$$

where x, y, z, c and t are the indices in the X, Y, Z, C, and T dimensions, respectively.

If the dimension order changes, for example, from XYCZT to XYZCT, the indexing equation must also be adjusted, as the relative position of the channel and slice dimensions shifts. Correctly interpreting this ordering is essential for reading multidimensional microscopy images without errors. Furthermore, digital image data are stored in binary form, where pixel intensities are mapped into arrays of bytes depending on the bit depth (**Figure 7**). Supporting multiple formats, therefore, requires careful handling of both dimensional indexing and binary encoding. This task can become complex and computationally demanding, particularly when working with libraries such as ImageJ that natively expect specific file structures.

To simplify this process, the Bio-Formats library (Linkert et al., 2010) was employed as a central tool for managing microscopy image data. Bio-Formats provides support for a wide range of proprietary file formats generated by major microscope manufacturers, ensuring that both image data and associated

metadata (e.g., pixel dimensions, acquisition settings, and temporal information) can be accessed in a standardized way.

Within ImageJ, Bio-Formats was used to directly import multidimensional image stacks while preserving calibration and experimental metadata. In MATLAB, the Bio-Formats reader facilitated batch access to large datasets, enabling automated preprocessing and quantitative analysis. For the Python workflows, the *aicsimageio* library was utilized as a lightweight, Bio-Formats-compatible solution to read multidimensional microscopy files and integrate them into downstream processing pipelines.

While there are other libraries that could be helpful in image read/write (Tifffile), implementing Bio-Format is important to read images from different acquisition systems, standardize, and save them in interoperable formats such as TIFF. In addition, Bio-Formats has well-organized and user-friendly documentation and is supported by the community.

2.2 Foreground detection

In image processing and computer vision, foreground detection refers to the process of identifying and separating foreground elements from the background. The principal objective for activity recognition is to detect the changes in an image sequence, which helps identify fluorescence intensity variation or moving objects in the foreground (i.e., granules, fusion events, etc.). A variety of techniques may be employed to achieve foreground detection, including temporal averaging, background subtraction, optical flow, neural networks, and other methodologies. I have opted to pursue a hybrid approach, integrating conventional techniques such as background subtraction mixed with a neural network to identify and classify events occurring in the foreground accurately. Background subtraction can be performed either by subtracting the initial frame from the current frame or by using a moving-window technique, defined as $\Delta I = I_i - I_{i-n}$, where i and n refers to the current frame and the nth frame, respectively.

2.3 Image segmentation

The image segmentation process encompasses partitioning the image into multiple regions, which are subsequently referred to as a "segmented image" or "labeled image". In image processing, a labeled image is a matrix whose elements are the IDs of connected regions rather than gray values. To illustrate, if an image contains 10 objects, the maximum value in the segmented image would be 10, with 0 representing the background values. The purpose is to represent the image in a more meaningful manner or to detect, extract, and track objects by assigning them IDs. There are different ways for image segmentation, including traditional and advanced methods such as thresholding, edge-based segmentation, morphological reconstruction, clustering algorithms, neural network instant segmentation, and others. Traditional methods typically employ a threshold value to segment an image, whether it is a global threshold or an adaptive threshold. A global threshold value represents a single

number applied uniformly across the entire image, whereas an adaptive threshold computes different thresholds for each pixel using the intensity statistics of the local neighborhood. This approach handles spatial variations in fluorescence intensity. Pixels with values below the specified threshold are set to zero, while those above are set to one, resulting in a binary image. Subsequently, a connected component labeling method is employed to assign unique identifiers to each connected region. Advanced methods using clustering algorithms or neural network segmentation typically detect, classify, and assign identifiers directly to the image, converting it into a labeled image. Depending on the technique used, some methods are advanced enough to separate individual objects or even segment the entire scene in the image. For example, techniques like semantic segmentation (Long et al., 2014) can label each pixel in an image with a class, while instance segmentation (Arefi et al., 2024) can distinguish between different instances of the same object type. A third deep learning approach, panoptic segmentation (Kirillov et al., 2018), combines both the semantic and instance segmentation. It provides a comprehensive view of the scene by labeling each pixel as either part of a specific object instance or background.

In our case, the target features, such as granules, boutons, and synaptic sites, do not exhibit consistent morphology or texture. These regions' signatures are characterized by transient fluorescence intensity variations in both space and time. The objective of segmentation in this context is to isolate dynamic signal changes rather than to delineate object boundaries. For this reason, IVEA implements a set of adaptive and computationally efficient classical methods for segmentation that are suited to fluorescence microscopy data (see discussion section). An iterative global thresholding approach is used to estimate the background and to highlight pixels that exhibit temporal variation. A difference-of-Gaussian (DoG) filter enhances local maxima, allowing adaptation to variable spot sizes and brightness levels. Subsequently, k-means clustering separates signal and background components based on pixel intensity statistics. Together, these operations enable the extraction of candidate regions with a high signal-to-noise ratio. Connected component labeling of these regions is then performed using an eight-connected recursive function to assign region identifiers.

2.4 Morphological filters

Morphological filters are a set of nonlinear image processing operations that analyze and modify images based on their geometric structure. They work by applying a structuring element (a small matrix or kernel) to the input image, probing its shape and boundaries. These filters are commonly used for tasks such as noise reduction and enhancement of image structures. Some are applicable to binary or grayscale images. Morphological operations utilized in IVEA for binary images are erosion, dilation, and the maximum filter.

The erosion and dilation are two inverse operations denoted by $\ominus$ and $\oplus$, respectively, one used to shrink objects in a binary image and the other for expansion. The erosion of the image I by a structuring element B is defined as:

$$I \ominus B = \{z | B_z \subseteq I\}$$

where B_z is the translation of the structuring element by a vector z. For a given point z to be retained within the image, it is necessary that the structuring element B, which is centered on z, fits entirely within the foreground of the image.

In contrast, the dilation expands the boundaries of objects in a binary image, expressed as:

$$I \oplus B = \{z | (B_z \cap I) \neq \emptyset\}$$

This means that the set of points z where the structuring element overlaps any part of the object, is included in the output. Dilation increases the size of bright regions, bridges small gaps, and fills narrow holes in the structure.

Typically, these operations are employed in combination, such as dilation followed by erosion, which is referred to as the "closing" operation. This results in the expansion of object boundaries and the filling of holes, which is then followed by erosion to shrink the object's size back. The combination of erosion and dilation is referred to as an "opening" operation. This is because the erosion operation shrinks and expands holes in objects, whereas dilation is utilized to restore the original size of the objects. However, if small regions are removed during erosion, they cannot be recovered through dilation.

In the IVEA framework, morphological operations such as erosion, dilation, and the maximum filter are used in the hotspot area extraction module. These operations were applied primarily to binary images at the iterative threshold step to refine object masks and improve segmentation outcomes over possible activities.

2.5 Ricker wavelet

The Ricker wavelet (Mexican hat) is a continuous wavelet commonly employed in both signal and image processing. This function serves as a high-pass filter, which emphasizes regions with rapid intensity change, such as edge detection and feature extraction. The Ricker wavelet is derived in the same way as the Laplace of Gaussian equation, but it is the inverted, normalized second derivative of a Gaussian, expressed as follows:

$$\psi(x,y) = -LoG(x,y) = \frac{1}{\pi\sigma^4}\left(1 - \left(\frac{x^2+y^2}{2\sigma^2}\right)\right)e^{-\left(\frac{x^2+y^2}{2\sigma^2}\right)}$$

where σ represents the standard deviation controlling the width of the wavelet, x and y are the special coordinates.

When applied to an image $I(x, y)$, the Ricker wavelet is convolved with the image to emphasize localized features and suppress uniform background regions, producing a new feature-enhanced image $\mathcal{R}(x, y)$ expressed as:

$$\mathcal{R}(x, y) = \psi(x, y) * I(x, y)$$

In the context of biological analysis, particularly in the field of vesicular studies, the Ricker wavelet can be employed as a helpful tool to emphasize the granule shape against the background. Given that the granule 2D images appear to be round Gaussian shapes, a suitable σ with an appropriate threshold would result in granules being identified as the foreground.

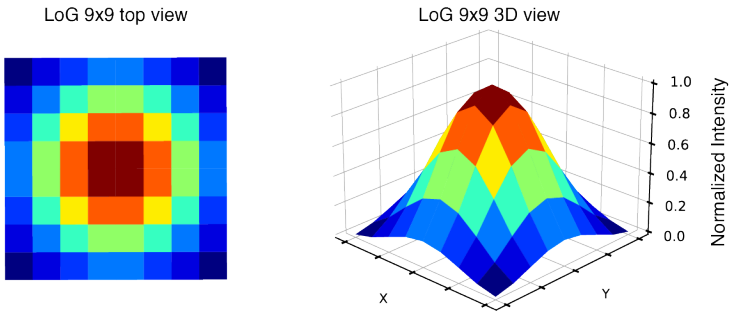


Figure 8. 2D and 3D representations of the Laplacian of a Gaussian (LoG) with kernel 9x9.
The left panel shows the 2D spatial distribution of kernel weights as a heatmap, where brighter colors denote higher positive values and darker pixels correspond to negative regions. The right panel depicts the same kernel in 3D surface form, highlighting the central positive peak surrounded by lower intensity regions, a structure that enhances edge detection by combining smoothing with second-derivative response. This kernel is normalized to the 0–1 range for visualization purposes.

A limitation of the wavelet is that it may amplify noise signals, which could result in false detection. The combination of denoising methods with the Mexican hat and a median absolute deviation (MAD) would enhance the detection of granules and render the process more robust. The default discrete Ricker wavelet kernel utilized in IVEA constitutes a zero-mean normalized matrix of dimensions [7×7] with a standard deviation σ = 1 for the detection of small granules. In the case of larger granules, it is advisable to employ a distinct kernel to align with the granule dimensions. For instance, a kernel size of [9×9] σ = 2 or 3 may be more appropriate (**Figure 8**). This kernel is commonly used to emphasize regions of rapid intensity change while reducing the effect of high-frequency noise relative to the kernel size. This kernel is commonly employed to enhance regions that exhibit rapid intensity variations while simultaneously attenuating high-frequency noise relative to the kernel's spatial scale (**Figure 9a, b, d**). When the kernel size and σ are appropriately matched to the object dimensions, the filter can effectively separate adjacent or merged structures. In this implementation, it is primarily applied to small granules to achieve higher detection accuracy. However, if the kernel parameters are not properly adapted to the granule size, the filter may produce false positives or noise-induced artifacts (see Discussion section 4.3 Granule Detection and Tracking Module).

2.6 Difference of a Gaussian

Difference of a Gaussian (DoG) is the difference between two Gaussian-blurred versions of an image with different standard deviation σ_1 and σ_2 (**Figure 9a, b-c**). This method is a simple and efficient technique used for feature detection and blob detection such as granules. The DoG approximates the results of LoG filter (**Figure 9c, d**), but with much less computation time, expressed as:

$$DoG(x, y) = G(x, y; \sigma_1) * I(x, y) - G(x, y; \sigma_2) * I(x, y)$$

Here $G(x, y; \sigma)$ represents the Gaussian function applied to the image, which is expressed as:

$$G(x, y; \sigma) = \frac{1}{2\pi\sigma^2} \cdot e^{-\frac{(x^2+y^2)}{2(\sigma)^2}}$$

The relation between the standard deviations σ_1 and σ_2 are often related by a simple constant factor, such as:

$$\sigma_2 = c \cdot \sigma_1$$

where constant c controls the separation between the two Gaussian scales. Increasing c widens this separation, which makes the DoG filter more sensitive to larger structures in the image.

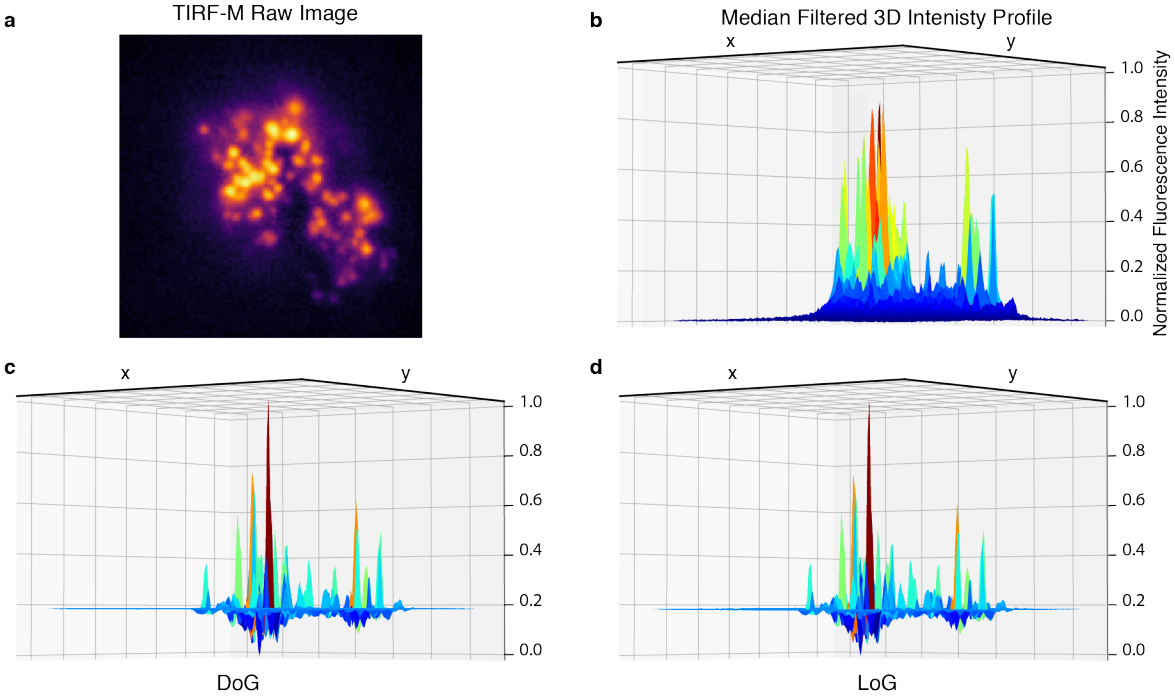


Figure 9 Demonstration of Gaussian-based filtering approaches for fluorescence feature enhancement. **a**. Raw TIRF-M fluorescence image of chromaffin cell granules labeled with mCherry. **b.** Three-dimensional surface plot of the median-filtered image showing the spatial distribution of fluorescence intensity. **c.** Surface representation after applying a Difference-of-Gaussian (DoG) filter with sigma ration = 1.2. **d.** The same region processed using a Laplacian-of-Gaussian (LoG) filter, kernel used is 7 pixels with sigma = 1. Both filters emphasize local intensity variations and reduce background noise, yielding similar high-frequency representations of the fluorescence pattern.

The value of c is commonly chosen as either 1.6 or $\sqrt{2}$. The 1.6 value used in the Scale-Invariant Feature Transform (SIFT) (Lowe, 2004) is empirically found to find a good balance between preserving fine details and smoothing noise while eliminating irrelevant information. The use of $\sqrt{2}$ doubles the area covered by the Gaussian filter (Lowe, 2004). While these values are experimentally tested and adjusted, other values can be used to get the desired results.

2.7 Median absolute deviation

The median absolute deviation (MAD) is a global threshold technique employed for the segmentation of images based on the measurement of statistical dispersion. MAD is calculated as the median of the absolute differences from the image median, such as:

$$MAD = median(|I(x,y) - median(I)|)$$

The MAD provides a robust measure of variability, particularly in the presence of noise and extreme outliers. This outcome arises from the utilization of the median rather than the mean value. The global MAD-based threshold is scaled by a tunable constant θ, which determines the sensitivity to intensity variations. The threshold is then adjusted by the image median to align it with the central intensity level, expressed as follows:

$$T_{MAD} = median(I) + \theta \cdot MAD$$

Subsequently, the threshold T_{MAD} is applied to the entire image, resulting in a binary image 0 or 1, represented in the following criteria:

$$I_{thresholded}(x,y) = \begin{cases} 1, & if\ I(x,y) > T_{MAD} \\ 0, & otherwise \end{cases}$$

2.8 Granule detection and recognition using gradient flow vector field

The granule detection method implemented in this section builds on the idea of particle convergence along a gradient vector field (GVF) derived from the image's spatial intensity gradients. This method aims to locate the center of each granule by following the local gradient direction from candidate pixels. As these paths propagate, they gradually converge toward stable endpoints, which serve as the estimated granule centroids. This approach differs from classical GVF diffusion implementation by avoiding the need for iterative diffusion and flux-based thresholding (G. Li et al., 2007). Instead, it integrates a computationally efficient forward Euler scheme over normalized gradients of distance-transformed images. The Euler flow-following step was inspired from the gradient flow tracking strategy used in Cellpose (Stringer et al., 2021).

First, each input frame is normalized to the [0, 1] range to ensure consistency across varying fluorescence intensities. Noise is then suppressed by applying a median filter with a fixed kernel size.

To enhance structures of interest, particularly granules, a DoG or LoG filter is applied (**Figure** 8). The result is a bandpass-enhanced image thresholded using MAD. Thresholds are computed as:

$$T_k = median(|DoG(x,y)|) + k \times MAD(|DoG(x,y)|)$$

where k is a sensitivity constant adjustable by the user (default = 5).

A binary mask $M(x,y) \in \{0,1\}$ is formed by setting pixels in the DoG image above the computed threshold to 1, indicating potential granule regions. In special cases (e.g., nonuniform intensity regions), multiple threshold layers may be combined to ensure robustness if we use the k-means to segment the images based on pixels' intensities (Pham et al., 2000).

$$M(x,y) = DoG(x,y) > T_k$$

Following thresholding, the Euclidean Distance Transform (EDT) $\phi(x,y)$ (Strutz, 2021), is computed over the binary mask. This transform encodes, for each pixel within the object mask, the shortest distance to the background expressed as:

$$\phi(x,y) = \min_{(x',y')\in\partial M} \sqrt{(x-x')^2 + (x-y')^2}$$

If the user disables the EDT computation, an alternative field is generated by multiplying the binary mask with the DoG image and then computing the gradient of the product.

$$\phi(x,y) = M(x,y) * DoG(x,y)$$

Although this preserves shape-specific cues and may yield finer detection in some cases, it is more sensitive to nonuniform and deformed shapes. In such cases, the distance-based filtering is deactivated, and clustering relies solely on the vote of the gradient convergence vectors.

When we compute the gradient field $\nabla\phi(x,y)$, it naturally points from boundaries toward the interior, peaking at the geometric center of each shape or at the local maxima if the EDT is not used (see Discussion section 4.3 Granule Detection and Tracking Module). The gradient field is expressed as:

$$\nabla\phi(x,y) = \left(\frac{\partial\phi}{\partial x}, \frac{\partial\phi}{\partial y}\right)$$

The gradient is calculated using finite differences. For stability and to ensure uniform step sizes during integration, each gradient vector is normalized such as:

$$g(x,y) = \frac{\nabla\phi(x,y)}{\|\nabla\phi(x,y)\| + \epsilon} \quad \text{with } e = 10e^{-8}$$

Each pixel in the normalized gradient image $g(x,y)$ is then treated as a particle and integrated through the normalized field using forward Euler steps:

$$(x_{n+1}, y_{n+1}) = (x_n, y_n) + \tau \cdot g_n(x_n, y_n), \quad n \leq N-1, \qquad \tau, n \in \mathbb{N}$$

where $\tau = 1$ pixel is the integration step length. The number of iterations, $N = 20$, defines the maximum number of steps. Integration halts early if the pixel exits the mask or the vector magnitude is negligible.

The resulting trajectories terminate at endpoints presumed to be positioned near local maxima of $g(x, y)$. These endpoints resembling possible centroid coordinates are clustered using Density-Based Spatial Clustering of Applications with Noise (DBSCAN) to identify converging flows (Ester et al., 1996). Each cluster is treated as a potential granule, with a minimum vote requirement used to eliminate spurious or poorly supported peaks. From each cluster, the granule centroid is selected corresponding to the endpoint, which is the highest value in the normalized gradient image or the EDT map (if EDT is enabled).

To further refine detection quality, a vote-based filtering step evaluates the reliability of each cluster by analyzing its number of members. A sensitivity parameter $s \in [0,1]$ is used to set a threshold relative to the mean vote count. Only clusters whose votes exceed $s.\mu$, where μ is the mean number of votes, and are retained. This allows dynamic control of detection sensitivity while avoiding hardcoded thresholds.

Finally, to support time-lapse analysis, the detected granule centroids are passed to a nearest-neighbor tracking module that links detections across frames. A Kalman filter is applied to smooth trajectories and interpolate across brief detection gaps, ensuring robust and biologically plausible granule tracking in challenging imaging conditions.

2.9 K-Means clustering

The k-means clustering algorithm is situated within the unsupervised machine learning partition (**Figure 1**). This algorithm is used to partition a dataset into k distinct clusters (Macqueen, 1965). The algorithm begins by randomly or selectively assigning each cluster's center. Each data point is then assigned to the cluster with the nearest gray mean value, which serves as the cluster's centroid. In an iterative process, the k-means algorithm refines the coordinates of each cluster by minimizing the squared Euclidean distance, a measure of intra-cluster variance. The objective of k-means is to minimize the within-cluster sum of squares (WCSS) such that:

$$c_i = argmin_j \left\| x_i - \mu_j \right\|^2$$

where c_i is the cluster assigned to the data point x_i, and $argmin$ is the index of the minimum value.

To update the cluster centroids' positions, we compute the mean of all points belonging to each cluster. This operation is done using the Update Step equation:

$$\mu_j = \frac{1}{|\mathcal{C}_j|} \sum_{x_i \in \mathcal{C}_j} x_i$$

where μ_j is the new centroid of the cluster j, and $\mathcal{C}_j$ is the number of data points in the cluster j.

Although the algorithm does not always find the optimum WCSS, it can converge when the WCSS becomes stable such as:

$$\sum_{j=1}^{k} \sum_{x_i \in \mathcal{C}_j} \|x_i - \mu_j\|^2$$

The k-means algorithm was applied in IVEA to increase the dimension of the image I into multiple layers $I(x, y) \rightarrow I(x, y, k) \mid I \in \mathbb{N}^k$. This helped segment the images for further analysis, assuming that the background is the group of pixels with the smallest mean cluster value. The k-means clustering algorithm was used to cluster image pixels based on their gray-level means (Pham et al., 2000). The algorithm was implemented in Java using the ImageJ library, making it available and runnable within ImageJ Fiji.

2.10 Convolution layer

Convolution, denoted by the symbol "$*$" is a mathematical operation that merges two functions, shaping a new one, such as one function modified by another. For two continuous functions $f(t)$ and $g(t)$, their convolution is defined as:

$$(f * g)(t) = \int_{-\infty}^{\infty} f(\tau) g(t - \tau) d\tau$$

where τ is a dummy variable used to shift and overlap the functions being convolved.

In the context of discrete signals and images, convolution is often used in image processing and computer vision. For 2D discrete convolution, we work with images as matrices convolved with a matrix (kernel) of $[n \times m]$ dimensions instead of continuous functions, such as:

$$(I * K)(x, y) = \sum_{i=-k}^{k} \sum_{j=-l}^{l} I(x - i, y - j) \cdot K(i, j)$$

Here, I denotes the input image and K the convolution kernel. The pair $(x, y)$are the coordinates of the output pixel, while (i, j) are the kernel indices. The parameters $k$and l are the half-width and half-height of the kernel, so the kernel size is $(2k + 1) \times (2l + 1)$.

In a neural network, a convolutional layer is employed to extract features from the input data. This is accomplished by applying 1D, 2D, or 3D convolution operations, which encode the input and pass the resulting feature maps to the next layer. The key components of this operation include the kernel (or

filter), stride, and padding. These three components determine the output results, values, and tensor dimensions. The kernel slides over the height and width of the input image, producing a 2D feature map. The stride is the number of pixels that defines how far the kernel slides each time it is applied to the input. If the stride is one, this means the filter advances by a single sample/pixel at a time, whereas a stride of two skips every other sample/pixel at a time, which would downsample the output image by a factor of 2. The padding operation adds artificial borders around the input, enabling convolution to be applied near the edges and allowing control over the final output size. There are different types of padding techniques used with convolution, but only two are built into Google TensorFlow: "valid" and "same" padding. The "valid" padding means no padding is applied. The output size will be smaller than the input size if the kernel size is greater than one. Using the "same" padding option, a symmetric zero padding is applied to extend the image borders, ensuring the output maintains the exact dimensions of the input.

To calculate the dimension of the output feature for a 2D convolution layer, we use:

$$output\ height = \left[\frac{input\ height - kernel\ height + 2 \times Padding}{Stride}\right] + 1$$

$$output\ width = \left[\frac{input\ width - kernel\ width + 2 \times Padding}{Stride}\right] + 1$$

In the 3D convolutional layer case, a third dimension is added, enabling the kernel to slide over this additional dimension and facilitating the extraction of features from depth, channels, or time-series data. The third dimension of the output layer in our case is the time dimension, while the output dimension is computed as in the previous equations; time is a dimension in this case.

2.11 Multilayer perceptron

The multilayer perceptron (MLP) is a simple deep learning model that represents a fully connected network (FCN). An MLP consists of multiple layers of nodes, where each layer is connected to the next. The MLP can be used separately for simple tasks or as a part of larger models for more complex tasks. It is usually formed of multiple Dense layers stacked on top of each other. A Dense layer is also known as a fully connected layer (**Figure 10**). It is a type of neural network layer in which each neuron is connected to all neurons in the preceding layers. This layer computes the weighted sum of its inputs, adds a bias term, and passes the result through an activation function, such as:

$$y = \sigma(W.x + b)$$

Here, y represents the output of the Dense layer, W the matrix of shape $[n \times m]$, where m is the number of inputs and n is the number of neurons in the Dense layer. σ represents the activation function applied elementwise. x is the input vector of shape $[m \times 1]$ and b is the bias vector of shape $[n \times 1]$.

2.12 Max pooling

Max pooling is a downsampling operation commonly used in neural networks to reduce the dimensionality of feature maps while retaining the most salient activations. It functions by sliding a fixed-size kernel across the input tensor and selecting the maximum value within each window to populate the corresponding position in the output. In spatial domains, an $n \times n$ kernel operates over local 2D regions, capturing prominent features such as edges or textures. In temporal domains, a 1D kernel of size n is applied across successive time steps at a fixed spatial location, extracting peak temporal values. This operation reduces computational complexity, introduces translational invariance, and promotes generalization by discarding redundant or non-informative variations.

2.13 Convolutional neural network

CNNs are a type of feedforward neural network that belongs to the category of deep learning (**Figure 10**). They are designed to process structured grid data using 2D convolutional layers and can also capture spatiotemporal features using 3D convolutional layers, such as the CryoONE neural network used for ab Initio 3D protein reconstruction from super-resolution 2D fluorescent images (Shaib et al., 2024). CNNs are employed in a wide range of tasks, including image denoising, classification, object detection, and segmentation, such as YOLO (Redmon et al., 2015) and U-Net (Ronneberger et al., 2015), scene reconstruction, 3D model reconstruction, such as NeRF (Mildenhall et al., 2020b), and activity recognition. In the present era, CNNs are utilized in a multitude of applications, including mobile devices for facial and scene recognition, such as camera focus using AI, as well as in search engines like Google, and language models like Copilot and ChatGPT. The CNN network was used in IVEA to build the encoder network (**Figure 18**), which was used to extract features from the input images and feed them to the ViT network.

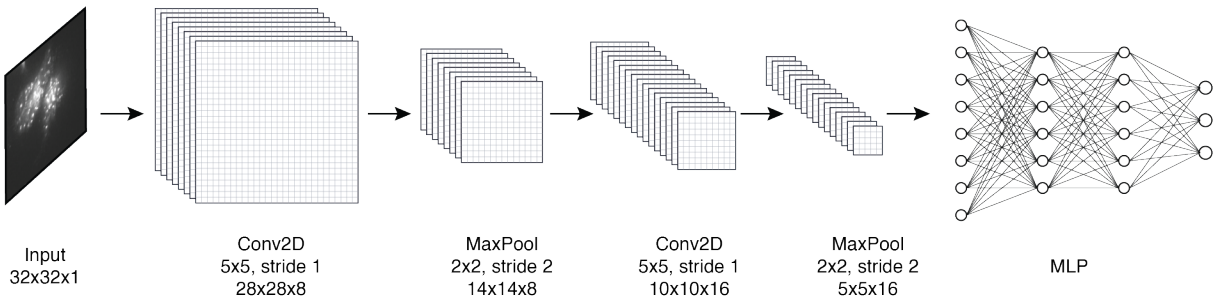


Figure 10. Simple CNN architecture illustration.
This image displays a simple CNN architecture, comprising two convolution layers paired with a max pooling layer. This is followed by a classification MLP that consists of four fully connected dense layers.

2.14 Recurrent neural network

A recurrent neural network (RNN) is a class of artificial neural networks specifically designed to process sequential data (**Figure 11a**). Unlike feedforward neural networks, which process independent inputs, RNNs incorporate directed cycles in their connections that enable the maintenance of a hidden

state. This hidden state serves as a dynamic memory, allowing the network to capture temporal dependencies by integrating information from previous inputs as a feedback mechanism. At each time step t, the hidden state h_t is updated using signal x_t and the previous state h_{t-1} as inputs such as:

$$h_t = \tanh(W_x x_t + W_h h_{t-1} + b)$$

Here, W_x , W_h weights are the trainable parameters multiplied by the input parameters, while b is a trainable constant. The output y_t at the time step t can be derived as:

$$y_t = W_y h_t + b_y$$

Here, W_y and b_y are the output weight and constant trainable parameter, respectively. This feedback mechanism makes RNNs suitable for time-series analysis tasks where temporal dependencies in the data are crucial for predicting the output. RNNs can be used for speech recognition, sequential pattern detection, weather forecasting and other signal processing, denoising and classification (Elman, 1990).

Standard RNN units (**Figure 11b**) are limited by the vanishing gradient problem during training. This problem severely restricts the neural network's ability to learn long-term dependencies in sequential data (Bengio et al., 1994). At this point, the LSTM architecture is used to solve the long dependencies and lack of memory issues (Hochreiter & Schmidhuber, 1997). The LSTM unit extends the RNN unit by introducing memory cells and gating mechanisms that regulate storage, updates, and the exposure of information across time steps (**Figure 11b, c**). LSTM introduces the cell gate C_t, the forget gate f_t, the input gate i_t, o_t, & $\widetilde{C_t}$, as well as the output gate h_t. Each gate has its own parameter and equation (**Figure 11c**), and combining all of them produce a hidden state h_t at time t, such as:

$$C_t = f_t \odot C_{t-1} + i_t \odot \widetilde{C_t}$$

$$h_t = o_t \odot \tanh(C_t)$$

Here the $\odot$ operator represents the Hadamard product (element-wise product of matrices).

Through this gating mechanism, LSTMs can selectively retain or discard information, thereby overcoming the limitations of conventional RNNs and enabling robust modeling of long-range temporal dependencies. In this work, the LSTM was implemented within the IVEA framework to classify one-dimensional feature vectors in the stationary burst events analysis module, where effective recognition of temporal activity patterns was essential.

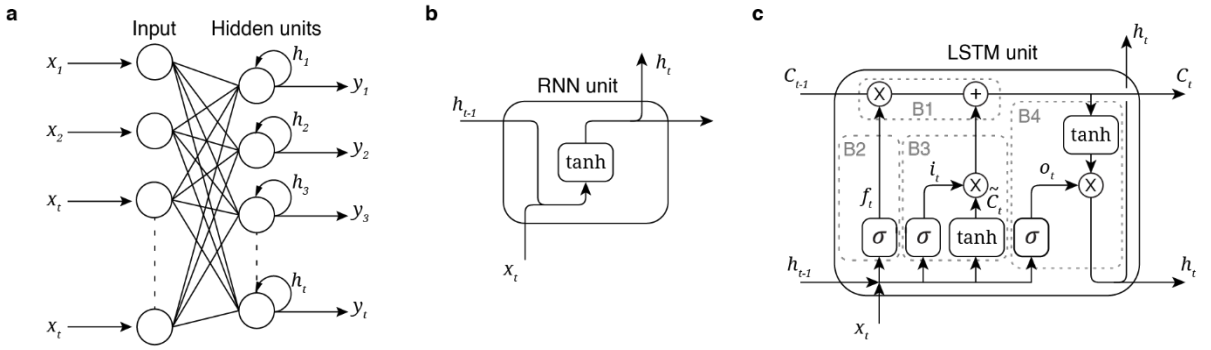


Figure 11. Comparison of feedforward, recurrent, and LSTM units.
These panels illustrate the evolution from static feedforward processing to gated architectures capable of retaining information over long sequences. **a**. Represents a simple feedforward network with multiple input features $(x_1, x_2, \dots, x_t)$ connected to hidden units with the feedback mechanism $(h_1, h_2, \dots, h_t)$. The hidden state h_t is the visible output of the block, which can also be projected to a task-specific output through an output layer y_t. **b**. A simple recurrent neural network (RNN) unit. At each time step, the hidden state h_t is updated as a nonlinear transformation of the current input x_t and the previous hidden state h_{t-1}. The hidden state serves both as the internal memory passed forward in time and as the block's output. **c**. A long short-term memory (LSTM) unit. In addition to the hidden state h_t, LSTM maintains a cell state C_t that carries long-term memory, which is called the cell gate located at block B1. Gating mechanisms (forget B2, input B3, and output gates B4) control which information is written, erased, or exposed.

2.15 Activation functions

Activation functions are mathematical functions that control the output of neural network nodes. They introduce non-linearity into the network, enabling it to learn and process complex features and patterns in the data. Without the activation functions, the neural network is a single linear transformation limited to solving only linear problems. Activation functions are crucial components of neural networks, determining whether a neuron should be activated (fire) or not based on its input values. This project employs different types of activation functions: The first one is the most common activation function, the Rectified Linear Unit (ReLU), expressed as $f(x) = \max(0, x)$. This function outputs the input directly if it is positive; otherwise, it returns zero. Although ReLU introduces non-linearity and is computationally efficient, it can hinder training if many neurons output zero consistently. ReLU is predominantly associated in convolution layers. The second activation function used is the Gaussian Error Linear Unit (GeLU). GeLU is an advanced activation function that incorporates properties of a Gaussian distribution. This function provides a smoother variant of ReLU, by incorporating small negative output and is expressed as:

$$f(x) = \frac{1}{2}\left[1 + \text{erf}\left(\frac{x}{\sqrt{2}}\right)\right]$$

Where the erf is the error function of the Gaussian distribution expressed as:

$$\text{erf} = \frac{2}{\sqrt{\pi}} \int_0^{\pi} e^{-t^2} dt$$

The third activation function is the softmax activation, which is utilized in the output layer of multiclass classification models. The softmax activation function is used to convert the raw, unnormalized scores (logit) output by the FCN into probabilities of a summation of a maximum value of 1. The softmax function is applied on the dense layer, which is the last layer for prediction, expressed as:

$$h(z) = \frac{e^{z_i}}{\sum_{j=1}^{n_c} e^{z_j}}$$

where $h(z)$ is the softmax function, z_i represents the logit for a specific class i and n_c is the number of classes. Hyperbolic tangent (Tanh) is another activation function that is often used in the LSTM network blocks. The Tanh function maps the input values to the range of [-1,1] and is expressed as:

$$\tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$$

2.16 Vision transformer network (ViT)

The transformer network represents a novel approach to dominant sequence transduction models, initially proposed in 2017 with the introduction of the self-attention mechanism (Vaswani et al., 2017). The self-attention network layer weighs the importance of different parts of the input data. Due to the attention mechanism, the neural network became highly capable of processing long text, which led to the development of language models such as Grok, ChatGPT, and others. Unlike RNNs, which process data in a sequential manner, transformers can process all data simultaneously. The parallelization in Transformers enables high computational speed and efficiency. However, this approach results in the loss of the sequence order of the data.

To convey the sequential order of the data, a positional encoding layer is employed to provide the model with positional information (Vaswani et al., 2017). For each token, a position-dependent signal is incorporated into each image embedding for an input sequence (Vaswani et al., 2017). An image embedding refers to the vector representation of an image within the neural network. A wide variety of positional encoding techniques exists. The commonly used positional encoding methods employ sine and cosine functions. These functions are of different frequencies, which generate a unique positional vector for each image patch (Y. Li et al., 2023). The output of the positional encoder layer has the same dimensionality as that of the embedding layer. For image analysis, this implies that each image embedding is encoded with a unique positional encoding, which represents the image position **Table 1**. Image embedding refers to the vector representation of a single image for the neural network. For a given position pos, dimension i, and the embedding dimension of the positional encoder layer d_{model}, the positional encoder for even indices in the given sequence is expressed as:

$$PE(pos, 2i) = \sin\left(\frac{pos}{10000^{\frac{2i}{d_{model}}}}\right)$$

Where for odd indices in the sequence:

$$PE(pos, 2i+1) = \cos\left(\frac{pos}{10000^{\frac{2i}{d_{model}}}}\right)$$

Table 1 Embedding vectors.
Alternating sin and cos example table for positional encoder vectors of embedding of even dimension n.The aforementioned example table demonstrates how sinusoidal positional encoding is applied across different positions and embedding dimensions. Each position t is encoded into a vector of length d_{model}, alternating between sine and cosine functions of varying frequencies, as $f_i = \frac{1}{10000^{\frac{2i}{d_{model}}}}$

Position	dim 0	dim 1	dim 2	dim 3	dim 4	... dim n
t=0	$\sin(0)$	$\cos(0)$	$\sin(0)$	$\cos(0)$	$\sin(0)$	… $\cos(0)$
t=1	$\sin(f_0 \cdot 1)$	$\cos(f_0 \cdot 1)$	$\sin(f_1 \cdot 1)$	$\cos(f_1 \cdot 1)$	$\sin(f_2 \cdot 1)$	… $\cos\left(f_{\frac{n}{2}} \cdot t\right)$
t=2	$\sin(f_0 \cdot 2)$	$\cos(f_0 \cdot 2)$	$\sin(f_1 \cdot 2)$	$\cos(f_1 \cdot 2)$	$\sin(f_2 \cdot 2)$	… $\cos\left(f_{\frac{n}{2}} \cdot t\right)$

Following the implementation of positional encoding in the transformer model, a novel mechanism, designated as the attention mechanism, is introduced (Vaswani et al., 2017). Such technology allows the model to get attention and learn from the essential features of the input sequence. In transformer architecture, the attention mechanism allows the model to assign higher weights to contextually critical tokens. For instance, negations in language or salient regions in images. Although two sentences may be lexically similar, their semantic representations in embedding space can diverge significantly due to a single token.

consider the sentences:

1. " My cat, Winnie Dixie, does like the new food we brought him."

and

2. "My cat, Winnie Dixie, doesn't like the new food we brought him."

In this case, without the attention mechanism, these two input sentences would yield embeddings that are closely aligned, as most tokens are identical. However, the attention mechanism amplifies the contribution of the token "doesn't", producing a semantic shift in the representation.

In the embedding space, the principal content-related dimensions, such as those capturing the topic (e.g., cat and food), remain similar, whereas a shift occurs along an orthogonal semantic axis representing sentiment polarity (**Figure 12**). The introduction of "doesn't" generates a displacement, or negation vector, that alters the overall embedding's orientation, leading the transformer model to treat the two sentences as distinct.

This phenomenon is not limited to text; in vision models, a salient feature (e.g., a small but significant object or region) can induce an analogous directional shift in the feature space, thereby altering the model's predictions despite global input similarity.

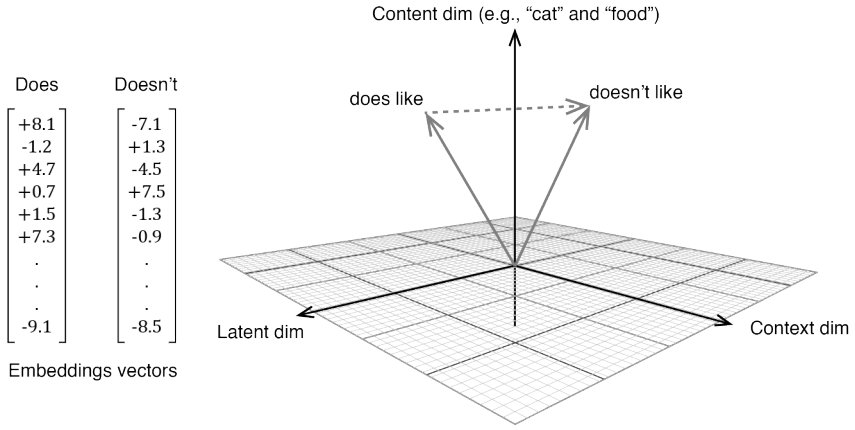


Figure 12. Simplified illustration of embedding space shifts caused by negation to the attention mechanism. The two columns on the left show dummy example embedding vectors for two nearly identical sentences. Most dimensions remain similar due to shared content, but the negation token "doesn't" shifts the embedding direction. The plot shows this shift in a reduced semantic space: the horizontal plane (latent and context dimensions) captures general features, while the vertical axis (content dimension) reflects shared subject matter (e.g., "cat" and "food"). Grey and dashed arrows represent "does like" and "doesn't like," respectively, with their angular difference illustrating the semantic change emphasized by the attention mechanism.

2.17 Gaussian non-maximum suppression techniques

Traditional non-maximum suppression (NMS) approaches, such as intersection over union (IoU)-based suppression, weighted boxes fusion (WBF), and related methods (Solovyev et al., 2021) are designed for object detection tasks where targets are represented by discrete spatial boundaries (e.g., bounding boxes or segmentation masks). Such techniques are unsuitable for the current application, as exocytotic events cannot be meaningfully represented by fixed boundaries. Instead, each event corresponds to a continuous spatiotemporal fluorescence transient exhibiting spatial diffusion and temporal evolution rather than a sharply delimited object.

Alternative strategies, including radius-based searches, clustering algorithms such as DBSCAN or Mean Shift, and adaptive neighborhood criteria, were considered. However, these methods rely on discrete point sets and local neighborhood definitions, which are not directly compatible with the continuous, intensity-weighted representation of fluorescence events in IVEA (see discussion section).

Therefore, to resolve redundant detections, a novel algorithm called the Gaussian non-maximum suppression (GNMS) was developed. This method models each event as a 3D Gaussian distribution in a continuous spatiotemporal space (Chouaib et al., 2025). The GNMS field g is evaluated in (x, y, t) around each detected event center (**Figure 13**), defined as:

$$g(x,y,t) = \Delta\mu \cdot e^{-\frac{(\Delta x^2+\Delta y^2)}{2(\sigma+SR)^2}} \cdot e^{-\left(\frac{\Delta t^2}{2\tau^2}\right)} \quad \text{with } \tau = \nu \cdot \sigma$$

where $\Delta\mu$ is the local intensity change measured on the background-subtracted images at the event center, σ is the average spatial spread of pixels around an event, SR is the user-controlled spread radius (default = 0), which can be adjusted to accompany the vesicle exocytosis spreading size, and τ is the temporal spread. The acquisition frequency provides the temporal scale used for τ, which is set to 10 Hz. For two events E_i at (x_i, y_i, t_i) and E_j at (x_j, y_j, t_j), the GNMS value induced by E_j at the coordinates of E_i is obtained by substituting $\Delta x = x_i - x_j$, $\Delta y = y_i - y_j$, and $\Delta t = t_i - t_j$ into g_j. Event E_i is suppressed by E_j if its local change is smaller than the GNMS field contributed by E_j at (x_j, y_j, t_j), such as:

$$\begin{cases} E_i = E_j & \text{if } \Delta\mu_i < g_j(x_j, y_j, t_j) \\ E_i \neq E_j & \text{otherwise} \end{cases}$$

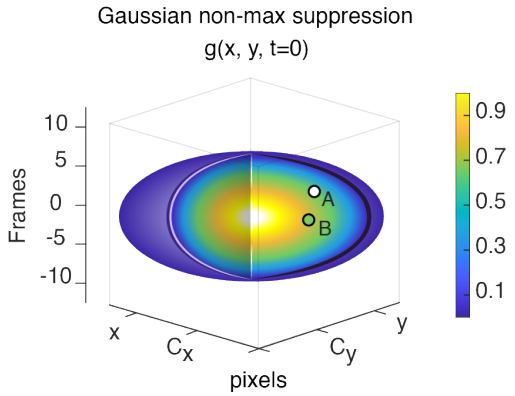


Figure 13. Gaussian non-maximum suppression algorithm in spatiotemporal coordinates (adapted from (Chouaib et al., 2025)).
The ellipsoidal envelope visualizes the GNMS field $g(x, y, t)$ centered on a detected event, with color indicating the field magnitude (normalized). Two nearby events, A and, B are illustrated. The GNMS value generated by A and evaluated at the coordinates of B, $g_A(x_B, y_B, t_B)$, reflects the expected contribution of A at B location after accounting for spatial and temporal separation via $(\sigma + SR)$ and τ. If the local change of$B$, $\Delta\mu_B$ (measured on the background-subtracted images), is smaller than $g_A(x_B, y_B, t_B)$, event B is suppressed as a redundant detection; otherwise, it is retained as a new event. The reciprocal test can be applied for A as well. In this manner, redundant detections within a shared spatiotemporal neighborhood are removed while the most significant event is preserved.

This criterion results in the removal of a weaker detection falling within the spatiotemporal "cloud" of a stronger neighbor, while a sufficiently strong and/or adequately separated detection is retained (**Figure 13**). The same test can be applied symmetrically, ensuring that only the most significant representative of a local spatiotemporal cluster is preserved. This continuous, ellipsoidal suppression in (x, y, t) avoids the limitations of box-based IoU/WBF methods and is tailored to transient, diffusely localized biological events.

2.18 Stationary and random burst events algorithm

The IVEA platform processes fluorescence burst events through two major components: first, detection, which involves automated identification of candidate regions of interest ROIs; and second,

classification, which uses two different neural network models, either the LSTM or the eViT (**Figure 14, Figure 15**).

The first step in the detection process is to scan the image sequence for abrupt changes in intensity. This is achieved by examining short temporal segments, typically four consecutive frames, to generate two complementary difference images. The first is a forward difference, $\Delta I_F = I_{i+n} - I_i$, which highlights increases in intensity and provides information about signal presence. The second is the backward difference, $\Delta I_B = I_i - I_{i+n}$, which captures decreases in intensity. This is particularly useful for the random burst event module, when pH-insensitive reporters are used to visualize fusion events, as they often exhibit sudden signal loss. After generating these difference images, the system automatically determines appropriate detection parameters. Rather than relying on user-defined thresholds, IVEA estimates baseline noise behavior from the earliest frames of the recording, under the assumption that the first four frames contain no fusion events. These frames serve as a reference for evaluating noise-driven peaks and estimating the expected peak prominence in the absence of exocytosis.

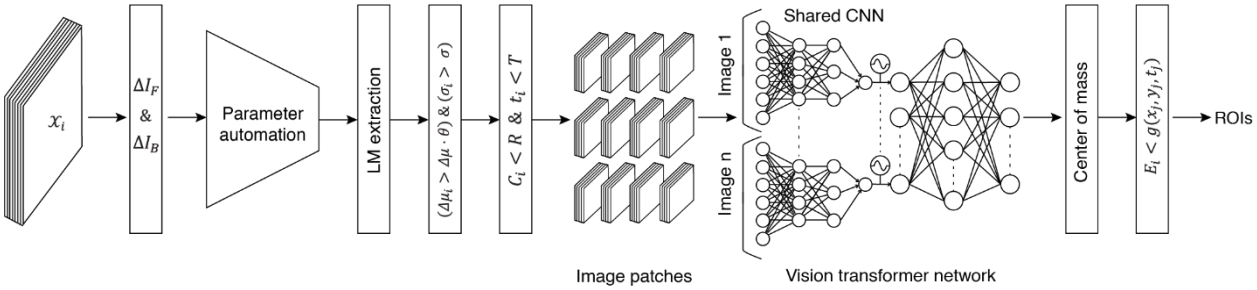


Figure 14. Algorithm flowchart for the random burst events (adapted from (Chouaib et al., 2025)).
The input $\mathcal{X}_i$ represents the image sequence. ΔI_F and ΔI_B are the forward and backward difference images derived from ΔI. Key variables derived from ΔI includes the mean intensity $\Delta\mu_i$, full-width at half-maximum σ_i, center coordinates C_i, and detection time t_i, each corresponding to a detected event E_i. Automated detection parameters $\Delta\mu$ and σ, are used for thresholding, while additional parameters θ, R (search radius = 3 pixels), and T (time interval = 4 frames) define the sensitivity and spatiotemporal constraints. Image patches of size 32×32 pixels are extracted for each region of interest over a specific time interval. The encoder-vision transformer network processes these patches. The center-of-mass step is used to adjust for the true event's center of mass. The final stage applies a non-maximum suppression to eliminate duplicates and retain the most significant events, utilizing a continuous spatiotemporal Gaussian function g(x, y, t).

The automation algorithm incrementally adjusts a prominence value $\mathcal{p}$ while repeatedly searching for local maxima (LMs). The initial value of $\mathcal{p}$ set to 10, and each iteration increases by 10 (i.e., $\mathcal{p}_n = \mathcal{p}_{n-1} + 10$). Ideally, in the earliest frames, where no events are expected, $\mathcal{p}$ is determined under the assumption that no local maxima should be present. For that reason, at each new prominence level, the local maxima are extracted to examine whether increasing $\mathcal{p}$ yields no detectable maxima (i.e., $l_n = 0$), or whether the number of maxima stabilizes to the same count observed four iterations earlier (i.e., $l_n = l_{n-4}$). In either case, the current value of $\mathcal{p}$ is treated as the optimal threshold. This adaptive procedure ensures that the peak detection remains tailored to the characteristics of each individual video. These reference frames are also used to estimate the typical noise width, expressed as the full width at half maximum (FWHM) σ, and to compute the average noise-related intensity change $\Delta\mu$. The latter is

obtained by measuring the mean intensity change $\Delta\mu_j$ around each noise LM at coordinates C_j within a radius r, and then averaging across all detected maxima, expressed as:

$$\Delta\mu_j = \frac{1}{4r^2+1} \sum_{c_j^{(x)}-r}^{c_j^{(x)}+r} \sum_{c_j^{(y)}-r}^{c_j^{(y)}+r} \Delta I_i(x_j, y_j) \quad \text{with } r \in \mathbb{N}$$

The region of interest nomination process follows the same procedure as parameter automation. For each event E_j at the center coordinates C_j, we determine $\Delta\mu_j$ and σ_j. To designate E_j as a selected region, we apply the following condition:

$$E_j \mid \left(\Delta\mu_j > \Delta\mu \cdot \theta\right) \wedge \left(\sigma_j > \sigma\right) \quad \text{with } \sigma, \theta \in \mathbb{R}$$

where, θ represents the sensitivity parameter, allowing user adjustment to refine detection performance.

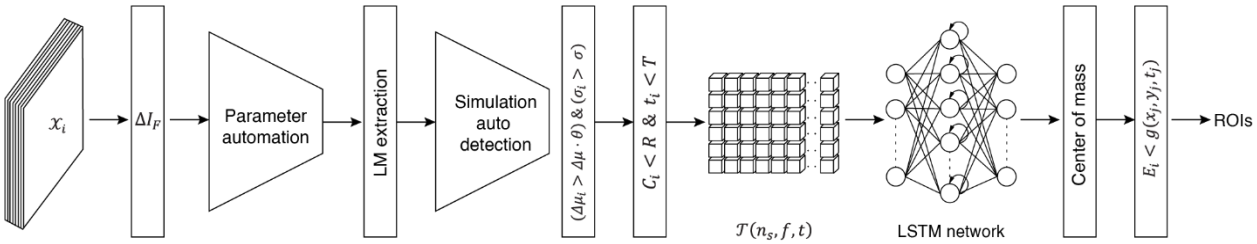


Figure 15. Algorithm flowchart for the stationary burst events (adapted from (Chouaib et al., 2025)).
Flowchart of the stationary burst events algorithm. Key parameters are defined as follows: $\mathcal{X}_i$ represents the raw image frames, while ΔI_F corresponds to the forward-difference image, and $\Delta\mu_i$, σ_i, C_i, and t_i, are the mean gray intensity, full width at half maxima, the spatial coordinates of the event center, and the temporal index for each event E_i, respectively. Threshold parameters $\Delta\mu$ and σ denote the mean gray value and FWHM thresholds. θ, T, R and indicates the detection sensitivity, the event temporal interval, and the search radius, respectively. Extracted data is represented as a 3D tensor $\mathcal{T}(n_s, f, t)$.

After the detection and nomination stage, IVEA executes spatiotemporal tracking across a defined radius and time window for each selected region. The stationary burst module is primarily designed to identify synaptic transmission events in neurons. To extend its functionality, an additional step was developed to handle stimulation-triggered activity at the detection step, including agonist/electric and NH_4^+-induced responses. This step distinguishes and organizes events based on their temporal progression, using a ratio of mean intensity between successive frames, defined as:

$$\mathcal{R}_i = \frac{\Delta\mu_{i+1}}{\Delta\mu_i} > \theta_s$$

where $\mathcal{R}_i$ is the intensity ratio between consecutive images, θ_s is the threshold ratio (default = 1.1), $\Delta\mu_i$ and $\Delta\mu_{i+1}$ are the mean gray values of the difference images ΔI_i and ΔI_{i+1}, respectively. To mitigate over-detection caused by elevated fluorescence during stimulation, users can compensate by scaling the detection threshold dynamically, defined as $\theta_i = \theta \cdot \mathcal{R}_i$, where θ is the baseline sensitivity parameter.

2.19 Feature extraction for LSTM

Spatiotemporal feature extraction was performed by selecting sequences of image patches centered at C_j over a defined temporal range. Each patch was subdivided into multiple subregions, and the mean pixel intensity of each subregion was computed (**Figure 6a**). For each frame c_t, the spatial neighborhood around $C_j$corresponding to the event $E_j$was extracted as a patch $\mathcal{M}_t^j$:

$$\mathbf{M}_t^j(x,y) = \begin{cases} I_{c_t}(x,y), & 1 \le x \le W,\ 1 \le y \le H, \\ 0, & \text{otherwise,} \end{cases}$$

where I_{c_t} is the image plane at frame c_t and W, H are the image dimensions. Pixels outside the image were assigned as zero values. The 2D matrix is of size k as $\mathcal{M}^j \in \mathbb{R}^{k\times k}$, where k is denotes the user-defined kernel size. If $k \neq 13$, the extracted patch at time t was bilinearly interpolated into square matrix $k = 13$ pixels:

$$\widetilde{\mathbf{M}}_t^j = \text{Bilinear}\big(\mathbf{M}_t^j, k, k\big), \qquad \text{j, t} \in \mathbb{N}$$

The spatiotemporal representation $\mathcal{V}_j \in \mathbb{R}^{k\times k\times T}$, describes the extracted data evolution of event E_j over T frames surrounding the event time t_j, as expressed in:

$$\mathcal{V}_j(x,y,t) \coloneqq \left\{\widetilde{\mathbf{M}}_t^j(x,y) \;\middle|\; t \in \left[t_j - n_b, t_j + n_a\right]\right\}$$

Where n_b and n_a correspond to the number of frames preceding and following t_j, respectively. Each matrix $\widetilde{\mathbf{M}}^j$ was divided into discrete non-overlapping regions $\{\Omega_r\}_{r=1}^{\text{ϝ}}$, where the index r denotes the current spatial region and ϝ = 13 is the total number of regions. The per-region feature at time t is the mean pixel intensity within region r (**Figure** 6**a**).

$$v_{t,r} = \frac{1}{|\Omega_r|} \sum_{(x,y)\in\Omega_r} \widetilde{\mathbf{M}}_t^j(x,y)$$

Where $v_{t,r}$ is the scalar intensity value for region r at time t. The row vector $\mathbf{v}_\text{t} = \left[v_{t,1},\ v_{t,2}, \dots, v_{t,\text{ϝ}}\right]$ which contains the feature for all ϝ regions at t. Stacking all time steps produces the event-specific features matrix $\mathbf{P}_j \in \mathbb{R}^{T\times\text{ϝ}}$ for event E_j such as:

$$\mathbf{P}_j(X(t), \text{ϝ}) = \begin{bmatrix} \mathbf{v}_0^\top \\ \mathbf{v}_1^\top \\ \vdots \\ \mathbf{v}_{T'-1}^\top \end{bmatrix} \in \mathbb{R}^{T'\times f}.$$

Here, n_f denotes the number of pixels within each region f. This process yields a dataset $\mathbf{P} \in \mathbb{R}^{n_s\times T\times\text{ϝ}}$, where n_s is the total number of detected events. Each $\mathbf{P}_j$ contains 13 temporal intensity profiles corresponding to distinct spatial segments (**Figure** 6**d**). The use of symmetrical subregions instead of circular masks improves the retention of spatial detail. In fact, the circular mask demonstrated lower sensitivity to motion because the mean intensity across each ring averages the signals from pixels on opposite sides of the granule's path (**Figure 16**). This spatial averaging dampened the apparent movement signal when the granule traveled in a single direction, as motion in one region was counterbalanced by static or opposing pixels in the same ring. The 13-region mask was ultimately selected over the 9-region mask for its superior sensitivity in detecting slow granule movements (**Figure 16a, b**).

When the temporal extent was increased by t_n, such as $X'(t) \in \mathbb{R}^{t}$ with $t = T + t_n$, the features matrix $\mathbf{P}(X(t), \dagger) \in \mathbb{R}^{T \times \dagger}$ was downsampled using a sliding mean filter:

$$X(t)_i = \frac{1}{w} \sum_{k=1}^{w} X'(t)_{w(i-1)+k} \quad w = \frac{t}{T} \text{ with } w \in \mathbb{N}$$

Where, $X(t)_i$ represents the $i\text{-}th$ sample of the downsampled sequence, and $X'(t)_j$ the corresponding element from the original extended sequence, with $j = w(i-1) + k$.

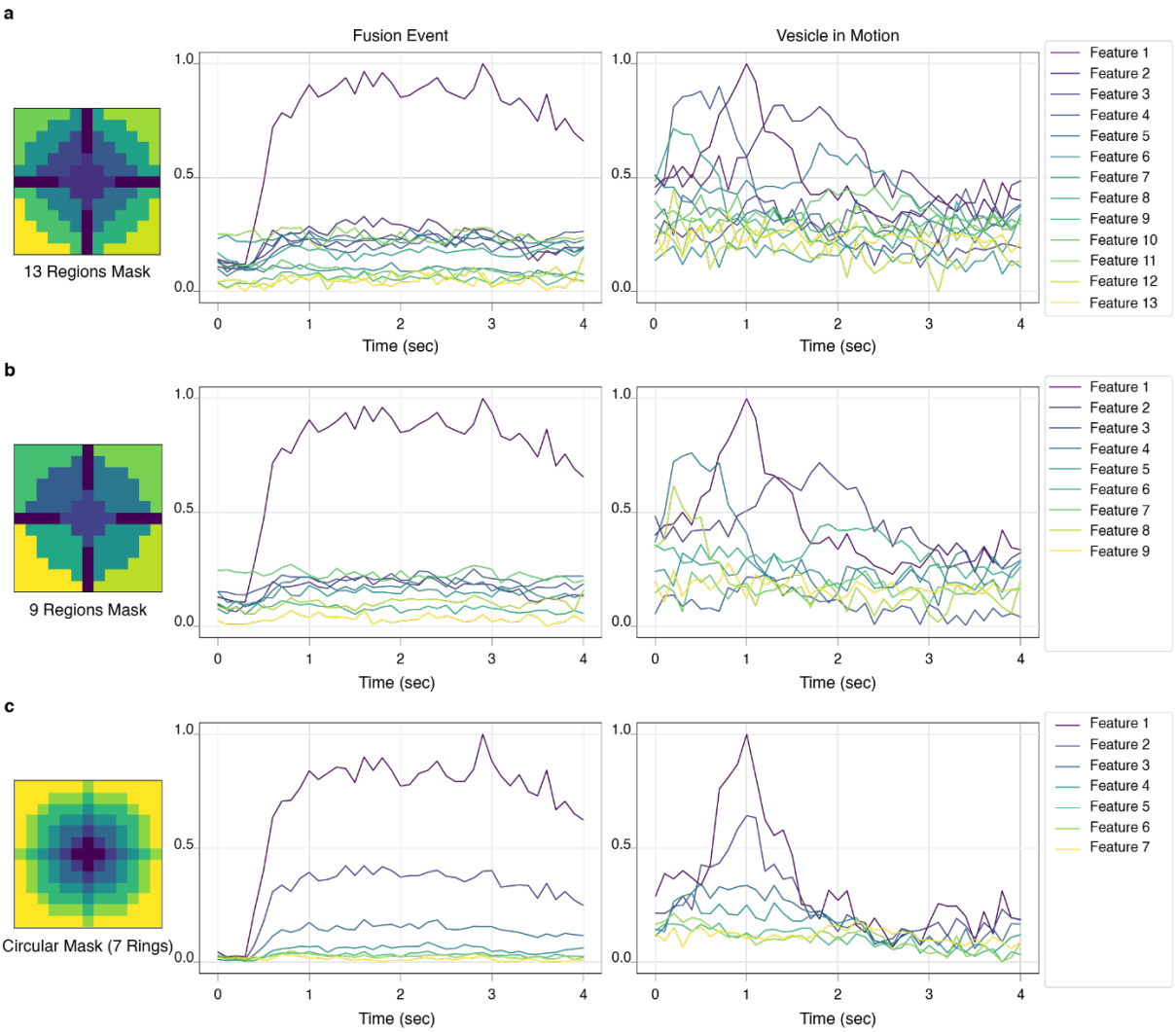


Figure 16. LSTM feature extraction using different labeled masks influences the extraction of time-series features for LSTM-based event classification (adapted from (Chouaib et al., 2025)).

The first column displays the spatial mask layout applied to an event, the second column shows the resulting time-series signals from a representative fusion event, and the third column presents signals from a vesicle movement event. Each row in these columns corresponds to a distinct mask layout. **a**. Default mask of 13 asymmetrically defined regions generated to be imported to the LSTM. **b**. Reduced configuration with 9 regions to assess the effect of fewer spatial partitions on feature capture. **c**. Circular mask comprising 7 concentric rings, offering a symmetrical alternative to region-based segmentation. Different mask layouts influence sensitivity to spatial detail and motion, with the 13-region configuration providing higher sensitivity to movements, while the circular mask is the least sensitive to motion.

If the extraction window extended beyond the last frame, missing steps were filled with the running mean of previous valid features:

$$\hat{\mathbf{v}}_t = \frac{1}{N_t}\sum_{\tau\in\mathcal{O}_t}\mathbf{v}_\tau \quad \mathcal{O}_t = \{\tau \le t : \mathbf{v}_\tau \; observed\},\; N_t = |\mathcal{O}_t|$$

Where $\mathcal{O}_t$ denotes the set of observed time indices up to step t. While processing $\hat{\mathbf{v}}_t$ vector, the index of the maximum value is stored to adjust for the time shift Δt related to the peak intensity for the event E_j. The time shift is used to correct the detection time t of E_j by adding Δt to t expressed as:

$$t' = t + \left(\frac{[T-1]}{2} + 1\right) - \arg\max_t \hat{\mathbf{v}}_t$$

Where t' is the new peak intensity for the event E_j. The new time value t' is exclusively employed for the adjustment of event coordinates during the measurement process and not for the data extraction process. Normalization was applied to the flattened ϱ × T feature vector, which is the 1D serialized form of $\mathbf{P}_j$ obtained by concatenating all $v_{t,r}$ values in time-major order. Thus, $V_x[j]$ corresponds to one element of $v_{t,r}$ after flattening. Values were clipped to 0.01:

$$\boldsymbol{V}_x[j] = \begin{cases} 0.01, & \text{if NaN or } \boldsymbol{V}_x[j] < 0.01, \\ \dfrac{\boldsymbol{V}_x[j] - min(\boldsymbol{V}_x)}{max(\boldsymbol{V}_x) - min(\boldsymbol{V}_x)}, & \text{otherwise} \end{cases}$$

For stationary bursts, $\mathbf{P}_j$ was reshaped to $\mathcal{T}_j \in \mathbb{R}^{ϱ\times T}$ and batched into $X \in \mathbb{R}^{n_s\times ϱ\times T}$. Axis ordering was selected based on the model objective: temporal-first $(n_s, T, ϱ)$ for learning temporal dynamics, or feature-first $(n_s, ϱ, T)$ for spatial pattern emphasis. Stationary bursts used $n_b = 10, n_a = 30, T = 41$ and random bursts $n_b = n_a = 10, T = 21$, with $ϱ = 13$ in both cases.

In stationary burst event analysis, $\mathbf{P}$ is reshaped into a 3D tensor $\mathcal{T} \in \mathbb{R}^{n_s\times ϱ\times T}$ as follows:

$$\mathbf{P}(j, X(t), ϱ) \rightarrow \mathcal{T}\big(j, ϱ, X(t)\big) \mid j \in [1, n_s] \;\text{ with } n_s, j \in \mathbb{N}$$

As for random burst events analysis, the tensor dimensions were reordered to $\mathcal{T}(j, X(t), ϱ)$ to prioritize the temporal dimension before the spatial segmentation index, expressed as:

$$\mathbf{P}\big(j, ϱ, X(t)\big) \rightarrow \mathcal{T}(j, X(t), ϱ) \mid j \in [1, n_s] \;\text{ with } n_s, j \in \mathbb{N}$$

Reordering $X(t)$ and ϱ determines which axis the LSTM treats as the recurrent time dimension. With input ordered as $(samples, ϱ, X(t))$, the network unfolds across spatial regions, treating the full-time vector as the feature set for each step. This makes it sensitive to the absolute temporal positions of patterns, which is advantageous for stationary bursts that occur at fixed positions for a long time. Conversely, ordering as $(samples, ϱ, X(t))$ unfolds across time, learning temporal dynamics directly.

This arrangement allows pattern recognition independent of their temporal position, which is advantageous for random burst events, as detections across multiple positions increase capture probability. Duplicate detections are then removed using the Gaussian non-maximum suppression method.

2.20 Multivariate LSTM neural network architecture

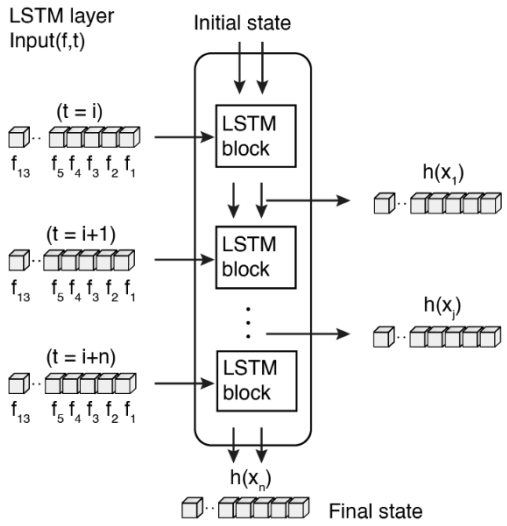


Figure 17. LSTM block architecture for sequential feature processing.
The input data consists of a one-dimensional vector with 13 features (f) representing characteristics of each event E_i. These features are fed into the LSTM model as a sequence over time (t), where each time step corresponds to a single feature vector. The sequence begins at time t and progresses through t=$i + n$, where $i + n$ represents the final time step in the sequence. Each LSTM block processes sequential data, capturing temporal dependencies and extracting meaningful spatiotemporal representations. The output at each block is passed to subsequent blocks or serves as the final output for the model, providing a comprehensive feature representation across the entire time series.

The LSTM network implemented in IVEA is designed as a four-layer architecture for the analysis of multivariate time-series data (**Figure 6c**). The initial stage involves specifying the input tensor shape, which defines the expected dimensions for the first layer to receive thirteen time-series data vectors (**Figure 17**). Although this is not a computational layer, this specification step ensures that the network is properly matched to the input format and feature dimensions.

The first computational layer is a one-dimensional convolutional layer (Conv1D) with a ReLU activation function. This layer extracts local patterns from the temporal sequences, enabling the network to detect short-term dependencies and localized variations within each feature channel.

Following the convolutional stage, an LSTM layer is employed to capture longer-range temporal dependencies and sequential patterns. This recurrent layer allows the model to retain relevant contextual information over time, which is critical for representing the temporal dynamics of exocytotic events.

To stabilize training and mitigate internal covariate shift, a batch normalization layer is applied after the recurrent stage, with normalization performed along the first non-batch axis (axis=1). This supports more consistent gradient propagation through the network and enhances its training performance.

The final stage of the network is utilized with a fully connected Dense layer with a softmax activation function, which produces probability distributions over the output classes. This configuration is used for multiclass classification tasks, a necessity for our data labeled with multiple categories.

The network is optimized using the Adam optimizer with a learning rate of 10^{-3} and trained with categorical cross-entropy loss $\mathcal{L}$, defined as:

$$\mathcal{L} = \frac{1}{n_s} \sum_{j=1}^{n_s} L\left(\hat{y}^{(j)}\right)$$

$$L\left(\hat{y}^{(j)}\right) = -\sum_{i=1}^{n_c} y_i^{(j)} \cdot \log\left(\hat{y}_i^{(j)}\right)$$

Where $L\left(\hat{y}^{(j)}\right)$ represents the loss for a single sample, n_s is the number of samples, n_c is the number of classes, $y_i^{(j)} \in \{0,1\}$ is the ground-truth label for class i, and $\hat{y}_i^{(j)}$ is the predicted probability for the class i, obtained from the softmax output $\hat{y} = h(z)$. This architectural design combines convolutional feature extraction, recurrent sequence modeling, and normalization, enabling the network to simultaneously exploit both local and long-term temporal patterns inherent in multivariate sequential data for the 13 segmented region mask.

2.21 Encoder-ViT network architecture

The encoder-Vision Transformer network consists of two main components: a shared convolutional neural network (encoder) for feature extraction from image patches, and a transformer-based classification module. The shared convolutional network contains seven layers (**Figure 18a**), beginning with a two-dimensional spatial convolution layer, followed by a sequence of three-dimensional convolution (**Figure 18b**) and max-pooling layers.

Two pretrained encoder-ViT models, GranuVision2 and GranuVision3, are used for random burst event classification, while a single model, NeuroVision1, is employed for stationary burst event classification. The encoder input layer accepts sequences of 26, 28, or 40 image patches for GranuVision2, GranuVision3, and NeuroVision1, respectively. Each patch has dimensions of 32×32×1, with the last dimension corresponding to the number of channels, expressed as the dimension of the input layer shape $\mathcal{X} \in \mathbb{R}^{t \times w \times h \times c}$. When user-defined patches differ in size, all patches are resized to match the encoder's expected dimensions using bilinear interpolation. This method was selected as a practical balance between computational efficiency and image quality. It avoids the pixelation artifacts of the nearest-neighbor interpolation method. Compared to bicubic interpolation, it is less computationally

demanding, although the latter produces smoother results. Given the relatively small patch size of 32x32 pixels, bilinear interpolation provides sufficient visual fidelity at a moderate computational cost. After resizing, normalization to the range [0, 1] is applied across each sample to ensure consistency in pixel intensity values across the dataset.

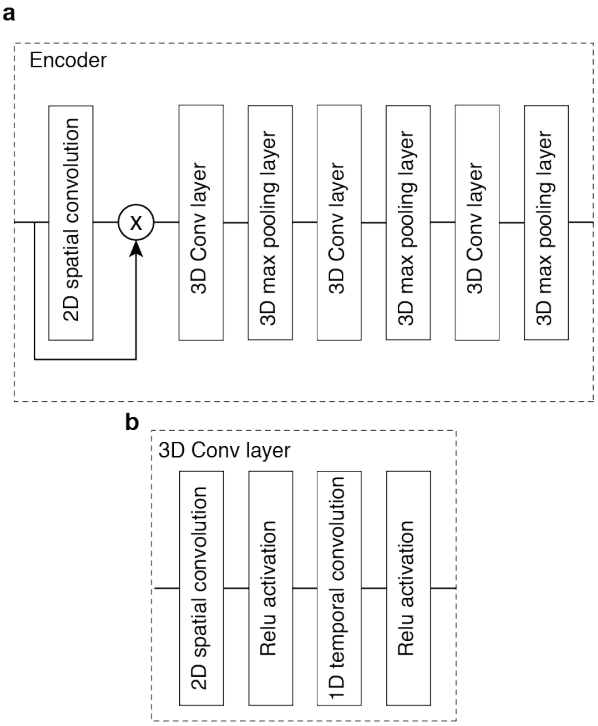


Figure 18. Encoder architecture (adapted from (Chouaib et al., 2025)).
a. The encoder begins with a two-dimensional spatial convolution layer, followed by a sequence of three-dimensional convolutional layers and three-dimensional max-pooling layers. b. Each three-dimensional convolutional block is factorized into separate spatial and temporal convolution stages. The spatial convolution operates independently on each temporal frame, followed by a temporal convolution applied along the time axis. Both stages employ rectified linear activation. The factorized design reduces parameter count, improves computational efficiency, and allows spatial and temporal dependencies to be modeled separately.

The initial layer of the shared convolutional network applies a two-dimensional spatial convolution followed by a multiplicative residual connection, enabling the model to integrate the input directly with residual features. This enhances the representation of pixel-intensity variations across heterogeneous samples. The network then applies a sequence of factorized three-dimensional convolutional blocks ($R(2+1)D$ design; (Tran et al., 2018)), separating spatial and temporal filtering to improve representational efficiency and reduce parameter coupling. The second layer performs a three-dimensional convolution with 32 filters and rectified linear activation, followed by a three-dimensional max-pooling layer to reduce spatial resolution. This is followed by a three-dimensional convolutional layer with 64 filters and rectified linear activation, again followed by max pooling. A third convolutional layer with 128 filters and rectified linear activation is then applied, followed by a final max-pooling layer.

After feature extraction, the output maps from each frame are flattened and projected through a fully connected layer of size 64. This converts each frame into a single feature token that represents its spatial information. The resulting sequence of T tokens encodes the temporal evolution of the region. A learnable one-dimensional positional embedding of length T is then added to the sequence to preserve the temporal ordering of frames before entering the transformer block. The transformer block comprises a multi-head self-attention mechanism followed by two Dense layers, with each preceded by a residual connection. The feed-forward module includes two fully connected layers: the first expands the key dimension to 2×64, and the second projects it back to 64, maintaining consistency with the embedding size.

Finally, a classification multilayer perceptron processes the output. This subnetwork composed of two Dense layers, with the first employing the Gaussian error linear activation and the second using a softmax activation to produce probability distributions over the output categories. GranuVision2 predicts ten classes (two exocytosis types and eight artifact types), GranuVision3 predicts eleven classes (three exocytosis types and eight artifact types), and NeuroVision1 predicts ten classes (three exocytosis types and seven artifact/noise categories).

Each three-dimensional convolution operation is factorized into a two-dimensional spatial convolution followed by a one-dimensional temporal convolution, implemented via a custom Conv2Plus1D layer. The spatial component uses a kernel size of $(1, k_h, k_w)$, applying convolution only across spatial dimensions at each temporal step. This is followed by a temporal component with a kernel size $(k_t, 1, 1)$, applying convolution along the temporal dimension. Both operations are followed by rectified linear activation. This factorization reduces the number of parameters compared to unfactorized three-dimensional convolution, improves computational efficiency, and enables independent modeling of spatial and temporal dependencies.

The choice to decompose three-dimensional convolution into independent spatial and temporal operations was driven by both computational and modeling considerations specific to the nature of the data. In the present context, the spatial resolution of each patch is relatively small (32×32 pixels), and the temporal sequences are short. A conventional 3D convolution applied directly over these volumes would couple spatial and temporal filtering into a single operation, increasing the parameter count without necessarily leveraging the distinct roles of spatial structure and temporal evolution in the events of interest. By factorizing the convolution, spatial feature extraction is performed independently at each time step, enabling the network to focus on static or slowly varying spatial patterns, such as granule shape, size, and local texture, without interference from temporal variability. Temporal filtering is then applied separately, allowing the model to detect motion patterns, event onsets, and other dynamic cues in isolation from spatial complexity. This separation reduces redundancy in learned filters, improves interpretability of the intermediate representations, and lowers computational cost. The resulting

architecture can therefore model spatiotemporal dependencies more efficiently while remaining well-suited to the high-throughput, patch-based design of the system.

2.22 Neural network training

In IVEA, both the Long Short-Term Memory and encoder-Vision Transformer networks were developed and trained in Python, using Microsoft Visual Studio Code as the integrated development environment. Preparation and initial handling of training data for the recurrent model were performed in MATLAB to facilitate efficient visualization and analysis of segmented image patch patterns (**appx. Figure 44**), while training data for the transformer-based model were labeled in ImageJ.

Human experts conducted the initial data labeling and verified the classifications produced by IVEA, as detailed in (**Table 2**). For transformer model training, video files were paired with ROI files containing the coordinates of the ROI center, frame numbers, and radii. Within the IVEA ImageJ plugin, labeled ROIs could be exported as ZIP archives via the "Custom Models" interface. To clearly distinguish training data from evaluation sets, the software appended the suffix "_training_rois" to the exported files.

Table 2. Human experts' evaluation table. Movies were evaluated by the biologist who acquired the data and has experience in exocytotic events (adapted from (Chouaib et al., 2025)).

Figure	Authors	IVEA validation
Figure 24	O. Khamis	A. Chouaib, U. Becherer
Figure 24	H.-F. Chang	A. Chouaib
Figure 24	N. Alawar, U. Becherer	A. Chouaib
Figure 24	S. Hugo(Hugo et al., 2013), S. Echeverry, U. Becherer(Becherer et al., 2007)	A. Chouaib, U. Becherer, S. Echeverry
Figure 24	S. Echeverry	A. Chouaib, S. Echeverry, S. Barg
Figure 31	A.Shaib (Shaib et al., 2018), A. Staudt (Staudt et al., 2022)	A. Chouaib, U. Becherer
Figure 39	S. Elizarova	A. Chouaib
Figure 26	L. Demeersseman	A. Chouaib, L. Demeersseman
Figure 41	U. Becherer(Becherer et al., 2007)	A. Chouaib, U. Becherer
Figure 28	Q. Tian	A. Chouaib, Q. Tian
Figure 31	A. Staudt(Staudt et al., 2022), U Becherer	A. Chouaib, U. Becherer

Before the integration of neural network inference into IVEA, events identified by automated analysis required manual labeling after export. Once the neural network module was integrated, events in the training dataset were automatically labeled using a standardized convention incorporating the list index, event identifier, frame number, and class category (e.g., "1-event (1) | frame 851_class_0").

Training data preparation was carried out using a dedicated Python script with a configuration file in JSON format. This process included reading ImageJ ROI files to obtain event positions and class assignments, extracting temporal sequences of patches centered on each ROI, and storing the data in a hierarchical HDF5 structure containing "x_train" and "y_train" datasets for streamlined management.

Label refinement was iterative throughout the training process. Initial training distinguished only between “exocytosis” and “non-exocytosis” events. Subsequent iterations expanded classification to distinguish exocytosis subtypes and various artifact categories such as noise and motion-related artifacts. Positive integer labels were assigned to exocytosis events, while negative integers were used for non-exocytosis categories. Misclassified events were relabeled or assigned new categories, and the network was retrained. Given the large volume of predictions generated by IVEA, some non-exocytosis categories were removed to reduce dataset complexity. All training datasets and tools required for reproducing the models are publicly accessible at GitHub, https://github.com/AbedChouaib/IVEA.

For stationary burst event classification using the recurrent model, data were exported as two CSV files, one for the extracted data samples and the other for the data samples’ labels. The data was flattened and serialized to a 2D CSV file with a relatively small storage size (82 MB).

For each detected event E_j, the corresponding sample is represented as $P_j \in \mathbb{R}^{T\times f}$, where each entry $P_j[t,r] = v_{t,r}$ denotes the feature value v at time t and feature index r. The sample P_j is flattened into a one-dimensional vector, such as:

$$x_j \in \mathbb{R}^{fT}, x_j[t\cdot f + r] = P_j[t,r],$$

for $t = 0,\dots,T-1$and $r = 0,\dots,f-1$.

Stacking all n_s samples yield the data matrix $X \in \mathbb{R}^{n_s\times f\times T}$, while the corresponding class labels are stored as one-hot encoded rows in $Y \in \mathbb{R}^{n_s\times n_c}$, where n_s denotes the number of samples and n_c the number of classes.

For later data loading, the dataset is reshaped from

$$\mathbb{R}^{n_s\times(fT)} \rightarrow \mathbb{R}^{n_s\times f\times T},$$

and subsequently transposed from (n_s, T, f) to (n_s, f, T).

The recurrent model for stationary burst events was trained using approximately 11,300 samples derived from 39 movies, each sample having a dimension of 13 × 41. For random burst events, training involved 12,600 samples extracted from 548 movies; in this case, the data was kept untransposed, following the ordering (n_s, T, f).

The transformer-based model input is extracted and organized as a five-dimensional tensor $\mathcal{X} \in \mathbb{R}^{n_s\times t\times W\times H\times C}$, where n_s is the number of samples, t is the number of frames per sequence, W and H are the patch width and height (in pixels), and C is the number of channels. Corresponding labels are stored as one-hot encoded vectors $\mathcal{Y} \in \mathbb{R}^{n_s\times n_c}$ where n_c is the number of classes.

For each event, IVEA extracts a spatiotemporal block of dimensions (t, W, H) from the image stack. The spatial cross-section (W, H) is centered at the ROI coordinates (x_c, y_c, f_c) provided by the labeled ROI file. The temporal dimension t is defined as $t = t_b + t_a$, where t_b is the number of frames before the event frame f_c and t_a is the number of frames after. Each spatial frame is cropped to a square of size $2r \times 2r$, where r is the user-defined radius, and rescaled to a fixed radius $2R \times 2R$, where R is the encoder input dimension, adjustable by the user $\frac{W}{2}$, with default $W = H = 32$ pixels. When cropping near image boundaries, the extracted patch is zero-padded to maintain consistent spatial dimensions. Temporal augmentation is optionally applied, which randomly removes frames from the start or end of the sequence while ensuring that a minimum number of frames before and after the event remain non-zero. This process simulates partial occlusions in the temporal profile of an event, which makes the neural network familiar with the lack of input frames (blank images or just no more images exist). Additional augmentation is performed by temporally shifting the center frame index f_c by small offsets (e.g., ±2 frames), re-extracting the sequence, and applying ±90° rotations to generate further training variations. All augmented sequences are concatenated along the batch dimension, such as:

$$X' = concat(X, X_{aug1}, X_{aug2}, \dots)$$

For random burst event classification, the final training set comprised 24,916 augmented samples generated from 7,931 original sequences extracted from 608 movies (**Table 3**). Depending on the encoder configuration, each sample had spatial dimensions of 32×32 and a temporal dimension of either 26 or 28 frames.

Source movies were recorded at 10 Hz, while movies recorded at 50 Hz were standardized to 10 Hz either by using ImageJ's "reduce" function or by applying temporal max pooling within IVEA with a downsampling factor of five. Both the recurrent and transformer-based networks were trained on an Nvidia RTX 3070 graphics card.

Table 3 Number of movies/events used to train the eViT and the LSTM models (adapted from (Chouaib et al., 2025)).

Neural network	Cell type	Indicator	Number of movies/events
eViT	CTL	pH sensitive (pHuji)	151/1156
	CTL	pH insensitive (tdTomato)	347/3806
	CTL or HEK cells	CD63-pHuji or SEP	47/309
	Chromaffin or INS cells	pH insensitive (mCherry)	50/2451
	Simulation movies	pH-sensitive (with cloud)	13/209
LSTM	DRG neuron	SypHy	39/11,300

2.23 Metrics calculation for neural network evaluation

Model evaluation was performed on entirely new datasets not used during training. Instead of deriving a validation subset from the training data, the evaluation used diverse datasets sourced from multiple laboratories and acquired with different microscopy platforms. These included recordings of lytic granule exocytosis in T cells, dense core granules in chromaffin cells and INS1 cells with both pH-sensitive and pH-insensitive markers, and dorsal root ganglion neurons expressing Synaptophysin-SEP, and dopaminergic neurons analyzed with infrared paint dopamine nanosensors ("AndromeDA").

Evaluation followed the default IVEA configuration with automated parameter estimation, although the software allows manual parameter adjustment. By default, events occurring within the first four frames of a recording were excluded from analysis to allow system calibration. As a note, users can optionally reduce sensitivity to 1 or lower to detect additional local maxima.

All evaluation outputs were reviewed by human experts. Detected events were classified as true positives if they represented correct detections of exocytosis, or false positives if incorrectly identified. Exocytosis events missed by IVEA but confirmed by human experts were counted as false negatives. Precision, recall, and the F1 score were calculated using the standard formulas:

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F1\ Score = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

IVEA performance testing was carried out on a range of hardware configurations, with the baseline system consisting of an Intel Core i3 processor and 32 GB of RAM without GPU acceleration.

2.24 Labeling a new data type

Accurate labeling of training data is a key prerequisite for developing reliable neural network models in IVEA. However, manual annotation in general-purpose tools such as ImageJ's *ROI Manager*, while flexible, can be time-consuming, error-prone, and inconsistent when applied to large datasets or by multiple annotators. To address these limitations, a dedicated *ROI Labeler* plugin (**Figure 19**) was developed and integrated into IVEA, designed to simplify and standardize the labeling process for exocytosis event detection.

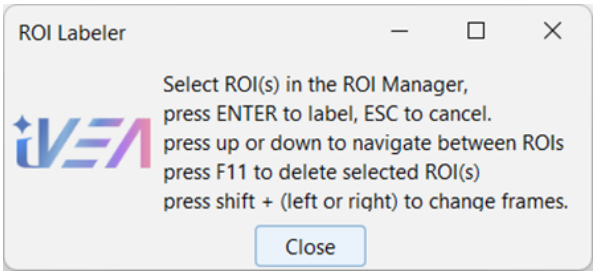


Figure 19. ROI Labeler ImageJ Plugin graphical interface.
The interface allows users to navigate through ROIs and video frames and assign labels efficiently using keyboard shortcuts.

This plugin minimizes repetitive actions, shortens annotation time, and maintains uniform label assignment across large-scale experiments. By enabling rapid navigation through ROIs and video frames via keyboard shortcuts and allowing single-keystroke category assignment, the ROI Labeler reduces user workload while maintaining high labeling throughput. This is particularly important when processing datasets containing thousands of candidate events, where small inefficiencies can accumulate into substantial delays.

Beyond speed, the ROI Labeler incorporates a predefined label mapping scheme that separates biologically relevant exocytosis events (positive classes) from imaging artifacts and noise (negative classes) (**Table 4**) (**Figure 20**). Positive classes are consistently mapped to positive integers, and negative classes to negative integers. This convention ensures clear semantic separation between event types during training and prevents class misinterpretation when datasets are merged or reused. This structured mapping allows labeled ROIs to be reliably exported to HDF5 format without additional manual verification, supporting a fully reproducible training pipeline.

The labeling categories remain configurable via IVEA's JSON configuration file, enabling researchers to adapt the labeling scheme to specific experimental conditions without modifying the underlying software. This flexibility enables the inclusion of new event categories, redefinition of classification criteria, and refinement of negative class distinctions, ensuring the annotation process remains aligned with evolving research objectives.

Table 4. Label reference table for ROI annotation and classification.
The table lists all predefined label categories, distinguishing positive exocytosis events from negative artifact/noise events to maintain consistency during manual and automated labeling.

Exocytosis (Positive)		Artifacts and Noise (Negative)	
Label	**Category Description**	**Label**	**Category Description**
0	Fusion with a cloud	-1	Random noise
1	Fusion without a cloud	-2	Granule intensity fluctuation
2	Latent granule fusion	-3	Moving granule
		-4	Random noise with intensity fluctuations

		-5	Intensity flickering and out-of-focus artifact
		-6	Intensity rise (granule docking)
		-7	Intensity fade (granule undocking)
		-8	Light artifacts (passing light, waves, etc.)

In practice, the ROI Labeler serves as a practical tool for efficiently preparing datasets used in model refinement or new training sessions within IVEA. It provides a simple and consistent workflow for assigning event categories, making it easier to integrate additional experimental data into existing models or create new labeled datasets.

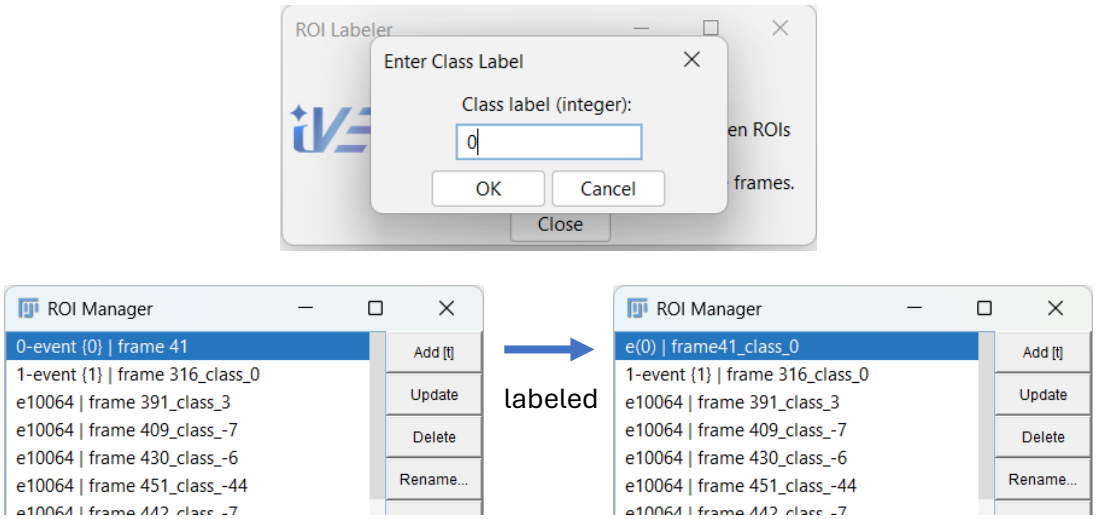


Figure 20. Labeling event in the ROI Manager using ROI Labeler plugin.
Example of assigning labels in a manner compatible with IVEA's automated dataset preparation pipeline.

2.25 Transfer learning and training on a new data type

The training process for new data types is initiated through the Python-based graphical interface provided by the "IVEA_main" script. This interface streamlines dataset creation, model training, and refinement procedures while maintaining a consistent and reproducible workflow. Labeled video files, annotated in ImageJ, are processed into structured training sets stored in HDF5 format for efficient loading during training. Newly labeled datasets can be merged with existing ones within IVEA to extend the available training data and maintain compatibility across experiments.

The interface allows for both the training of models from scratch and the refinement of pre-trained architectures such as GranuVision3 and NeuroVision1. Upon completion, trained networks are saved in Keras format together with a JSON configuration file containing all relevant parameters and architecture specifications. These models can then be re-imported into IVEA via the ImageJ plugin for direct deployment. If no user-defined network is selected, IVEA defaults to its bundled pre-trained

models. To ensure consistent project management, both the models and processed datasets are stored within predefined directories specified in the application settings, with all paths and configuration options modifiable through the associated JSON settings files.

Transfer learning allows a model trained on one dataset to be repurposed for a related task by leveraging the general feature representations learned during previous training (Raghu et al., 2019). In the present implementation, the focus was on reusing the feature extraction capabilities of high-performing networks while updating only their classification layers. The current version of IVEA (v2.3) includes five classification models: three for random burst events (*GranuVision3*, *GranuVision2*, *GranuLSTM*) and two for stationary burst events (*NeuroLSTM*, *NeuroVision1*) (**Table 5**). Models with "Vision" in their name employ a vision transformer backbone, while those ending in "LSTM" use recurrent architectures. Although all five are capable of activity recognition and classification, only *GranuVision3* and *NeuroVision1* are currently supported for refinement via transfer learning (**Figure 21**), having demonstrated superior baseline performance compared to the other models. Transformer-based networks generally outperform their LSTM counterparts in our current project, especially for recognizing random burst events. Additionally, our multivariate LSTM model tends to be more challenging to train due to the feature extraction complexity for normal users to adapt to.

Table 5. Available classification model within IVEA

Analysis Module	Neural network	Model name	Description	Status
	eViT	GranuVision3	With cloud, without cloud, and latent granule fusion	Default
Random	eViT	GranuVision2	With cloud and without cloud	Option 2
	LSTM	GranuLSTM	Cloud, without cloud, and latent granule fusion	Option 3
Stationary	LSTM	NeuroLSTM	Burst events and slow events	Default
	eViT	NeuroVision1	Burst events and slow events	Option 2

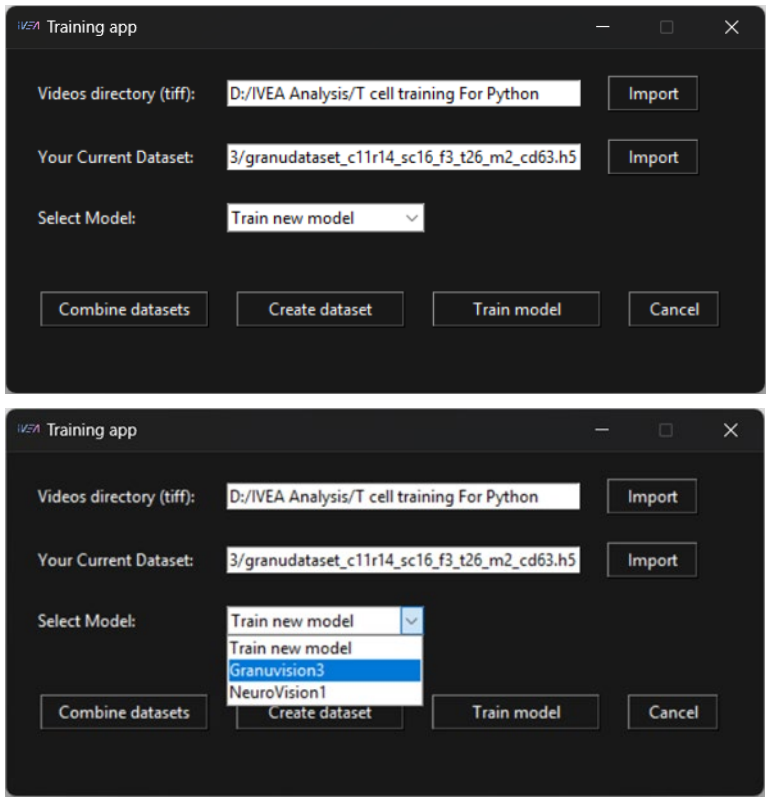


Figure 21. IVEA_main script GUI model selection.
The interface allows users to choose between available models, here showing GranuVision3 selected from the list.

For refinement, all layers of the base network, including the convolutional encoder, positional encoder, and transformer blocks, were kept frozen; this is known as fine-tuning or layer freezing (Tan et al., 2018). Only the final two Dense layers were retrained: an intermediate layer with 128 units and the final classification layer with 10 output units using the softmax activation function. This strategy retained the spatiotemporal feature representations already learned by the network, significantly reducing computational demands and enabling refinement to be carried out on systems without GPU acceleration. The decision to retrain only the final layers was based on the need to adapt to updated class distributions without disrupting the deeper learned features (Yosinski et al., 2014), which remain relevant across related tasks.

To prevent performance degradation in previously learned classes during refinement, a memory-buffer approach was adopted (Rolnick et al., 2018). Twenty representative samples per class were stored and selectively replaced when new labeled samples for the same class were introduced. This approach reduced catastrophic forgetting while preserving the model's ability to recognize all classes. New data were further augmented through rotation, scaling, and temporal shifting to increase variability and robustness to spatiotemporal transformations.

2.26 Google TensorFlow-Java implementation

The recurrent and transformer-based neural networks in IVEA were developed in Python (v3.8.10) using the TensorFlow framework (v2.9.1–v2.10.1) and the Keras API for model construction and training. After training, models were exported in Protocol Buffer format (.pb) for deployment. To enable direct integration of these models into the ImageJ Fiji environment, the TensorFlow Java library (v1.15.0; *artifactId*: *tensorflow*) and its native bindings (*artifactId*: *libtensorflow_jni*) were employed in combination with the DeepLearning4J framework (*artifactId*: *deeplearning4j-core*, v1.0.0-M1.1). This approach ensured compatibility with Fiji's Java-based architecture and avoided the need for external Python execution during inference, allowing models to run natively within Fiji plugins.

Although Java offers strong cross-platform compatibility and tight integration with ImageJ, deploying TensorFlow within this environment proved technically challenging. Compared with Python, Java's machine learning ecosystem remains relatively limited, with TensorFlow's Java bindings providing only partial functionality and suffering from outdated or incomplete documentation as of 2024. Furthermore, ensuring Fiji compatibility required packaging all dependencies into a single Java Archive (JAR) file, which imposed additional constraints on plugin architecture and distribution.

To execute TensorFlow models from within a .jar file in Fiji, it was necessary to manually embed specific library components from the Maven "*m2*" repository into the plugin JAR. The directories "*org/tensorflow, org/nd4j*", and "*tensorflow*" were placed at the root level of the JAR to ensure proper runtime resolution of dependencies during inference via TensorFlow's *serve* functionality.

In Eclipse, the functional plugin JAR was generated using the *Export → Runnable JAR* option from the Maven project, which allowed manual inclusion of the TensorFlow dependencies. This approach produced a compact and platform-specific JAR, with the Windows version reaching approximately 63 MB after integrating the required TensorFlow binaries.

Alternatively, for macOS and Linux environments, the entire project was built as a self-contained ("fat") JAR using a Maven clean build, which embeds all dependencies directly into the archive. While this method ensured full compatibility across systems, it increased the final file size to over 130 MB, substantially larger than a typical ImageJ plugin.

These additional steps were introduced to maintain portability and reproducibility across user systems without requiring complex local installations. To support adoption and facilitate reproducibility, a detailed technical guide with installation instructions, configuration steps, and example code has been made publicly available at https://github.com/AbedChouaib/IVEA. This guide is intended to assist researchers in integrating TensorFlow Java into Fiji for deep-

learning-based image analysis workflows, reducing the entry barrier for extending Fiji's capabilities with advanced neural network models.

2.27 Video simulation and noise control

A simulation environment was developed in MATLAB to evaluate the performance of the IVEA detection algorithms and neural network models under controlled conditions. This setup enabled the generation of synthetic video data simulating granule dynamics in near-ideal scenarios, both in the absence of noise and under varying levels of Poisson noise. The aim was to obtain deeper insights into model behavior and robustness in conditions that mimic total internal reflection fluorescence microscopy (TIRFM).

Simulated videos were constructed as 3D image stacks (x, y, t), each of size 200 × 200 pixels over 200 frames, representing granule activity across time. Granules were modeled as small spherical structures exhibiting random motion, fusion events, and Gaussian-distributed fluorescence intensity, consistent with TIRFM imaging. The spatial intensity profile of each granule was represented by a truncated 2D Gaussian function with a cutoff at 2σ:

$$g(x,y) = \mu \cdot e^{-\frac{\left((x-x_c)^2+(y-y_c)^2\right)}{2(\sigma)^2}}$$

where μ is the intensity of the granule, and $\sigma \in [1.1, 3]$, $\sigma \in \mathbb{R}$ is the standard deviation controlling the spatial spread of the distribution to mimic the radius of real vesicles.

Each simulation contained nine granules of varying sizes, with seven of them undergoing exocytosis. Granules were assigned randomized initial positions, intensity spreads, and peak intensities ranging from 50 to 100. Motion was introduced through random steps in both x and y directions, constrained within the image boundaries. At the fusion moment, granules became stationary and faded rapidly by subtracting a secondary Gaussian profile; negative values were clipped to zero. This behavior was encoded in the spatiotemporal volume to provide a realistic simulation for TIRFM-like dynamics.

This synthetic dataset not only enabled controlled benchmarking of detection algorithms but also supported the training of the eViT network under diverse imaging conditions, strengthening its ability to generalize across experimental noise environments.

The next step is to add the granule exocytosis simulation. At the time of the simulated granule fusion, another Gaussian model is added to represent exocytosis. The event simulation should correspond to a dynamic spread of high-intensity pixels similar to a cloud that expands and dissipates over time. For this case, a spatiotemporal Gaussian distribution model was used to control the temporal behavior of the exocytosis event (Chouaib et al., 2025), such as:

$$h(x,y,t) = \mu e^{\frac{-\Psi(x,y,t)}{(2\sigma_s)^2}} \cdot e^{\frac{-(T-t)^2}{(2\tau)^2}}$$

$$\Psi(x, y, t) = (\Delta x^2 + \Delta y^2) \cdot e^{\frac{(T-t)^2}{(2\tau)^2}} \quad \text{with } \sigma_S, \tau > 0$$

The function $\Psi(x, y, t)$ models how the granule's contents disperse radially over time. Here, t denotes the current frame, T marks the frame at which peak fluorescence is reached, τ is the time interval, i.e., number of frames, in which the fusion event occurs, σ_S represents the radial extent of the spread during fusion, and μ determines the overall fluorescence amplitude.

For model training and robustness testing, batches of eight videos were simulated (5 for testing, 3 for training), each subjected to nineteen different Poisson noise scaling factors. These noise variations were introduced using MATLAB's built-in functions. Initially, noise-free baseline videos were generated, after which artificial white noise was added using *"imnoise()"*. To simulate photon shot noise typical in microscopy, *"poissrnd()"* was applied, followed by Gaussian blurring to mimic the microscope's point spread function. The resulting noise was then combined with the baseline videos to construct a range of signal-to-noise ratios (SNRs) suitable for comprehensive noise control evaluation.

The Poisson noise reflects the photons striking the microscope's sensor cameras, resulting in random increases in the fluorescence signal and the generation of noise (Apergis et al., 2025). This type of noise is mathematically expressed as:

$$I_{noise}(x, y) \coloneqq Poisson(\lambda \cdot I(x, y))$$

where, λ serves as the Poisson noise scaling factor.

The noise scaling factor λ determining the level of noise applied to the image. The scaling factor directly influences the variance of the Poisson noise, as the noise is proportional to $\lambda \cdot I(x, y)$, thereby controlling the intensity of noise in the resulting image I_{noise}.

2.28 Hotspot area detection algorithm

Hotspot detection in IVEA is adapted from the Dopamine Analysis Recognition Tool (DART) framework (Elizarova et al., 2022), which employs k-means clustering to perform frame-wise intensity segmentation. While DART partitions each frame into multiple intensity-based layers, its original ratio-based foreground subtraction method is sensitive to spatially non-uniform fluorescence, often leading to inconsistent background removal.

To address this limitation, IVEA implements the Multi-Layer Intensity Clustering (MIC) algorithm, an enhanced approach that improves robustness against heterogeneous illumination. MIC applies k-means clustering to the first frame of the sequence, segmenting it into k layers, each grouping pixels of similar grayscale intensity (Pham et al., 2000):

$$I(x, y) \rightarrow I(x, y, k) \mid k = \mathbb{N}^K$$

where K represents the user-defined number of clusters, with a default value of 5. In most applications, the lowest-intensity cluster is assumed to represent the background. Unlike DART, which calculates inter-frame intensity differences within each cluster, MIC uses a ratio-based normalization:

$$I_i'(k,x,y) := \left(\left(\frac{\mu_{i-n}(k)}{\mu_i(k)} - 1\right) \cdot \theta + 1\right) \cdot I_i(k,x,y)$$

where i is the frame index, n is the temporal offset, k is the layer index, θ is a tunable scaling parameter (default $\theta = 1$), and $\mu_i(k)$ is the mean intensity of the cluster k at frame i. In the case of spatially uniform fluorescence, the segmentation reduces to a single layer ($k = 1$), and MIC becomes equivalent to the classic simple ratio operation.

Following intensity normalization, foreground detection is performed using a two-phase iterative thresholding procedure. In the first phase, two reference frames containing no events are selected to establish a baseline. Their pixel-wise difference ΔI is computed and converted to an 8-bit value to limit the search space to 255 threshold steps, optimizing computational efficiency. This difference image is processed iteratively through three sequential operations: First, thresholding, initialized at half the mean intensity of ΔI. The second, using morphological erosion with a structuring element K_e (default size 3×3), to eliminate isolated pixels:

$$\Delta I = \Delta I \ominus K_e$$

Third, median filtering, with a user-defined or default radius, to further suppress noise and preserve coherent structures.

After each iteration, the mean gray value of the processed image is computed. The threshold is incremented by one gray level per iteration until the mean reaches zero. This value is recorded as the first threshold decision v_1.

In the second phase, the threshold is refined for the remaining frames. A region of the segmented background is selected from each image (Fig. 8b) to compute a correction threshold v_2. The final global threshold is then calculated as:

$$v_i = v_2 \cdot \alpha$$

Where α is the sensitivity parameter. If $\alpha = 0$, the system takes two additional frames without events to estimate and adjust α automatically. This adaptive step compensates for differences between full-image thresholding and background-only thresholding.

Pixels exceeding the global threshold v_i are labeled as event candidates. Each contiguous foreground region is assigned a unique identifier, and its fluorescence intensity is subsequently tracked both

spatially and temporally. For each detected event, the mean intensity $\mu_e(t)$ is computed over a fixed spatial region at each time point. The mid-intensity is then defined as:

$$\mu_{mid} = \frac{1}{2}\left(\mu_e(t_{min}) - \mu_e(t_{max})\right)$$

Once the event's fluorescence drops below μ_{mid}, tracking is stopped because the event is considered to have vanished (Fig. 8c,d).

By combining MIC-based illumination correction with iterative, two-phase thresholding, the algorithm achieves high sensitivity in detecting transient fluorescence hotspots while maintaining robustness against heterogeneous background signals.

2.29 Biological datasets and live-cell imaging

The development of the IVEA framework and its Python extension, IVEA-Py, was based on a diverse collection of live-cell imaging datasets provided to me through multiple collaborations. These datasets, which originated from independently conducted biological studies, were essential for defining the requirements, architecture, and robustness of the analysis pipeline. Each dataset represents a distinct model of regulated secretion, including neurons, cytotoxic T lymphocytes, chromaffin cells, insulinoma cells, and dopaminergic neurons, recorded with different microscopy setups and fluorescent reporters. This diversity ensured that IVEA was developed and validated on heterogeneous experimental conditions, reflecting real-world imaging variability in vesicle dynamics and exocytosis.

The following subsections summarize the imaging configurations used for each biological system and indicate their origin and technical parameters.

2.29.1 Murine CD8$^+$ T cells

Two total internal reflection fluorescence (TIRF) microscopy configurations were used for imaging mouse cytotoxic T lymphocytes, depending on the fluorescent reporters expressed.

Setup 1, Synaptobrevin-mRFP / GzmB-mTFP / GzmB-tdTomato recordings. These experiments were carried out on an Olympus IX70 stand with a 100×/1.45 NA Plan Apochromat objective coupled to a TILL-TIRF condenser (TILL Photonics, Germany). Image sequences were acquired using either a QuantEM 512SC EMCCD or a Prime 95B sCMOS (Teledyne Photometrics, USA), yielding effective pixel sizes of 160 nm or 110 nm. Fluorophores were excited with a multi-line argon laser (488 nm) and a 561 nm laser (Melles Griot), and recordings were controlled using Visiview (v4.0.0.11). Movies were captured at 10 Hz..

Setup 2, pHuji-based reporters (Synaptobrevin-pHuji, GzmB-pHuji, CD63-pHuji). A separate configuration based on an Olympus IX83 microscope equipped with an autofocus module and a 100×/1.49 NA UAPON100XOTIRF objective was used for experiments involving pH-sensitive pHuji

constructs. Excitation was provided by 445, 488, and 561 nm lasers (100 mW each), and TIRF incidence was controlled with an iLAS2 illumination module (Roper Scientific). Recordings were acquired with the same camera models used in Setup A, resulting in 110–160 nm pixel sizes. Frame rates ranged from 5–10 Hz, with total acquisition durations of 10–15 minutes per cell.

2.29.2 Murine dorsal root ganglion neurons

DRG neuron datasets were collected using the same TIRF configuration described in Setup A above. These recordings served to evaluate the algorithm's performance on neuronal vesicle fusion events, which typically occur in smaller synaptic boutons and exhibit more heterogeneous background intensities.

2.29.3 Chromaffin cells

Exocytosis in chromaffin cells was recorded at 10 Hz with 100 ms exposure time. Imaging was performed using either a Micromax 512BFT (Princeton Instruments) or a QuantEM 512SC (Photometrics) camera paired with 100×/1.45 NA objectives (Olympus Plan Apochromat or Zeiss Fluar). Resulting pixel sizes were 130 nm or 160 nm. These datasets were used to demonstrate IVEA's applicability to large dense-core vesicle fusion.

2.29.4 INS-1 (Insulinoma-1) cells

INS-1 cells (clone 832/13) transiently expressing NPY-tagged fluorophores (NPY-mGFP, NPY-mNeonGreen, NPY-mCherry, or NPY-eGFP) were imaged using custom-built TIRF microscopes based on Zeiss AxioObserver systems with 100×/1.46 NA objectives. Excitation was provided by 473 nm and 561 nm diode lasers controlled through an AOTF. Emission channels were separated using a Dual-View splitter, allowing simultaneous dual-color imaging on a Prime 95B sCMOS camera, producing ~110 nm pixel resolution. Frame rates were 50 Hz for NPY-mNeonGreen and 10 Hz for NPY-mCherry. Additional eGFP recordings were acquired on an AxioObserver Z1 with a QuantEM 512SC EMCCD at 160 nm per pixel.

2.29.5 Human $CD8^+$ T lymphocytes

Human CTLs were imaged on a Nikon Eclipse Ti2-E inverted TIRF microscope equipped with a 100×/1.45 NA Plan Apochromat LBDA objective and an iLAS2 illumination module. Fluorescence excitation used a 561 nm, 150 mW diode laser (Coherent) routed through a ZET405/488/561/647 optical path (Chroma). Images were recorded with a Prime 95B sCMOS camera at 110 nm per pixel. Single recordings typically lasted 20–30 minutes at a frame rate of 9 Hz. Data acquisition was managed using MetaMorph (v7.10.5.476) and Modular v2.0 (GATACA Systems).

2.29.6 Dopaminergic neurons

Recordings of dopaminergic neuron activity were obtained with a 100× oil-immersion UPLSAPO100XS objective (Olympus) in combination with a Cheetah-640-TE1 InGaAs camera

(Xenics). The final pixel size was 150 nm, and recordings were acquired at 15 Hz. These datasets were used for nanosensor-based analysis of dopamine release using AndromeDA.

2.30 Writing support and language refinement

For clarity and consistency in English writing, Grammarly (v1.2.212.1789) and ChatGPT-5 were used to improve grammar, spelling, and phrasing. They were applied only for language checks. They did not create scientific content or affect the study design, data, or conclusions.

2.31 Statistical analysis and software environment

SigmaPlot (Version 14.5.0.101) was used for all statistical tests. Reported p-values came from two-tailed tests with 95% confidence intervals. Student's t-tests, one-way ANOVA, or ANOVA on ranks were used based on the data. MATLAB (Release 2024b) and Excel 2021 handled data processing and calculations. A trained expert reviewed and annotated imaging datasets in Fiji (ImageJ 1.54p). Results were compared with other tools, including ExoJ 1.09, pHusion, and SynActJ 0.3.

2.32 IVEA software development in java

The Java implementation of IVEA was developed using Java SE 1.8.0_322 within the Eclipse IDE. The software incorporates several libraries, including ImageJ (ij Version 1.54c), Bio-Formats and LOCI plugins, OpenCSV, DeepLearning4J Core v1.0.0-M1.1, Google TensorFlow v1.15.0, and libtensorflow_jni v1.15.0. The Java version of IVEA includes modules for automated region selection, preprocessing through intensity-based filtering, iterative thresholding, Difference-of-Gaussian operations, Gaussian non-maximum suppression, event tracking, event classification, multi-layer brightness correction, and mono-exponential curve fitting.

2.33 IVEA-Py software development in python

Model development and training were conducted in Python Version 3.8.15 using Visual Studio Code Version 1.100.2. Neural network models were implemented with TensorFlow (versions 2.9.1 and 2.10) and Keras, along with other coding libraries such as NumPy, scikit-image, scikit-learn, tifffile, OpenCV, pandas, h5py, read_roi, shutil, and Tkinter. IVEA-Py framework uses several algorithms, techniques and machine learning models, including long short-term memory networks, Vision Transformer architectures, convolutional neural networks, k-means clustering, Gaussian non-maximum suppression, and multi-layer temporal intensity correction, gradient vector field with Euler integration, Euclidean distance transform, gradient vector elastic diffusion, Kalman filter, Laplace of Gaussian filter, difference of Gaussian filter, MAD thresholding. These components form the core of IVEA's detection, classification, and tracking workflow.

2.34 Figure preparation

Figures and schematics used throughout the thesis were generated and edited primarily using Adobe Illustrator Version 29.5.1 and, to a lesser extent, CorelDRAW Version 23.5.0.506. Additional

adjustments to image layout, annotations, and basic contrast correction were performed using Fiji. Graphs and plots with a fancy style were created using Python code, while bar graphs were created using MATLAB.

3 Results

Developing a unified and fully automated framework for exocytosis analysis is inherently challenging due to the broad variability in event signatures. These variations arise from differences in cargo release dynamics, cell morphology, fluorophore properties, imaging conditions, and the SNR. Consequently, reliable analysis requires methods capable of distinguishing between biologically distinct event types and adapting computational strategies to each with high accuracy. Within the context of vesicle fusion, exocytic events can be broadly categorized into three groups (**Figure 3. Figure 4. Figure** 5), based on their spatial and temporal characteristics, their underlying biological mechanisms, and the imaging signatures they generate. Two of these exocytosis categories correspond to localized, transient fluorescence increases associated with individual vesicle fusion events commonly referred to as fluorescent burst events. Despite sharing a common fluorescence signature, these two subcategories differ in the spatial features and stability of their occurrence.

The first subcategory, termed random burst events, is characterized by individual vesicle or granule fusion events that often exhibit visible pre-fusion mobility within the evanescent field (**Figure 3b, c**). Such events can occur at virtually any location along the plasma membrane and exhibit highly variable spatial and temporal dynamics. Following fusion, the vesicle typically collapses into the membrane, and its signal disappears, as the entire cargo is released. However, alternative behaviors are also observed: some vesicles undergo incomplete or transient fusion and can fuse again at the same site; in other cases, sequential fusion events arise when a second vesicle rapidly follows and fuses at or near the same location (Ge et al., 2022; Tsuboi & Rutter, 2003; Yuan et al., 2015). Vesicles may also cluster in close proximity, giving rise to neighboring or simultaneous fusion events. When vesicles are fluorescently labeled and thus trackable prior to release, fusion can often be anticipated by observing their approach and arrest at the membrane before exocytosis. In the absence of visible vesicle trajectories, exocytosis manifests as stochastic, transient bursts of fluorescence that emerge and fade across the membrane. Such bursts resemble scattered flashes distributed irregularly in both space and time. The resulting variability motivated the designation of random burst events, highlighting their apparent unpredictability and dependence on vesicle mobility and availability. Therefore, detecting them requires a pipeline capable of capturing short-lived, localized fluctuations in fluorescence intensity regardless of their position within the field of view.

The second subcategory, termed stationary burst events, occurs repeatedly at fixed locations, such as presynaptic active zones in neurons (**Figure 4a**). These sites serve as hotspots for neurotransmitter release, where multiple vesicles fuse in rapid succession over short time intervals (Jahn & Sudhof, 1994). The fluorescence profile of such events is characterized by one or more bursts at the same coordinates without any substantial object displacement between bursts. The detection strategy for such

events must prioritize temporal recurrence at a single location over spatial features, in order to distinguish true repeated fusions from unrelated transient signals.

The third category, hotspot area events, differs fundamentally from the discrete-vesicle burst classes. These events are commonly observed in experiments using specialized neurotransmitter-sensitive probes such as near-infrared dopamine nanosensors (e.g., AndromeDA) or nanofilm-based detectors. When a neurotransmitter binds these sensors, they produce a spatially spreading fluorescence signal whose intensity scales with the local ligand concentration (Elizarova et al., 2022) (**Figure** 5). Unlike discrete vesicle fusions, hotspot events are diffuse, often covering larger contiguous regions, and may persist for extended durations depending on neurotransmitter clearance kinetics.

To accommodate these distinct signal profiles, IVEA integrates three specialized detection modules (Modules 1–3), each incorporating machine learning architectures optimized for its target event type (**Table 5**). This modular approach enables each detection pipeline to operate with the optimal trade-off between sensitivity, specificity, and computational efficiency. IVEA is distributed as a Fiji plugin with a user-friendly graphical interface (**appx. Figure 42**), allowing researchers to either run the detection fully automatically or adjust analysis parameters, retrain models, and fine-tune outputs according to their experimental needs.

3.1 Event detection workflow for modules 1 & 2

The IVEA workflow is divided into two main stages. In stage 1, the software scans the image stack for local fluorescence maxima, identifying potential event centers and recording their spatiotemporal coordinates (x, y, t). For each candidate, a square region of interest, typically 32 × 32 pixels, is extracted, centered on the detected coordinates. A temporal sequence of frames is also selected, encompassing both pre- and post-peak activity (e.g., 10 frames before and 10 frames after the detected maximum). This ensures that the model is provided with the full temporal context of the event, including baseline activity, the rise to peak fluorescence, and the decay phase.

In stage 2, these extracted spatiotemporal ROIs are classified using the neural network model appropriate for the event type. For random burst events (Module 1), an encoder-based Vision Transformer (eViT) processes the sequence (**Figure** 6 **a, b**), capturing spatial patterns in individual frames and their temporal evolution. For stationary burst events (Module 2), a Long Short-Term Memory (LSTM) architecture is employed (**Figure** 6**c**), providing a lighter yet effective framework for recognizing temporally recurring signals at fixed spatial locations. Prior to LSTM input, spatial frames are flattened to reduce memory usage and computational overhead, preserving only the features necessary for temporal pattern recognition (**Figure** 6 **a, d**). This preprocessing step is particularly important for large-scale recordings, where memory efficiency can be a bottleneck.

3.2 Validation using simulated datasets (module 1)

Before applying IVEA to experimental recordings, the detection accuracy was validated using simulated datasets with precisely defined ground truth (**Figure 22a**). The simulations represented random burst events with vesicle radii of 2.7 ± 0.36 pixels, matching the optical profile of lytic granules observed in cytotoxic T lymphocytes under TIRF microscopy. These synthetic videos were processed using the random burst detection module on standard CPU hardware without GPU acceleration, demonstrating that IVEA is computationally efficient even without specialized hardware. All parameters were kept at default values except for the neural network radius, which was set to 16 pixels to match the simulated vesicle size.

The initial evaluation showed that all simulated events were correctly detected, with zero false positives. To test noise robustness, the videos were incrementally distorted with additive white Gaussian noise and Poisson noise. The Poisson noise scaling factor λ was varied from 0.1 to 10 times the signal amplitude, corresponding to SNR ranging from high-quality experimental conditions to severely degraded cases (see section 2.27 video simulation and noise control). For low noise levels (λ = 0.1–1), IVEA achieved a recall of 99.71 % ± 0.29 %, precision of 94.49 % ± 3.23 %, and F1-score of 96.71 % ± 1.91 % (**Figure 22b**). At higher noise intensities (λ = 1–10), performance decreased due to small vesicles becoming indistinguishable from the noise base, with recall dropping to 96.86 % ± 2.55 %, precision to 79.22 % ± 4.68 %, and F1-score to 86.51 % ± 3.27 %.

These tests confirm that IVEA maintains high detection performance even under noise conditions significantly worse than those encountered in typical TIRFM experiments. The combination of robust preprocessing, tailored neural network architectures, and specialized modules enables the platform to generalize well to diverse acquisition conditions, making it directly applicable to experimental biological data without the need for extensive parameter re-optimization.

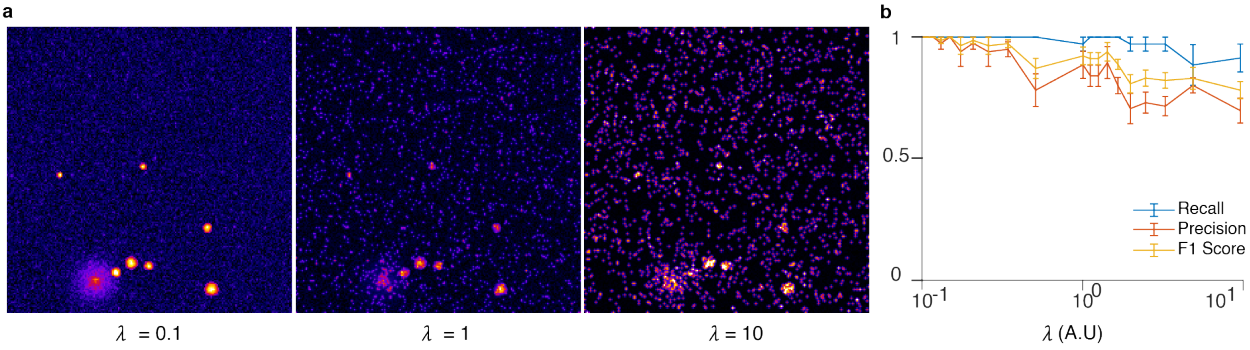


Figure 22. IVEA eViT model simulation analysis (adapted from (Chouaib et al., 2025)).
a. Influence of Poisson noise on simulated exocytosis recordings. The left panel shows a synthetic video sequence containing an idealized fusion event with low noise (λ = 0.1). The middle panel illustrates the same sequence with moderate noise (λ = 1). The right panel demonstrates a high-noise condition in which Poisson noise is set to ten times the signal amplitude. **b.** Quantitative assessment of eViT performance across noise conditions. Simulated datasets with noise levels ranging from λ = 0.1 to λ = 10 were analyzed, and the resulting recall, precision, and F1-score values are plotted as mean ± SEM (n = 5).

3.3 Ablation study (module 1)

An ablation study was conducted on the eViT architecture to systematically evaluate the contribution of individual network components to the overall classification performance. This involved selectively disabling or removing specific layers and examining the resulting effects on the model's ability to detect exocytosis events. The evaluation dataset was divided into two broad categories: positive labels corresponding to exocytosis events and negative labels corresponding to non-exocytosis. This binary grouping was chosen to simplify the evaluation of classification performance. Conducting ablation across all ten original classes would have required extensive examination of each noise or artifact category individually, which is both time-consuming and not central to the primary research target. Since the main objective was to distinguish genuine exocytosis events from non-events, all noise and artifact categories were merged into a single negative class. I made this choice for the ablation study to enable a clearer interpretation of the model's performance in identifying true exocytosis activity.

Two complementary approaches were employed. In the first, components were disabled only during inference without retraining, thereby isolating their role in the forward pass and assessing their direct contribution to the prediction probability (Desai & Ramaswamy, 2020). In the second step, selected layers were physically removed from the architecture, and the model was fully retrained from scratch, allowing an evaluation of how their absence affects representational learning and feature extraction capacity (Meyes et al., 2019).

In the inference-only experiments, the focus was on the positional encoder, the two transformer layers, and the final Dense layer. Predictions were consistently evaluated at a decision threshold of 0.5 to ensure comparability. The results revealed that Transformer layer 1 was the single most critical component for maintaining high recall (**Figure 23a**, 2nd col.). Disabling it reduced recall from 94% to 67%, demonstrating its central role in capturing temporal dependencies necessary for accurate exocytosis detection (**Figure 23a**, 3rd col.). Disabling Transformer layer 2 produced a milder reduction in recall (to 91%), indicating that while it contributed meaningfully, its absence was less detrimental than the removal of Transformer layer 1.

The positional encoder had a nuanced effect (**Figure 23a**, 4th col.). The dataset presented to the model contained events centered at the middle frame, with additional preceding and following frames included to account for slight temporal misalignments in acquisition. When the positional encoder was enabled, the model learned to map specific temporal positions explicitly, resulting in more precise and temporally strict detections (**Figure 23a** bottom). However, disabling the positional encoder increased recall by relaxing these temporal constraints, enabling detection of more true positives even when events occurred slightly off-center in time. This improvement in sensitivity came at the cost of increased false positives (**Figure 23a** top), highlighting a trade-off between recall and precision driven by positional encoding.

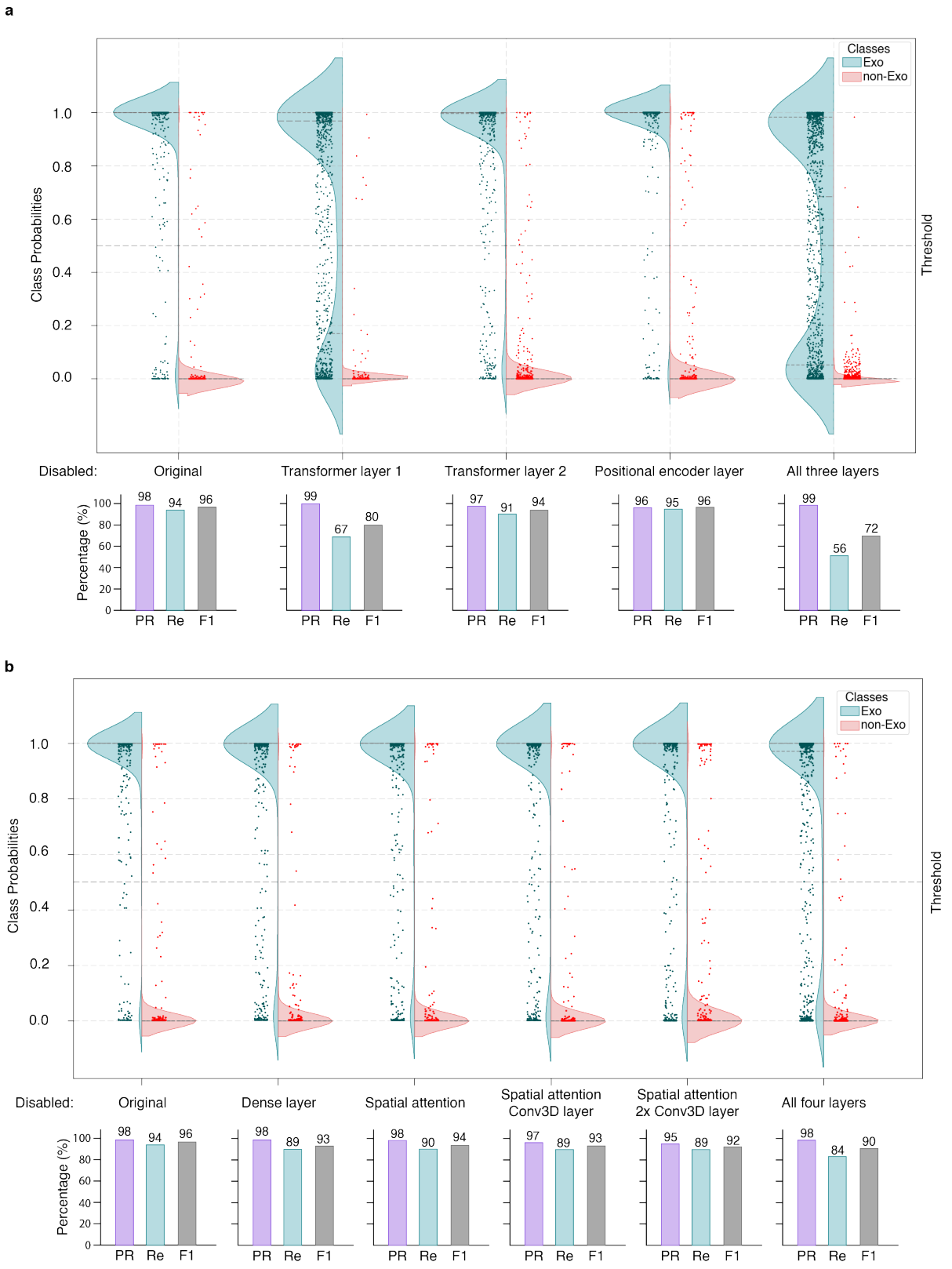


Figure 23. Ablation study for the eViT model (panel b is adapted from (Chouaib et al., 2025)).
a. Inference-only ablation results showing the effects of disabling individual layers without retraining. Violin plots display the distribution of predicted class probabilities for 2558 test samples (1467 non-exocytosis, red; 1091 exocytosis, turquoise). In the bar graphs below, each plot summarizes mean precision (purple), recall (blue), and F1-score (grey) values at a 0.5 decision threshold. **b.** Retraining-based ablation results, obtained by successively removing specific layers and retraining the model in six independent runs (five retrained models plus the baseline). Violin plots show prediction probability distributions, while the corresponding bar graphs summarize

performance metrics. The experiments reveal that transformer layers, particularly Transformer layer 1, and the convolutional backbone, are critical for maintaining both spatial and temporal detection capabilities. The positional encoder enforces temporal strictness, increasing precision but reducing recall when active, whereas disabling it increases sensitivity at the expense of specificity.

For the retraining experiments, the layers examined included the shared convolutional backbone layers and the penultimate Dense layer. Progressive removal of convolutional layers led to a stepwise decline in performance, with the most severe deterioration observed when only a single convolutional layer remained (Simonyan & Zisserman, 2014) (**Figure 23b**). This highlights the cumulative importance of multiple convolutional stages for extracting both spatial and temporal features prior to the eViT (Dosovitskiy et al., 2021; Vaswani et al., 2017).

Overall, this ablation analysis (**Figure 23**) demonstrates that the eViT's performance depends on a balanced interplay between convolutional feature extraction, temporal encoding, and positional awareness. The results show that each architectural component, from early convolutional processing to self-attention-based sequence modeling, contributes collectively to achieving robust exocytosis classification performance.

3.4 Random burst events analysis (module 1)

We analyzed multiple live-cell TIRF microscopy videos. We started the analysis of videos in which the fluorescent signal of granule exocytosis was feature-rich and moved step by step to videos with reduced amounts of features. To do so, we first analyzed CTLs undergoing secretion of relatively large lytic granules (granule diameter was 5–8 pixels **Figure 24a–c**). Lytic granules were fluorescently labeled via granzyme B fusion proteins carrying different fluorescent reporters. Depending on the experiment, the reporter was either pH-sensitive pHuji (**Figure 3c** high level of features), weakly pH-sensitive eGFP, or pH-insensitive tdTomato (**Figure 3b** low level of features). These labels produced distinct event intensity profiles, as illustrated in (**Figure 3b–c**). Each dataset consisted of recordings acquired at 10 Hz for a duration of around 8-15 minutes, containing between 1 and 33 individual CTLs per video. We further evaluated the software on CTL datasets acquired with alternative granule markers, such as LysoTracker Red (**Figure 26a, c**). These datasets were obtained in a different laboratory using a separate TIRF microscope, ensuring diversity in imaging conditions. To examine IVEA's capacity to generalize across distinct cellular systems, additional training and analyses were conducted on recordings with lower discernible features, including chromaffin cells (**Figure 24d**) and INS-1 β-cells (**Figure 24e**). Both cell types release smaller and more densely packed granules than cytotoxic cells, resulting in low-feature images that are more difficult to detect and classify. In these datasets, chromaffin cell vesicles were labeled with Neuropeptide Y (NPY)–mCherry fusion constructs, while INS-1 β-cell vesicles were tagged with NPY constructs carrying either weakly pH-sensitive fluorophores (mGFP, mNeonGreen) or the pH-insensitive probe mCherry.

The first neural network implemented in IVEA was the LSTM model, which was applied to both the random and stationary burst modules. Its main advantage was low computational demand during training and inference, allowing rapid prototyping across datasets. However, extensive testing on random burst events revealed consistent limitations. The LSTM struggled to detect small or clustered granules, particularly those undergoing abrupt disappearance (**appx**. **Figure 41a–b**). Analysis of the extracted features confirmed that such events exhibited weak and heterogeneous temporal signals, which the LSTM often failed to generalize effectively (**appx**. **Figure 41c**).

To address these shortcomings, a transformer-based architecture was introduced. Transformers usually offer greater representational power but require higher computational resources and a different training strategy. Rather than relying solely on temporal features (**Figure** 6**c, d**), a learnable feature extraction stage was designed through integrating a shared CNN encoder with the transformer. This enables the model to capture both spatial and temporal characteristics. The new encoder-Vision Transformer (eViT) model was trained directly on spatiotemporal image patches, eliminating the need for extensive post-processing.

Since IVEA already had a functioning LSTM model, it was initially used to generate ROI candidates and provide these automatically labeled events for training the eViT. For cases where the LSTM failed to detect events, candidate ROIs were exported and manually picked to produce reliable training sets. Through this iterative workflow, the eViT was established as the primary model within IVEA. Subsequent benchmarking focused on comparing the eViT's performance to the LSTM to guide model selection and validation. The eViT model used for random burst analysis was trained to classify events into 10 categories: Three distinct exocytosis types are fusion accompanied by a spatially spreading fluorophore cloud, fusion without spreading (sudden disappearance), and latent granule fusion (abrupt onset); Seven non-fusion categories are fast drift or focus change, granule movement, random noise, noise with intensity fluctuation, granules plus noise, and granule docking/undocking.

In this evaluation, five distinct datasets were compiled, each representing a different labeling strategy and cell type. The datasets were organized as follows: Set 1: CTL-GzmB–pHuji (**Figure 24a; Figure 25a**); Set 2: CTL-GzmB-tdTomato(**Figure 24b; Figure 25b**); Set 3: CTL-CD63–pHuji and CTL-CD9–SEP (**Figure 24c; Figure 25c**), representing exosome-associated exocytosis; Set 4: MCC or INS1-mCherry (**Figure 24d; Figure 25d**); and Set 5: INS1-mNeonGreen and eGFP (**Figure 24e; Figure 25e**). In addition, a small testing sample of LysoTracker Red–labeled CTLs was analyzed to evaluate cross-probe generalization.

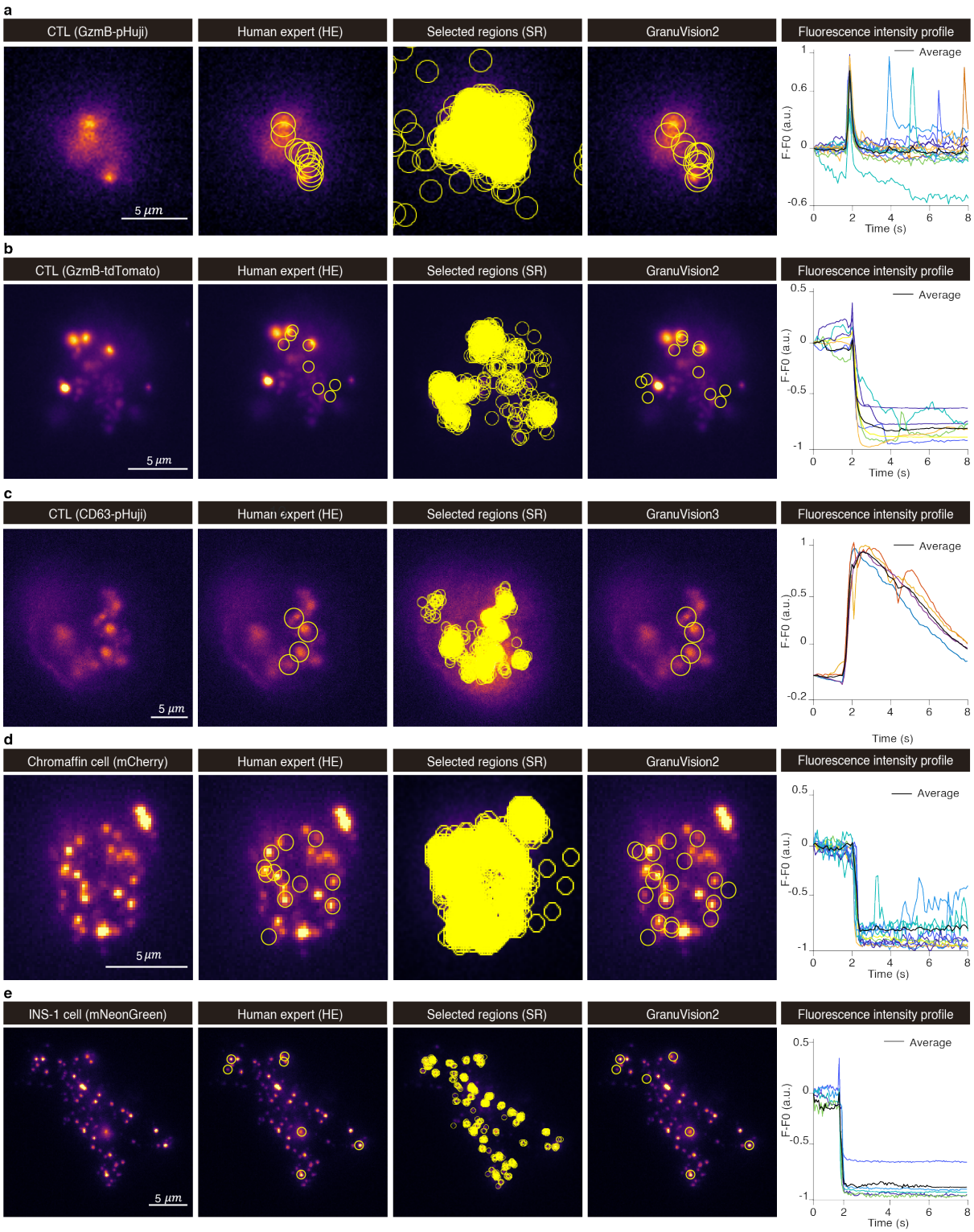


Figure 24. Result display for all video sets using the random burst event analysis (adapted from (Chouaib et al., 2025)).

Panels (a–e) illustrate the detection process compared with annotations from the human expert (HE). For each dataset, the raw TIRF image is shown alongside the expert's annotated exocytic events, followed by the set of all candidate regions selected before classification, and the events retained as positive by the eViT classification model (model name used indicated above each panel). The last column displays representative fluorescence traces of true-positive events detected by IVEA, aligned to their respective detection frames. The prominent fluorescence peaks appearing around 2 seconds correspond to the vesicle fusion. **a. Set 1**: This group includes recordings of cytotoxic T lymphocytes labeled with the pH-responsive reporter granzyme B-pHuji. A total of thirteen single-cell movies were evaluated (n_{cell} = 13). **b. Set 2**. The second dataset contains CTLs expressing

granzyme B-tdTomato, a pH-insensitive fluorescent probe. Seven recordings were analyzed, each comprising between one and eleven cells, resulting in 33 cells in total (n_{cell} = 33). **c. Set 3**. This dataset consists of CTLs expressing CD63-pHuji. A total of five recordings were analyzed, supplemented by a single HeLa cell dataset labeled with CD9-SEP that was retrieved from Zenodo (Liu et al., 2024). **d. Set 4**. For chromaffin cell analysis, NPY-mCherry was used as a pH-insensitive reporter. Five recordings from chromaffin cells were evaluated and combined with five additional movies of INS-1 cells expressing the same probe, yielding a total of ten cells (n_{cell} = 10). **e. Set 5**. The final dataset comprises INS-1 cells expressing either NPY-mGFP or NPY-mNeonGreen. Nine individual recordings were included, representing two weakly pH-sensitive reporters that differ in their release-cloud characteristics (n_{cell} = 9). These datasets were acquired at the Medical Cell Biology Department, Uppsala University, Sweden. The dataset is available on Zenodo (Dataset, 2025; Zenodo, 2025).

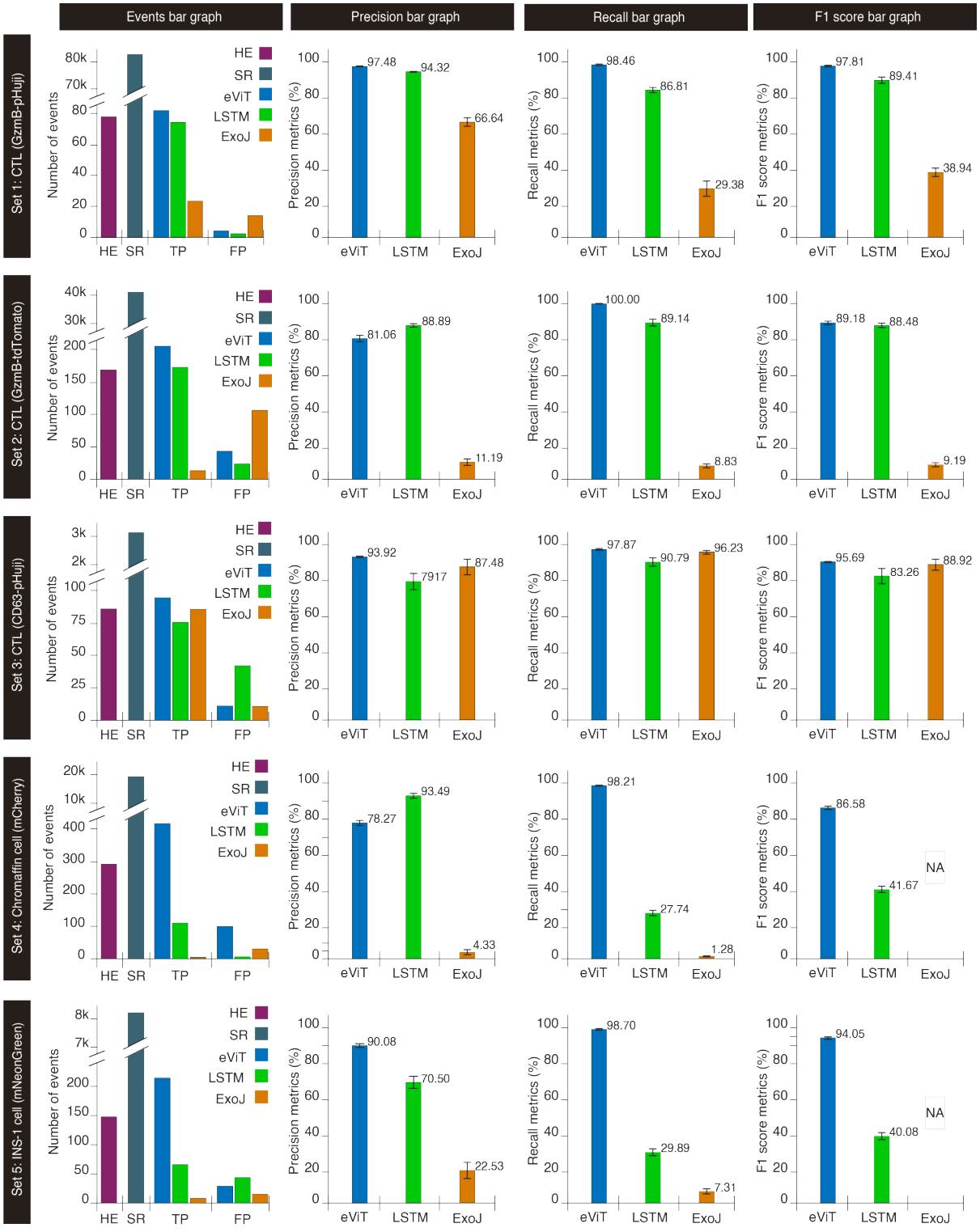


Figure 25 Bar graphs performance comparison of IVEA (eViT and LSTM) with ExoJ across five datasets (related to Figure 24).

Bar graphs summarize detection outcomes and classification metrics for random burst exocytosis events across five datasets. From left to right, the first panel shows the number of detected events, with counts separated into human expert (HE) annotations, selected regions (SR), true positives (TP), and false positives (FP). The next three panels display the precision, the recall, and the F1 score for eViT, LSTM, and ExoJ models. Datasets included CTLs expressing granzyme B-pHuji (**Set 1**), CTLs expressing granzyme B-tdTomato (**Set 2**), CTLs transfected with CD63-pHuji together with a HeLa dataset expressing CD9-SEP (**Set 3**), chromaffin cells expressing NPY-mCherry pooled

with INS-1 cells expressing NPY-mCherry (Set 4), and INS-1 cells expressing NPY-mGFP or NPY-mNeonGreen (Set 5). Across these datasets, eViT consistently achieved the highest precision, recall, and F1 scores, LSTM showed reduced recall, and ExoJ performed adequately only on CD63-pHuji but failed on pH-insensitive or weakly pH-sensitive reporters.

Using these five main datasets, manual analysis by a human expert (HE) identified 770 fusion events. When analyzed using IVEA with its default eViT-based classification module, the software initially registered ~156,000 ROI candidates, of which 2,418 were classified as true events by the network. In cases where exocytosis was accompanied by a signal spreading over a wide area (**Figure 3b–c**), a single biological event could be detected multiple times due to spatial overlap. Similarly, events that persisted over several frames produced additional duplicates arising from temporal redundancy. To address this, spatiotemporal non-maximum suppression is implemented utilizing a 3D Gaussian spread model (**Figure 13**). By applying this method across all 2,418 initial detections, IVEA removed 1,393 duplicates, leaving 1,025 unique true-positive events. This approach ensured that closely spaced detections in time and space were merged appropriately, while genuine independent events were preserved (**Figure 13**). Consequently, IVEA facilitated the identification of 255 additional TP events that were not initially detected by the HE. All additional detections were re-evaluated by the HE, who confirmed that they were either weak/small events difficult to detect by eye, events overlooked due to attentional fatigue, or events with atypical cloud morphology.

To assess model performance across cell types and labeling strategies, both architectures, the eViT and the LSTM, were systematically compared for the random burst detection module (**Figure 24; Figure 25**), evaluating them per cell type and fluorescent reporter. The evaluation was performed to determine the relative strength of each architecture under varying experimental conditions. The eViT GranuVision2 model was used for all datasets, except for the CTL-CD63–pHuji series (**Figure 24c**), where the eViT GranuVision3 variant was employed due to its additional training on latent fusion events.

The eViT model consistently outperformed all alternative tested methods, showing a clear and statistically significant advantage in classification performance. Based on these results, eViT was established as the primary model for the random burst event analysis module. As shown in **Figure 25**, the average performance across all datasets, eViT delivered a recall of 98.95 ± 0.40%, precision of 88.94 ± 3.64%, and F1 score of 93.13 ± 2.05%, compared to the LSTM's recall of 64.87 ± 13.18%, precision of 86.87 ± 3.05%, and F1 score of 68.58 ± 10.16% (Chouaib et al., 2025). To formally assess the significance of these differences, statistical comparisons were performed between the eViT, LSTM, and ExoJ ImageJ plugin (Liu et al., 2024) across all datasets (**Table 7**). The eViT compared to the LSTM often yielded non-significant differences in precision but revealed significant advantages for eViT in recall and overall F1 score, particularly in chromaffin and INS-1 datasets (**Table 7**). The eViT was also tested on new dataset type consisting of LysoTracker Red-labeled CTLs. Exocytosis events

labeled with LysoTracker exhibit fluorescence intensity profiles comparable to those of the pH-sensitive probe pHuji, but with longer decay phase, suggesting that the neural network should ideally be refined on this probe type (**Figure 26**c). Using the GranuVision2 model, the eViT identified 26 true-positives and 12 false-positives, whereas the HE detected 25 events. Hence, IVEA was able to detect all events observed by the HE with an additional event not detected manually (n_{cell}=4, **Figure 26b**). These results indicate that the eViT can effectively generalize to new datasets exhibiting probe-dependent differences in fluorescence kinetics. By relying on learned visual features rather than predefined intensity profiles, the model maintains robust performance across heterogeneous experimental conditions. Together with IVEA's automation and batch-analysis capabilities (**Table 7**), this establishes eViT as a reliable and scalable classifier for random-burst detection within the IVEA framework.

Table 6 (related to Figure 24): Performance summary for all five video sets for the eViT, LSM and ExoJ software (adapted from (Chouaib et al., 2025)).

In the "Quality" column, a greater number of asterisks indicates higher performance, reflected by improved recall, precision, and F1 score, as well as enhanced functionality such as batch analysis, automated parameter handling, and ROI data export. Conversely, the absence of asterisks denotes poor performance, indicating that the software is not suitable for the intended task.

	# of TP events	Recall (%)	Precision (%)	F1 score (%)	N_{cells}	Quality
CTL expressing pH-sensitive probe (Fig. 3a)						
HE	77					
eViT	85	98.46 ± 1.48	97.48 ± 1.29	97.81 ± 0.98	13	*****
LSTM	70	86.81 ± 4.42	94.32 ± 3.78	89.41 ± 3.79	13	****
ExoJ	29	26.85 ± 5.73	66.64 ± 9.68	38.94 ± 6.40	13	*
CTL expressing pH-insensitive probe (Fig. 3b)						
HE	168					
eViT	219	99.24 ± 0.34	81.71 ± 3.65	89.31± 2.12	33	*****
LSTM	172	89.14 ± 4.78	88.89 ± 2.32	88.48 ± 2.79	33	***
ExoJ	13	8.86 ± 2.67	11.14 ± 4.19	9.29 ± 2.76	33	
CTL expressing CD63-SEP for long-lasting events (Fig. 3c)						
HE	86					
eViT	96	97.84 ± 1.39	98.58 ± 1.30	98.16 ± 1.07	6	*****
LSTM	78	90.79 ± 5.70	79.17 ± 10.90	83.26 ± 9.30	6	***
ExoJ	82	96.23 ± 2.71	87.48 ± 9.97	88.92 ± 7.28	6	***
Chromaffin or INS1 cells expressing pH-insensitive probe (Fig. 3d)						
HE	292					
eViT	412	98.21 ± 6.64	78.27 ± 3.48	86.58 ± 9.81	10	****
LSTM	110	27.74 ± 6.64	93.49 ± 3.48	41.67 ± 2.79	10	**
ExoJ	4	1.2 ± 0.01	4.30 ± 3.34	10.00 ± 5.00	10	
INS1 cells expressing probes with little pH-sensitivity (Fig. 3e)						
HE	147					
eViT	214	98.70 ± 0.93	90.08 ± 2.54	94.05 ± 1.53	8	*****
LSTM	66	29.89 ± 4.35	78.50 ± 8.67	40.08 ± 4.63	8	*
ExoJ	8	8.38 ± 3.88	25.25 ± 12.36	23.75 ± 7.00	8	

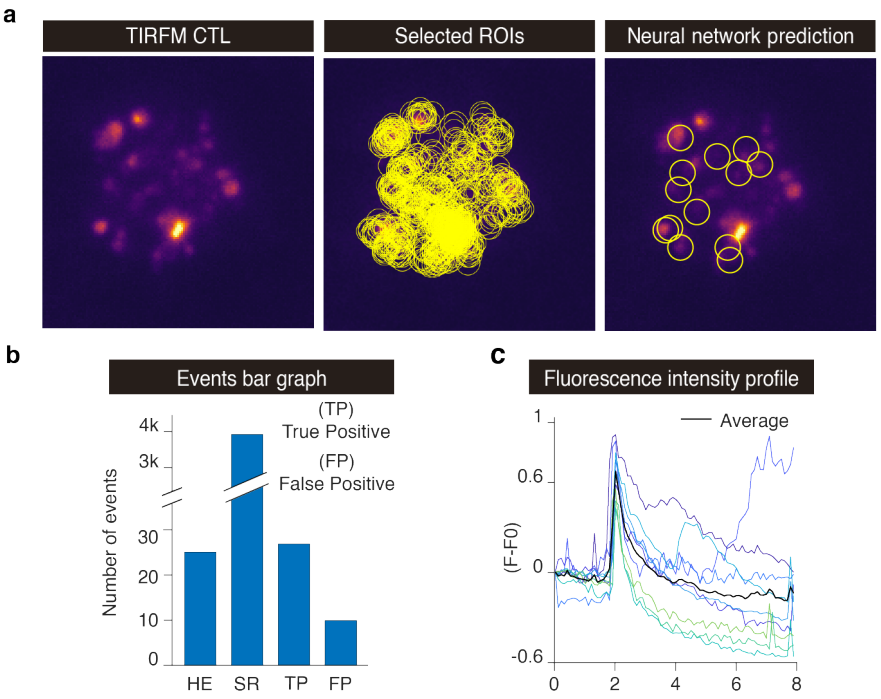


Figure 26. Analysis of lytic granule exocytosis in CTLs labeled with LysoTracker Red (adapted from (Chouaib et al., 2025)).

a. Raw TIRF microscopy image of a CTL with lytic granules labeled using LysoTracker Red. The second panel shows the overlay of selected regions of interest (ROIs) prior to classification. The third panel displays the neural network output, with overlays marking events predicted as true positives. **b.** Bar graph comparing the outcomes of the human expert (HE) with IVEA, including the number of selected ROIs (SR). **c.** Fluorescence intensity profiles of representative fusion events identified as true positives. Data were acquired at the Cancer Research Center of Toulouse, France.

Building on the previous evaluation, we next examined IVEA's performance on additional datasets that represented distinct labeling strategies and fluorescence behaviors. Specifically, the eViT model was tested on CTL-CD63-pHuji and CTL-CD9-SEP recordings, which differ from the earlier granzyme B-based datasets in that the labeled granules are not visible before fusion. In these experiments, the CD63-pHuji or SEP fluorophores are targeted to the membranes of multivesicular bodies and exosomes. Consequently, granules remain dark until exocytosis exposes the lumen to the extracellular environment, where the fluorophore rapidly dequenches, generating a sudden, high-contrast fluorescence spike against a low background (**Figure 24c**). This signal type contrasts sharply with that of pH-sensitive cargo fusions such as granzyme B–pHuji, tdTomato, or mNeonGreen, where granules are visible prior to fusion and display gradual intensity fluctuations while moving inside the cell. The abrupt fluorescence onset in CD63- and CD9-labeled samples explains why rule-based algorithms such as ExoJ perform well under these conditions (Liu et al., 2024). Such software is optimized for threshold-based detection of sharp signal transitions. In contrast, the earlier GranuVision2 model, which was trained on visible, pre-fusion granules, showed reduced recall in these datasets, as its learned temporal features reflected more gradual exocytic patterns.

To expand IVEA's applicability across labeling strategies, the eViT network was retrained on combined datasets including CD63- and CD9-labeled cells. This new model, termed GranuVision3, learned to recognize both exocytosis for visible and non-visible granule types and their distinct fluorescence kinetics. GranuVision3 demonstrated improved accuracy across all tested probe types and cell models, integrating the strengths of prior networks while extending coverage to nearly all exocytosis signatures observed. This retraining established GranuVision3 as the default model in IVEA (version 2.3), now capable of categorizing an additional fusion event type.

Following the inclusion of CD63- and CD9-labeled recordings in the training set, we evaluated the generalized GranuVision3 model against ExoJ on both local and external CD63-pHuji datasets. ExoJ reached a value of 88.92 ± 7.28% for the F1 score, approaching the performance of GranuVision3 (98.16 ± 1.07%). However, its performance declined markedly on the pH-sensitive GzmB–pHuji–labeled dataset (**Figure 25a)** and dropped to near-complete failure on pH-insensitive datasets (**Figure 25 b, d, e**; **Table 6–7**), even after extensive parameter optimization (**Table 8**). Consistent with the measured metrics, the precision, recall, and F1 scores were significantly higher for eViT compared to ExoJ in nearly all probe and cell-type combinations, with p-values below 0.001 in most cases (**Table 7**). This limitation reflects ExoJ's reliance on fixed rule-based detection criteria, which lack the flexibility to adapt to variable signal types, whereas IVEA's deep learning framework maintains consistent performance.

Table 7 (related to Fig. 3): Statistical analysis for the eViT, LSTM, and the ExoJ comparison (adapted from (Chouaib et al., 2025)).

The values shown are the p-value of the analysis performed on the video's sets. ANOVA on rank with Dunn's post-test is indicated by *, ANOVA with Holm-Sidak post-test by # and paired two-tailed Student's t-test by §. Non-significant are labeled as ns. When the p-value is below 0.2, the exact value is given in brackets. Comparisons that could not be performed due to insufficient data are marked with //.

	eViT to LSTM	eViT to ExoJ	LSTM to ExoJ	N
CTL expressing pH-sensitive probe (Fig. 3a)				
Recall *	ns	<0.001	0.001	13
Precision *	ns	ns	ns (0.078)	13
F1 score *	ns	<0.001	0.001	13
CTL expressing pH-insensitive probe (Fig. 3b)				
Recall *	ns	<0.001	0.020	7
Precision #	ns (0.155)	<0.001	<0.001	7
F1 score *	ns	<0.001	<0.001	7
CTL expressing CD63 for long-lasting events (Fig. 3c)				
Recall *	ns	ns	ns	5
Precision #	ns	ns	ns	5
F1 score *	ns	ns	ns	5
Chromaffin or INS1 cells expressing pH-insensitive probe (Fig. 3d)				
Recall *	0.033	<0.001	0.033	10
Precision *	ns (0.126)	0.016	<0.001	10
F1 score §	0.002	//	//	10

INS1 cells expressing probes with little pH-sensitivity (Fig. 3e)				
Recall *	0.030	<0.001	ns	8
Precision #	ns	<0.001	0.002	8
F1 score §	0.00000793	//	//	8

Table 8 (related to Figure 24a): ExoJ analysis parameter iterative adjustment (adapted from (Chouaib et al., 2025)).
ExoJ provides ten adjustable parameters for event detection. The first eight (ranging from minimum points for fitting procedure to maximum displacement) did not influence detection outcomes. In contrast, the final two parameters, the minimum R^2 for the decay fitting procedure (DFP) and the minimum R^2 for the estimated radius fitting procedure (RFP), substantially affected detection performance. Parameter tuning was performed iteratively, beginning from default values, first adjusting RFP and subsequently DFP until optimal detection was achieved. The numbers shown here are the true positive (TP), false positive (FP), and false negative (FN) events. The evaluation was conducted on a representative movie from Figure 24 set 1.

Metric	ExoJ							eViT
	Default 0.9 – 0.75	DFP-RFP 0.9 – **0.4**	DFP-RFP 0.9 – **0.5**	DFP-RFP 0.9 – **0.6**	DFP-RFP **0.8** – 0.5	DFP-RFP **0.9** – 0.5	DFP-RFP **0.95** – 0.5	Default
TP	3	5	5	3	5	5	2	14
FP	1	11	5	1	9	5	1	2
FN	11	9	9	11	9	9	12	0

As part of the same external benchmarking, we also evaluated IVEA against pHusion, another mathematical rule-based tool designed for vesicle fusion analysis. Unlike IVEA, pHusion is restricted to Apple macOS systems and operates by chaining together a sequence of image-processing scripts. In our benchmark tests on CTLs labeled with the pH-sensitive probe GzmB–pHuji, pHusion consistently underperformed compared with ExoJ, frequently yielding no detections ("//" in **Table 9**) or producing results that could not be improved through parameter tuning.

Table 9: Performance comparison between eViT and pHusion on CTL datasets labeled with the pH-sensitive probe GzmB-pHuji (Figure 24a) (adapted from (Chouaib et al., 2025)).
For pHusion, cells where no events were detected are indicated with "//". The grid spacing was adjusted to 0.11 μm (matching the pixel size), and the "good trace" criterion was set to minimum R^2 for the estimated radius fitting value of 0.9; all other parameters were kept at default settings. Results are displayed in the "traces collapsed to a single event" mode. The numbers shown are the true-positive (TP), false-positive (FP), and false-negative (FN) events identified by each method.

	pHusion software			IVEA (eViT)			HE
Cells	TP	FP	FN	TP	FP	FN	TP
1	//	//	//	4	0	0	4
2	//	//	//	4	0	0	4
3	//	//	//	4	0	0	3
4	//	//	//	2	0	0	2
5	0	0	16	16	0	0	14
6	//	//	//	10	1	0	7
7	1	3	3	4	0	0	2
8	14	2	0	14	2	0	14
9	6	3	2	8	1	0	8
10	4	3	3	7	0	0	7
Total	25	11	24	73	4	0	65

During this evaluation, the vision radius for event classification was adjusted based on granule size and microscope resolution. For example, for recordings with 110 nm pixel size we used radii between 14–

16 pixels (**Figure 24a–c**), while for those with 130–160 nm pixel size we used radii between 7–12 pixels (**Figure 24d–e**). The vision radius, which defines the spatial window for classification, strongly influenced detection outcomes. The vision radius defines the spatial window provided to the eViT for event classification. It must be carefully matched to the approximate size of the exocytic signal, as it strongly influences detection outcomes. In tests performed on representative videos from each dataset, smaller radii increased sensitivity for detecting small fusion events, while larger radii were more suitable for events with broader fluorescence spread. As an example, in a CTL video with relatively small fusion signals, a radius of 8 pixels yielded 38 detections compared to 26 detections at 14 pixels (**Figure 27**). This illustrates that inappropriate radius selection can either truncate event features or dilute them with excessive background, directly influencing the number of detected events.

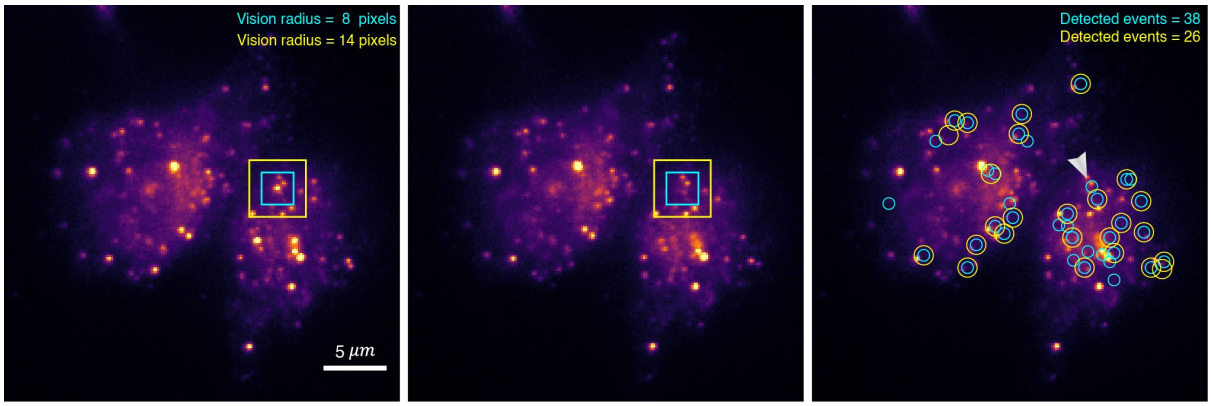


Figure 27. Effect of vision radius selection on exocytosis detection (adapted from (Chouaib et al., 2025)). Two vision radii, corresponding to the neural network input patch size, were evaluated: 8 pixels (cyan) and 14 pixels (yellow). In the first panel, a representative granule is centered within both masks, indicating the region analyzed for classification. The second panel illustrates the disappearance of the granule as it undergoes exocytosis. The third panel compares the detected events for the two radii, with 38 events identified using the 8-pixel radius and 26 using the 14-pixel radius. These results demonstrate how the chosen vision radius modulates detection sensitivity. Each pixel corresponds to 160 nm in physical space.

IVEA substantially reduced the time required for analysis. Human experts had to inspect each cell individually, often reviewing long recordings frame by frame to confirm subtle signal changes. This procedure also proved prone to oversight, particularly in videos containing dim or low-contrast fusion events. For instance, analyzing CTLs labeled with pH-insensitive probes required roughly 10 minutes per cell, while cells labeled with pH-sensitive probes could be reviewed in about 5 minutes due to their sharper signal dynamics. Consequently, a single 10-minute video containing 30 cells would demand nearly 300 minutes of manual review. In contrast, IVEA processed an equivalent dataset, a 256 × 256-pixel video of ~3000 frames in under one minute per cell. On an Intel Core i9 (10th generation) workstation, this corresponded to roughly 15 minutes total for the 30-cell dataset, regardless of the fluorescent label type.

In addition, IVEA provides a structured and reproducible output format designed for quantitative analysis and transparent reporting. For each analyzed video, the software generates an output folder containing two subdirectories and a log file. The measurements subdirectory stores event-based data in comma-separated value (CSV) format. One file contains the fluorescence intensity profiles for all detected events, extracted over a user-defined temporal window (e.g., 50 frames before and 100 frames after the fusion frame). Each event is represented as a column, allowing straightforward plotting or multigraph visualization in programs such as Excel. A second CSV file stores quantitative event metrics, including event ID, spatial coordinates (x, y), peak intensity, frame index, and kinetic parameters such as rise and decay times. The rise time is defined as the interval between 90% and 10% of the peak amplitude (measured backward from the peak). The decay time is obtained by fitting each event's intensity trace with a mono-exponential decay function and measuring the time to reach 1/e (≈ 37 %) of the peak-above-baseline amplitude. These parameters characterize the temporal kinetics of exocytosis and the relaxation dynamics of the fluorescence signal. The second subdirectory contains ImageJ ROI files (zip) that include all classified events with their corresponding identifiers, spatial coordinates, and time points. These files can be directly imported into the ImageJ ROI Manager, allowing users to navigate instantly to each detected event for visual inspection or validation. Optionally, if the ROI export mode is enabled, IVEA also generates cropped image patches of the detected regions labeled by event category (**Table 4**), which can serve as training data for model refinement or transfer learning. Finally, the log file summarizes all key metadata, including the video name, user-defined and automatically inferred parameters, the number of detected events, and software settings. This file acts as an internal archive, ensuring full reproducibility of the analysis and allowing identical reprocessing of datasets using the same parameters.

In summary, the eViT consistently outperformed the LSTM and ExoJ across cell types and probes. Conversely, ExoJ exhibited competitive performance only for latent fusion in pH-sensitive datasets, and pHusion failed to achieve comparable reliability or detection rates. IVEA's deep learning models, especially the eViT model, are trained directly on spatiotemporal patterns of true fusion, enabling them to generalize across probes and imaging conditions. Whether vesicles are invisible until fusion or visible throughout their trajectory, the network learns to recognize the characteristic features of exocytosis. This provides a consistent detection where other methods fail.

3.5 Transfer learning on calcium sparks data

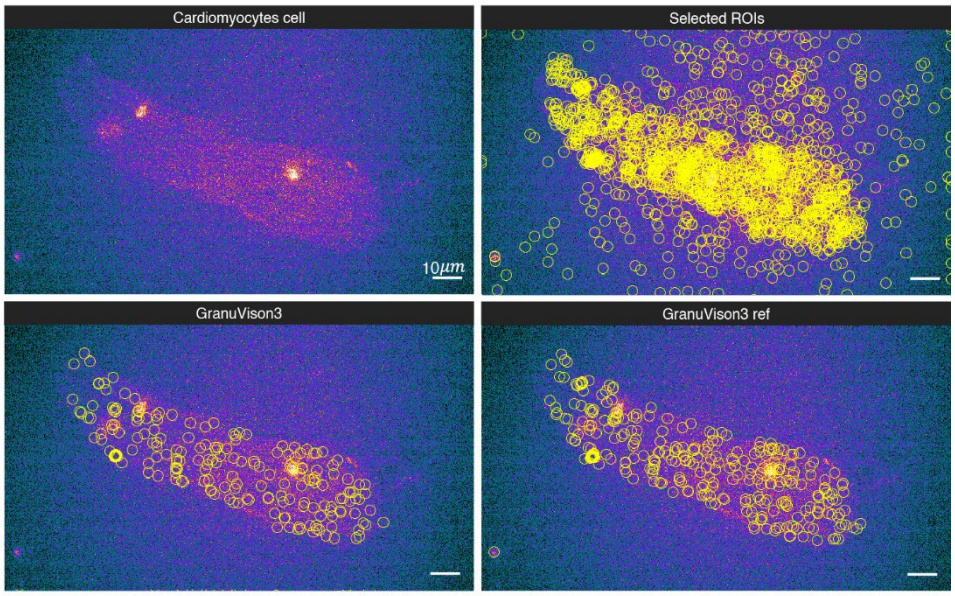
The potential of transfer learning within the eViT framework was evaluated using calcium spark datasets acquired by Fluo-4 imaging in cardiomyocytes (Tian & Lipp, 2021). A limited dataset consisting of 30 calcium spark events was used for refinement of the GranuVision3 model, replacing the original third classification category, latent vesicle fusion (class "2"), with calcium sparks. This substitution was motivated by the similarity in visual features: calcium sparks, like latent fusion events,

emerge abruptly without a visible precursor structure. Training was performed over 10 epochs using the IVEA Python implementation. Among several available functions, the refine "GranuVision3" option was selected for this experiment (**Figure 28**). Due to the small dataset size and short training schedule, the retraining process required less than one minute on a GPU.

Following refinement, the updated GranuVision3 model was evaluated on the same calcium spark dataset to assess its adaptability and generalization capacity. Post-training inference demonstrated a marked improvement in detection performance: 298 true-positive sparks were identified with only 2 false positives, compared to 200 true positives prior to refinement. Importantly, this gain was achieved without compromising model stability or introducing spurious detections.

This experiment highlights the versatility and robustness of IVEA's transfer learning framework, showing that minimal retraining with small, user-labeled datasets can effectively repurpose the model for new biological phenomena. Beyond vesicle exocytosis, this adaptability positions IVEA as a broadly applicable computational platform for analyzing diverse dynamic cellular processes, including calcium signaling.

a



b

IVEA parameters used in this analysis	Model	GranuVision3	GranuVision3 ref
Vision radius: 12 Search radius: 4 Prominence: 30 Event spread: 1 Model confidence: 0.5 Sensitivity: 0.5	TP	200	298
	FP	1	2
	Selected ROIs	1558	1558

Figure 28. GranuVision3 refinement training on calcium sparks results (adapted from (Chouaib et al., 2025)). **a.** Representative image of a mouse cardiomyocyte displaying calcium sparks detected via Fluo4-based fluorescence imaging. The top-left panel shows a single frame capturing multiple calcium spark events. The top-right panel illustrates the regions of interest (ROIs) selected prior to processing with the neural network. The bottom-left panel overlays IVEA-detected ROIs on the raw data using the original GranuVision3 model, while the bottom-right panel shows the corresponding output following model refinement. All images share a pixel resolution of 0.217 × 0.217 µm and were acquired at a frequency rate of 124 frames per second. **b.** Quantitative

summary of IVEA detection performance using the original and refined GranuVision3 models. Refinement led to a substantial increase in true positive (TP) detections, from 200 to 298, while introducing only one additional false positive (FP). The number of candidate ROIs generated by IVEA was fixed at 1558, confirming that the observed improvement arose solely from the refined classifier rather than from changes in detection parameters.

3.6 Stationary burst events analysis (module 2)

Analysis of stationary burst events (SBEs) was carried out through the LSTM-based module of the IVEA framework (Fig. 2a,c). Datasets consisted of recordings from dorsal root ganglion (DRG) neurons expressing the synaptic vesicle pHluorin-based reporter SypHy. They were acquired at 10 Hz with approximately 3,000 frames per movie and image dimensions of either 512 × 512 pixels or 512 × 256 pixels. Recordings were obtained from two published sources (Shaib et al., 2018; Staudt et al., 2022). The datasets included experiments with 1-minute electrical stimulation and dual stimulation episodes of 30 s and 1 minute separated by a 10 s recovery interval, providing distinct event kinetics across stimulation paradigms. In these recordings, stationary bursts appeared as spot-like signals with a rapid onset. The typical rise time was approximately 4.1 s (≈ 41 frames at 10 Hz **Figure 29 a–f**, while the random burst event LSTM model is 21 frames (**Figure 29 g–i**). To capture these dynamics, the default time-series window length is set to 41 frames, enabling the LSTM to learn the temporal features of vesicle exocytosis in DRG neurons. For slower events (**Figure 30b**), the number of frames could be extended by adding additional frames after the detected event. The initial number of frames preceding the event was invariably set at 10. For example, selecting a 100-frame window yielded 10 pre-event frames and 90 post-event frames, plus the event frame itself (101 frames total). This ensured that temporal features remained aligned while accommodating variable event kinetics. In contrast to the preprocessing, the LSTM input layer was fixed at 41 frames to cover the average normal fusion time (**Figure 30a**), and therefore longer sequences were resampled to this length during preprocessing (**Figure 30c**).

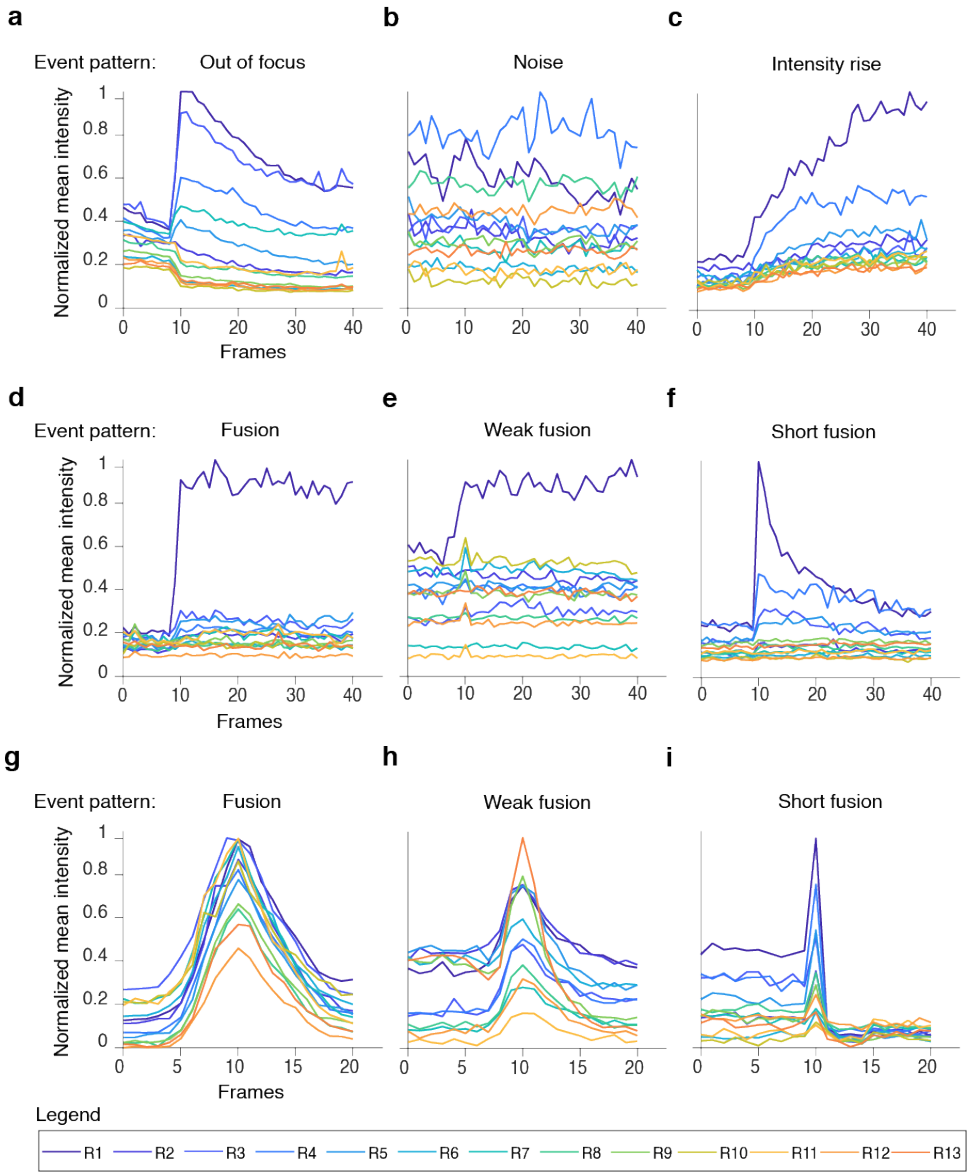


Figure 29. Events' 13 regions mask patterns imported to the LSTM network (adapted from (Chouaib et al., 2025)).

These regions' temporal patterns represent the DRG neurons extracted image patches around different event types (**a–f**), while (**g–i**) represent patterns for events acquired from CTL image patches. a. Patterns of the out-of-focus artifact class. **b.** Random white noise. c. slow intensity rise. **d.** Classic fusion event. **e.** Weak fusion event. **f.** short fusion event. **g.** T cell vesicle fusion event. **h.** T cell weak fusion event. **i.** T cell short fusion event around 1 frame length (0.1 sec at 10 Hz).

During the detection stage, the automatic parameters generated a large pool of candidate regions of interest per movie (up to ~30,000). These ROIs were subsequently classified by the LSTM. Because stimulation often induces widespread intensity fluctuations either globally across the field of view or locally within activated neurons, a dynamic sensitivity threshold was implemented. This threshold adjusted automatically in response to changes in image brightness, increasing proportionally as fluorescence levels rose during stimulation. As a result, the detection sensitivity decreased, reducing the number of ROIs identified during high-intensity periods, thereby preventing the accumulation of excessive detections. Users can also manually specify stimulation intervals through the graphical interface to further control threshold behavior during defined periods. The software is set to detect the

stimulation periods automatically. However, manual input proved useful in cases where neuronal activation was partial or absent, and the automatically inferred stimulus onset was insufficient. For instance, entering "300-0_700-1200_0-0" would instruct IVEA to parse three stimulations: one beginning at frame 300 with automatically inferred termination, one from frame 700 to 1200, and one left entirely for automatic detection. While this automatic functionality addresses most stimulation paradigms, its accuracy was limited by video fluorescence intensity fluctuations.

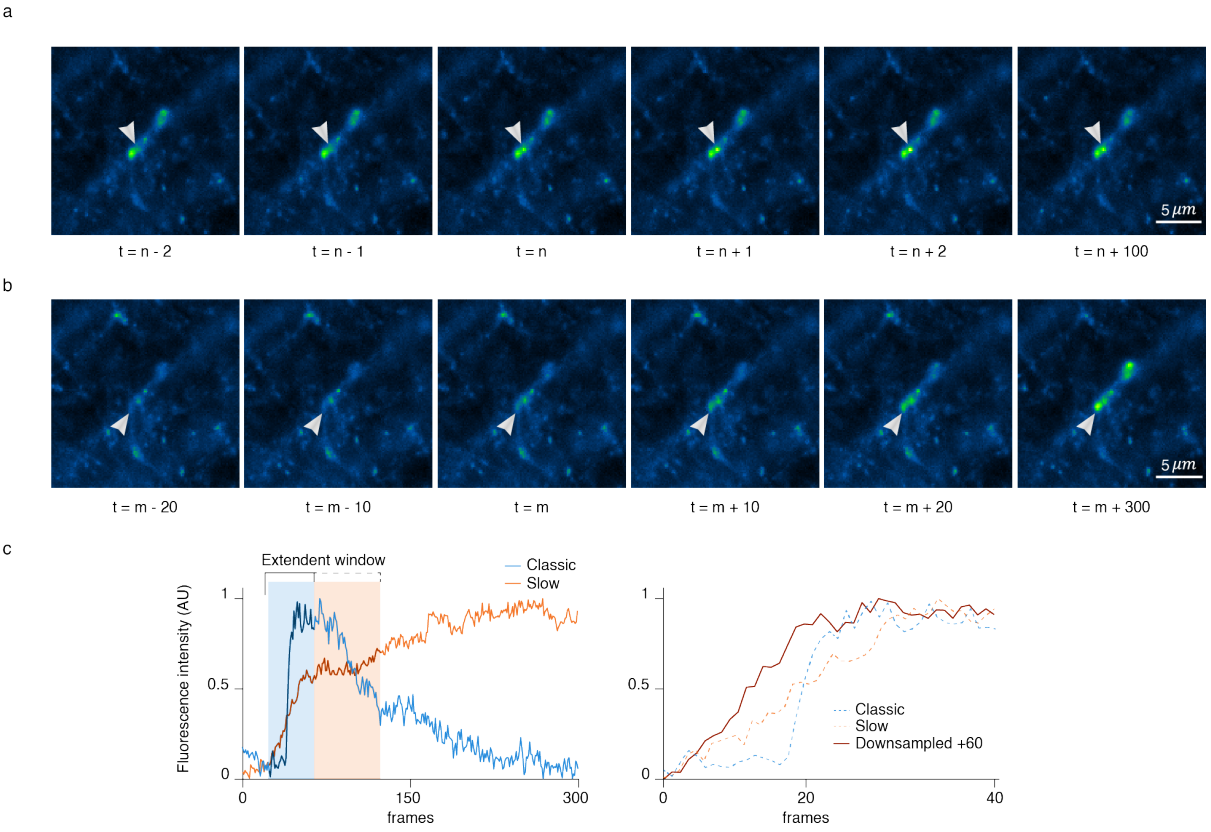


Figure 30. Slow vs classic fusion events (adapted from (Chouaib et al., 2025)).
a. Time-lapse series depicting a representative classic fusion event, indicated by arrowheads. **b.** Time-lapse series displaying a slow fusion event. **c.** Intensity profiles extracted over 300 frames. In the left panel, the blue shaded area denotes the default measurement window, while the orange shaded area illustrates an extended measurement window adjusted by the user to capture prolonged signal dynamics. The right panel compares downsampled temporal profiles, showing differences between a typical classic fusion event and a slow-rising intensity event. Together, these panels highlight the temporal heterogeneity of synaptic transmission kinetics, with slow fusion events requiring extended analysis windows to fully capture their gradual signal rise.

Given the high level of vesicle motion in DRG recordings, a spatiotemporal grouping step was introduced before classification. ROIs were clustered within a user-defined spatial radius of 3 pixels per frame and separated temporally by 30 frames following an event. This procedure reduced redundancy in the candidate pool down to approximately half the original ROIs (~14,000), while preserving biologically distinct events. Because in DRG neurons, stationary bursts did not spatially spread (**Figure 30a, b**), the Gaussian non-maximum suppression method was unnecessary, and simple

radius-based clustering was sufficient. In addition, recurrent activations at the same spatial location were registered to allow quantification of repeated events at fixed synaptic sites.

Initial benchmarking and iterative refinement of IVEA were conducted using published datasets in which synapses were pre-annotated. The first training batch was generated using IVEA's automatic parameter estimation and threshold-based detection. During development, exported events were visualized as a set of 13 representative intensity traces each, allowing inspection of network behavior and identification of distinct fusion patterns. These visualizations guided the definition of new event categories and the removal of ambiguous ones in successive training rounds. After several cycles of retraining and optimization, the LSTM was trained to classify nine event categories. These comprised four positive classes: regular fusion events lasting 20–60 frames (2–6s at 10 Hz), short fusion events lasting approximately 4 frames (0.4s at 10 Hz), responsive fusion events (electrical or agonist stimulation), and NH_4^+ stimulation events. The remaining five categories constitute the negative classes, including noise, out-of-focus artifacts, vesicle displacement, white noise, and fluctuations in fluorescence intensity (**Figure 29 a-f**). Training utilized 39 videos across multiple genetic backgrounds (single knockout, double knockout, and wild type), all labeled with SypHy to visualize exocytosis. The optimized network successfully generalized across datasets, accurately capturing the temporal patterns of vesicle fusion under different experimental conditions.

The ability of the LSTM to differentiate event categories enabled IVEA to annotate exported events by duration (short, normal, long) and by stimulus relationship (synchronized vs. non-synchronized). This classification was integrated into the ImageJ ROI Manager, where event metadata (ID, timing, status) is embedded in the ROI labels, facilitating downstream inspection by users (**Figure 31 b, d, e**).

For quantitative evaluation, 11 previously unseen DRG recordings were analyzed. A human expert (HE) manually annotated 705 fusion events (**Figure 31a**). In comparison, IVEA initially registered ~84,000 ROIs, the vast majority of which were discarded by the LSTM. Ultimately, 2,049 events were classified as true positives with 70 false positives, yielding a mean recall of 88.12 ± 2.70%, precision of 96.37 ± 0.45%, and F1 score of 91.83 ± 1.61% (**Figure 31c**).

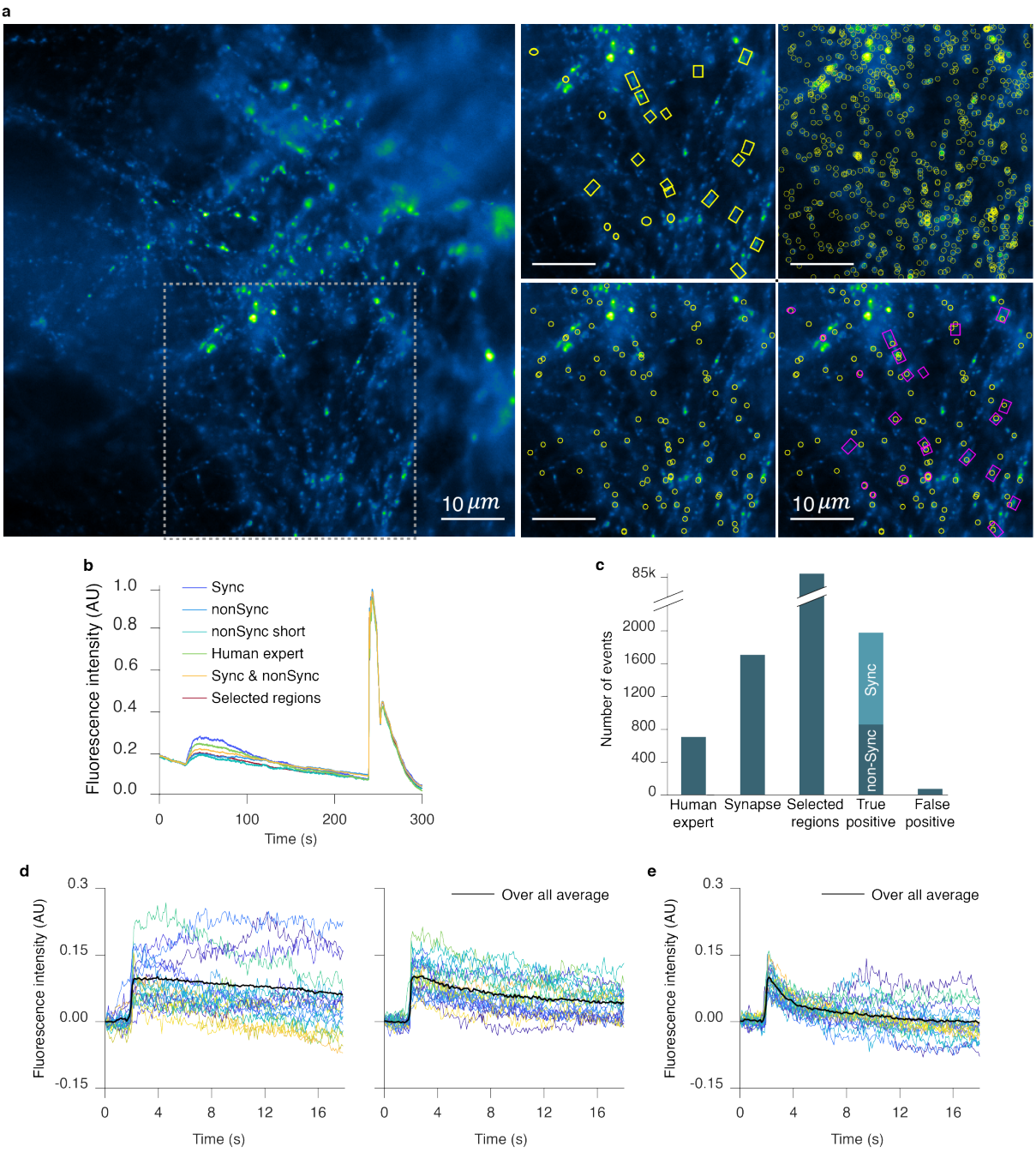


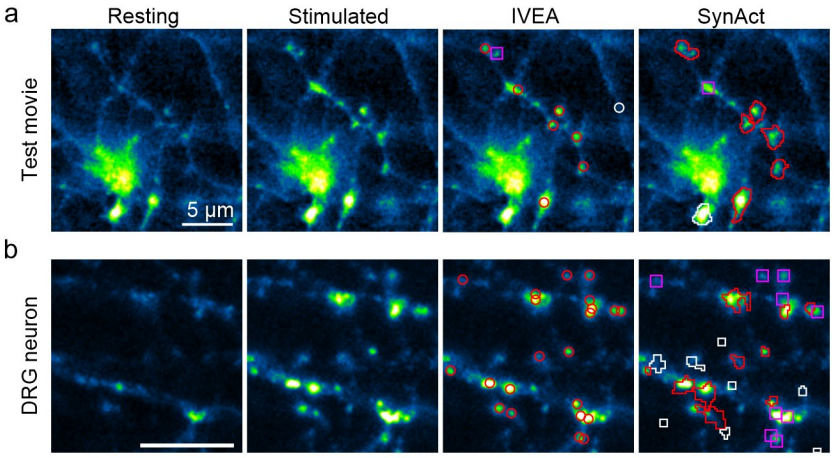
Figure 31 Automated vs manual detection of synaptic events in DRG–spinal cord co-cultures (adapted and modified from (Chouaib et al., 2025)).

a. Left: Raw TIRF microscopy image representing dorsal root ganglion (DRG) neurons overexpressing SypHy, forming synaptic connections with spinal cord neurons. Right: The magnified region (dashed box) illustrates detected regions of interest (ROIs) obtained using different approaches. Shown are ROIs annotated by a human expert (HE), selected regions (SR) derived from statistical detection, neural network–predicted ROIs, and a merged overlay comparing HE and neural network ROIs, colored as magenta and yellow, respectively. Scale bars, 10 µm. **b**. Average intensity profiles of all detected ROIs, color-coded according to event categories as indicated. **c**. Quantification of total events detected across 11 DRG neuron recordings using IVEA with default analysis parameters. **d**. Mean fluorescence traces of synchronized (left) and non-synchronized (right) events relative to the stimulation time point, showing temporal alignment and signal evolution. **e**. Mean fluorescence traces for short-duration events. Colored lines correspond to individual events, with the bold black line denoting their mean profile.

Notably, the LSTM frequently detected weak or transient events that were difficult to annotate manually, while very slow-rising signals (> 41 frames) were temporarily placed in an "intensity-rise" holding category **Figure 29c**. Extending the analysis window (e.g., +60 frames for a total of 101) allowed these events to be recovered into the true-positive pool under conditions of slow stimulation kinetics (**Figure 30 b, c**).

Manual expert review required ~60 minutes per recording, whereas IVEA completed automated analysis in under one minute per video. Batch processing further eliminated repeated parameter tuning by automatically adapting to the properties of each input movie.

Performance was also compared to SynActJ, an open-source ImageJ plugin distributed via GitHub. SynActJ consists primarily of image-processing routines to extract intensity variations in DRG recordings. On test movies provided by the developers, both IVEA and SynActJ successfully detected most active synapses, each missing one event and producing one false positive (**Figure 32a, c**). However, when applied to our DRG datasets, IVEA identified all human-annotated events without false positives, whereas SynActJ detected only a limited and seemingly random subset, accompanied by numerous false positives (**Figure 32b, c**).



c

SynActJ Settings		Test movie	DRG neuron
Background	Gauss sigma	4	4
	Selected threshold	MinError	MinError
Boutons : Detect	LoG sigma	0.5	0.5
	Prominence	0.005	0.005
Segment	Gauss sigma	1.0	0.5
	RollingBall radius	30	20
	Selected threshold	Triangle	IJ-IsoData
	Erode Mask	OFF	ON

Figure 32 Comparison of IVEA and SynActJ for active synapse detection (adapted from (Chouaib et al., 2025)). (**a–b**) Representative examples illustrating detection performance in two datasets. From left to right: resting condition, stimulated condition, and with the overlaid IVEA detection and SynActJ detection. Detected regions of interest (ROIs) are color-coded: red outlines indicate true positive events, white outlines denote false positives, and magenta squares represent missed events. **a.** SynActJ test movie provided on GitHub. For IVEA analysis, the sequence length was extended to 60 frames by duplicating the first and last frames, while SynActJ was run with its default recommended settings. **b**. Movie acquired from DRG neurons overexpressing SypHy stimulated at frame 60. For this dataset, IVEA was applied directly to the raw 300-frame recording. In contrast, SynActJ required

preprocessing with frame reduction and Gaussian blurring ($\sigma = 0.5$) to detect events. SynActJ parameters were further adjusted iteratively to improve detection performance; furthermore, the detection seems random. **c.** SynActJ parameter settings used for the analyses in (**a**) and (**b**). IVEA was run with default parameters for (**b**), while in (**a**) automatic parameter adjustment was disabled, prominence was set to 10, and sensitivity to 1.

For downstream analyses, events were organized and saved as zip files relative to stimulation (synchronized vs. unsynchronized), based either on user-defined or automatically inferred timings. Outputs included labeled ROI files (event ID, frame, classification) and per-event summary tables containing intensity measurements over user-specified intervals and across the entire movie, saved as CSV files to load in Excel easily.

To further accommodate the long temporal windows typical of neuronal recordings, a temporal max pooling procedure was introduced at the detection and data-collection stage. This compressed the time axis by replacing user-defined frame windows with their maximum intensity projection, thereby reducing data length while preserving peak signal information. For example, a pooling factor of two halved the number of frames, while a factor of three compressed three consecutive frames into one. Importantly, this compression was applied only during event detection and ROI collection and did not alter the raw movies or the exported intensity measurements. Temporal max pooling substantially reduced computational load and proved particularly advantageous when preparing inputs for transformer-based models.

Finally, in parallel with the LSTM-based classification, an experimental eViT architecture was adapted for stationary burst analysis. The encoder was modified to handle extended temporal windows while reducing spatial dimensions. Preliminary training of this model on the stationary burst dataset yielded promising results, although further optimization and head-to-head comparison with the LSTM remain ongoing (see subsequent results).

3.7 Evaluating IVEA using downsampling analysis

To assess the versatility and robustness of IVEA across varying image acquisition frequencies, an additional evaluation was performed utilizing downsampling analysis. Given that IVEA models were initially trained on datasets acquired at 10 Hz, this analysis explored whether retraining on datasets acquired at different frequencies would be necessary or if existing pretrained models could be directly applicable. Specifically, original 10 Hz acquisition videos were downsampled to 1 Hz to evaluate IVEA performance at significantly reduced acquisition rates. The results indicated that IVEA effectively detected exocytosis events even at lower acquisition frequencies; however, the detection of rapid fusion events was naturally diminished due to the inherent loss of temporal resolution, rendering such events undetectable both by manual inspection and computational analysis (**Figure 33a–c**). Furthermore, for datasets acquired at higher frequencies, a temporal max pooling function with an adjustable window size was implemented. This function enables users to systematically downsample video data, preserving critical temporal features that would otherwise be lost in conventional downsampling approaches.

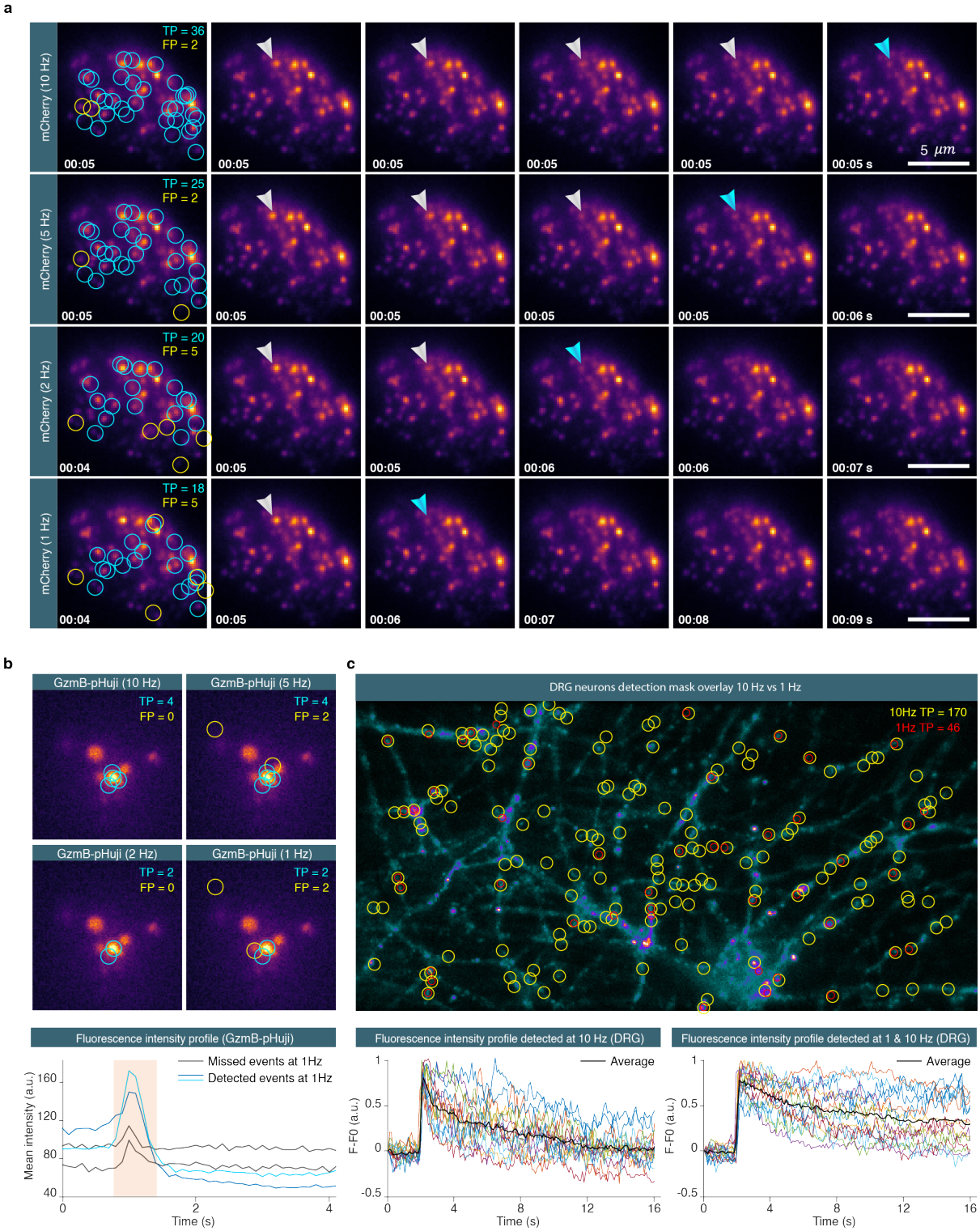


Figure 33. Effect of acquisition frequency on IVEA detection performance (adapted from (Chouaib et al., 2025)).

a. Time-lapse sequences of chromaffin cells expressing NPY-mCherry at four acquisition frequencies (10, 5, 2, and 1 Hz). The first image in each row shows the raw data overlaid with IVEA-identified regions of interest (ROIs): cyan circles indicate true positives (TP) and yellow circles indicate false positives (FP). Subsequent frames depict the temporal progression of a representative exocytosis event (arrowheads) captured at each sampling rate. **b.** Detection results in CTL expressing GzmB-pHuji acquired at 10 Hz and downsampled to the same four frequencies as in (**a**). ROIs are color-coded as in (**a**). The accompanying fluorescence intensity profiles (see below) demonstrate a progressive loss of temporal fidelity with decreasing acquisition frequency, resulting in the omission of events and, consequently, a reduction in accuracy at 1 Hz. **c.** Comparative detection of exocytosis events in DRG neurons

imaged at 10 Hz and downsampled to 1 Hz. Cyan ROIs correspond to TP events detected at 10 Hz, while red ROIs correspond to TP events detected at 1 Hz. The lower panels display fluorescence intensity traces: left, events captured at 10 Hz but not at 1 Hz; right, events consistently detected at both frequencies. Collectively, these data demonstrate that high acquisition frequencies are essential for capturing rapid fusion events, while reduced frame rates lead to underestimation of event number and distortion of temporal dynamics.

Temporal max pooling proved particularly advantageous for analyzing recordings containing slower biological events. This operation is integrated into both the random and stationary burst modules, where it condenses temporal information into a representation more compatible with the temporal receptive field of the neural network, particularly the eViT model. In preliminary tests on slower recordings without temporal reduction, both GranuVision2 and GranuVision3 failed to identify any events. When temporal max pooling was applied with a reduction factor of five, the model successfully detected the same four fusion events that had previously been identified using the Reduce function in ImageJ (**Figure 33b**). Unlike the ImageJ Reduce function approach, which physically downsampled and shortened the video, temporal max pooling is implemented internally within IVEA as part of the preprocessing pipeline. It accelerates event detection while maintaining the original temporal structure of the data. The detected event positions are automatically adjusted to correspond to their correct frame indices in the raw video, ensuring consistency across the full dataset. This demonstrates that temporal max pooling enhances computational efficiency and model compatibility with slow-dynamic datasets while preserving the integrity of the original recordings.

3.8 Granule detection and tracking

Accurate detection and tracking of secretory granules are essential for characterizing the spatiotemporal dynamics of exocytosis. To capture vesicle trajectories preceding membrane fusion, a dedicated tracking module was implemented within the Python version of IVEA (IVEA-Py). This functionality allows visualization and quantification of granule motion prior to exocytosis, particularly for random burst events. Tracking can be optionally enabled depending on image quality and granule visibility. The Python implementation was chosen for this module due to its efficiency in vectorized numerical computation and its flexibility for integrating advanced tracking algorithms.

Three approaches for fluorescence granule localization were investigated: the ComDet and ExoJ plugins (Liu et al., 2024) for ImageJ/Fiji (Rueden et al., 2017; Schneider et al., 2012), and the Gradient Vector Flow (GVF) flux method (G. Li et al., 2007). Each approach was adapted and benchmarked to assess suitability for fluorescence granule detection (**Figure 36**).

ComDet detects bright spots through Laplacian- or top-hat filtering followed by global thresholding at three standard deviations above the mean intensity. Although computationally efficient, this approach missed dim vesicles under heterogeneous illumination. ExoJ employs an à trous wavelet transform (Olivo-Marin, 2002) combined with MAD-based thresholding and Gaussian fitting to refine detections (Liu et al., 2024). While this improved robustness to noise, sensitivity declined in regions of locally

variable background. The GVF flux method instead analyzes the diffused image gradient field, identifying blob centers from the negative flux minima of the vector field. This approach effectively separated adjacent vesicles and maintained accuracy under high noise conditions but remained sensitive to flux-threshold selection. Based on these methods IVEA-Py adopted a hybrid detection framework that integrates advantageous features from all mentioned methods (**Figure 34–35**). Each frame is first denoised with a median filter and enhanced using Laplacian- or Difference-of-Gaussian convolution to emphasize vesicle-like structures. A low MAD-based threshold generates a binary mask for candidate regions, which is then refined using gradient vector diffusion. Local convergence points within candidate regions are determined via Euler integration combined with an iterative voting scheme (**Figure 35** 1st image 2nd row), improving detection sensitivity under low-contrast conditions.

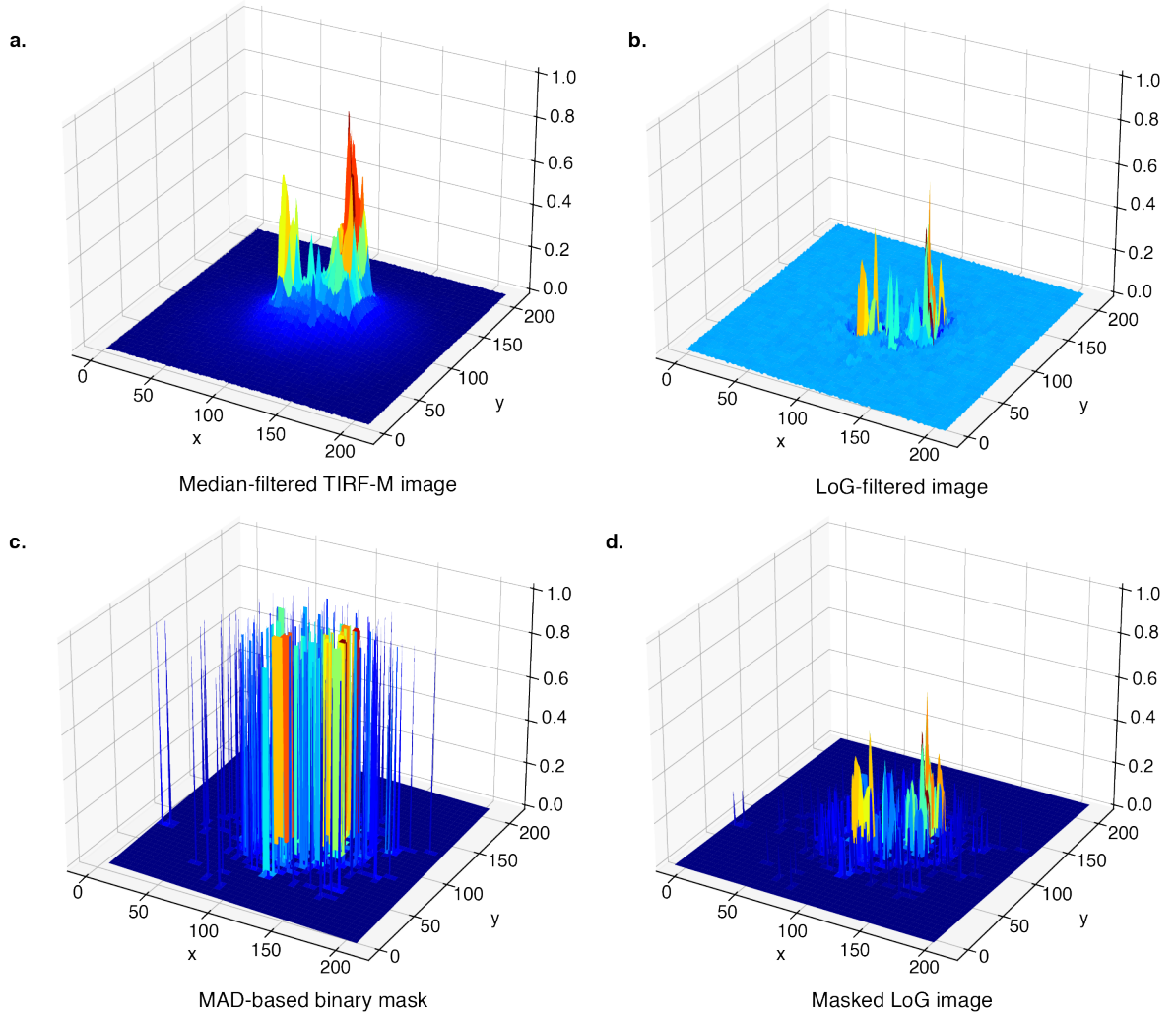


Figure 34. Image preprocessing pipeline prior to GVF field computation.
a. Median-filtered fluorescence image. **b.** Laplacian-of-Gaussian (LoG) filtered image obtained via 2D convolution. **c.** Binary mask generated from the LoG response using a global MAD-based threshold. **d.** Masked LoG image $h(x, y)$ obtained by multiplying the LoG response with the binary mask to suppress background noise before GVF computation.

Instead of flux thresholding as in the original GVF formulation, IVEA-Py first computes the normalized gradient field (g_x, g_y) (**Figure 35,** 2nd, 3rd images) from the masked image $h(x, y)$ (**Figure 34d**), and

then applies forward Euler integration along (g_x, g_y) to drive trajectories toward local intensity centers. This allows trajectories to converge toward local intensity centers even when objects are closely spaced. The endpoints of these trajectories are clustered using DBSCAN to identify convergence points corresponding to granule centroids (**Figure 35**). This implementation provides an adaptive non-maximum suppression mechanism that eliminates the rigidity of fixed-radius approaches (see IVEA-Py Evaluation section).

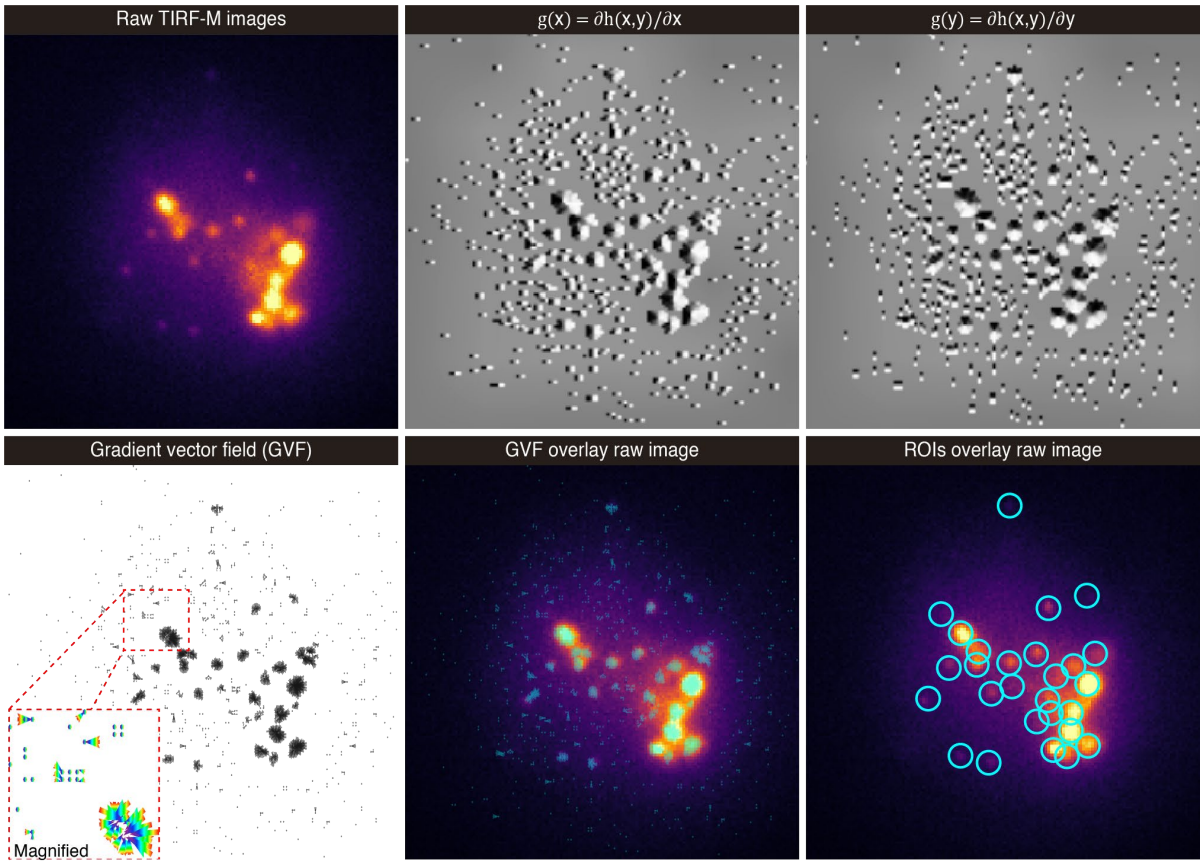


Figure 35. Image preprocessing pipeline for GVF-based convergence and centroid extraction.
The first image is the Raw TIRF-M fluorescence image of CTL cell labeled with pHuji. The next two images are the spatial gradients of the potential field (**Figure 34d**) $h(x,y)$: $g_x = \partial h/\partial x$ and $g_y = \partial h/\partial y$ (dark areas are positive, while bright ones are negative). Where, $h(x,y) =$ mask ·"LoG or DoG" (MAD-thresholded, noise-suppressed filter response); with EDT enabled, $h(x,y) = \mathrm{EDT}(\mathrm{mask})$. The fourth image is the gradient vector field visualization (GVF) used for the flow integration, alongside which is the magnified colored region for better demonstration of the pointing vectors. The fifth is the GVF overlaid on the raw image, illustrating converging flows toward granule centers. The final image is the ROI centroids (DBSCAN-clustered endpoints) overlaid on the raw image.

The LoG and DoG usually produce similar results but differ in efficiency at small spatial scales (**Figure 8**). Testing LoG filtering for small vesicles provided finer control, yielding clean circular responses with shorter runtime. At the same time, the DoG offered greater adaptability to heterogeneous granules by tuning the σ_2/σ_1 ratio. Both were retained in IVEA-Py: LoG for high-precision small-granule detection, DoG for more variable morphologies. Following detection, granule centroids were refined using DBSCAN clustering (Ester et al., 1996) to merge nearby points and suppress spurious detections.

Optional modules further enhance performance in difficult datasets, including the Euclidean Distance Transform for geometric center estimation and temporal smoothing across 3–4 frames for high-frequency acquisitions (≥ 10 Hz).

Tracking of detected granules was implemented as an offline, batch process. Detected coordinates were linked across frames using a nearest-neighbor algorithm, followed by optional Kalman filtering to smooth trajectories and bridge brief detection gaps (default = 10 frames). Linear interpolation ensured trajectory continuity while preventing erroneous merging of distinct vesicles. When temporal pooling was active, interpolation was extended to maintain alignment with pooled windows. The combined use of nearest-neighbor association, Kalman filtering, and interpolation produced smooth and biologically consistent trajectories (**Figure 38**). In addition, the Euclidean Distance Transform (Strutz, 2021) can be used to approximate vesicle centers based on circular geometry. This provides stable centroids when the signal-to-noise ratio is very low and fluctuates between frames, but it would eliminate fine detailed granule motion.

To address some cases where the cell membrane is clearly visible and risk being detected as multiple vesicles, or where regional non-uniformity in image intensities produces artifacts, an optional k-means clustering step is implemented. This procedure groups pixels by their intensity values and generates multiple thresholds and masks per cluster, improving detection in difficult conditions. However, because this method requires computing k thresholds and masks over each frame, it is slow and therefore disabled by default. This option is available only when necessary for reducing false positive detections in complex environments.

A further enhancement involved the utilization of temporal smoothing over a defined time window at the detection level. This step does not alter the original data but accelerates detection by reducing noise across frames. By averaging over a small temporal window, typically three to four frames, random noise is suppressed while true signals are reinforced. This temporal pooling strategy is particularly effective at high acquisition rates (≥10 Hz), where biological processes appear slower relative to frame capture, allowing additional information to be recovered without loss of temporal resolution.

3.8.1 IVEA-Py evaluation

The comparative evaluation of detection performance is illustrated in **Figure 36**. As shown, ComDet, ExoJ, and the GVF-based algorithms all perform well under moderate imaging conditions, yet important differences emerge when granules are closely spaced or exhibit heterogeneous intensity. In panel (a), ROI colors indicate detection outcomes: cyan marks true vesicles, yellow highlights uncertain or weak signals that may correspond to real vesicles, red denotes false detections, and green shows vesicles missed by a given algorithm but confirmed by other methods.

ComDet demonstrates robust performance on bright, isolated vesicles but frequently misclassifies faint objects due to its reliance on global thresholding. Lowering the global threshold (σ) can recover weak vesicles, but at the cost of introducing additional false positives (**Figure 36**, top and middle rows). Furthermore, morphological reconstruction steps in ComDet occasionally merge closely spaced vesicles, reducing resolution in dense regions (**Figure 36**).

ExoJ benefits from its multiscale wavelet framework, which enhances robustness against noise and reliably isolates vesicle-like structures. However, the global MAD threshold remains limiting, often discarding low-intensity granules that remain visible to the eye (**Figure 36**, third row). Both ComDet and ExoJ are effective tools and perform well under many conditions, but in challenging low-SNR datasets, their performance was surpassed by IVEA-Py.

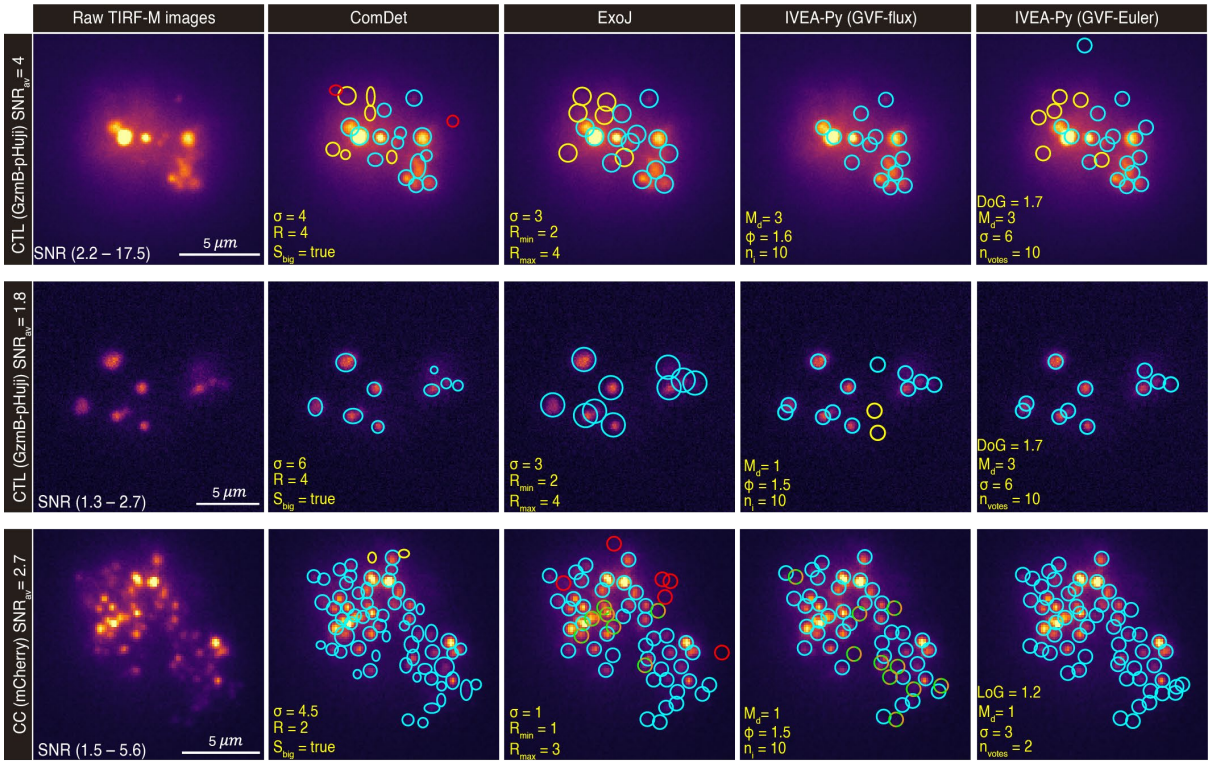


Figure 36. Comparative evaluation of granule detection and tracking in IVEA-Py and existing algorithms. Representative examples of granule detection in raw TIRF-M recordings compared across ComDet, ExoJ, and the two IVEA-Py implementations (GVF-Flux and GVF-Euler). The first column shows the raw input images together with the corresponding cell type, average signal-to-noise ratio (SNR), and full SNR range (min–max). Columns two and three display results obtained with the ComDet ImageJ plugin and the ExoJ plugin, respectively. The fourth and fifth columns show IVEA-Py detection using the gradient vector flux (GVF) method and the enhanced GVF-Euler integration approach. The Euler integration method is the default in IVEA-Py, but both options are available to the user. Detection performance is illustrated for three datasets of varying SNR, highlighting the relative strengths and weaknesses of each algorithm. ROI colors indicate the outcome of detection: cyan marks true vesicles, yellow indicates weak signals that may correspond to vesicles but are uncertain, red denotes false detections, and orange-green highlights vesicles that were missed by a given algorithm but are true events identified by other methods.

The hybrid approach in IVEA-Py, particularly with GVF-Euler integration, proved superior in detecting weak signals and separating overlapping vesicles around ~220 nm apart (**Figure 37a, b**). The separation of granules with a distance less than 220 nm presents a significant challenge, as the resulting overlap in fluorescence intensity would result in the appearance of a single larger granule. (**Figure 37b,** region A1). Unlike ComDet and ExoJ, IVEA-Py does not rely directly on global MAD thresholding to make the final detection decision. Instead, the MAD threshold is used primarily to reduce computation by suppressing background noise, while the Euler integration and voting scheme identify true convergence points. This distinction allowed IVEA-Py to maintain accuracy across datasets, as illustrated in (**Figure 36**). Notably, when analyzing small vesicles (third row), IVEA-Py-Euler clearly outperformed the other methods, which failed to consistently resolve fine details. The GVF-Flux adaptation in IVEA-Py remained stable across all three conditions, but the Euler variant achieved better resolution and sensitivity and was therefore selected as the default detection algorithm in IVEA-Py.

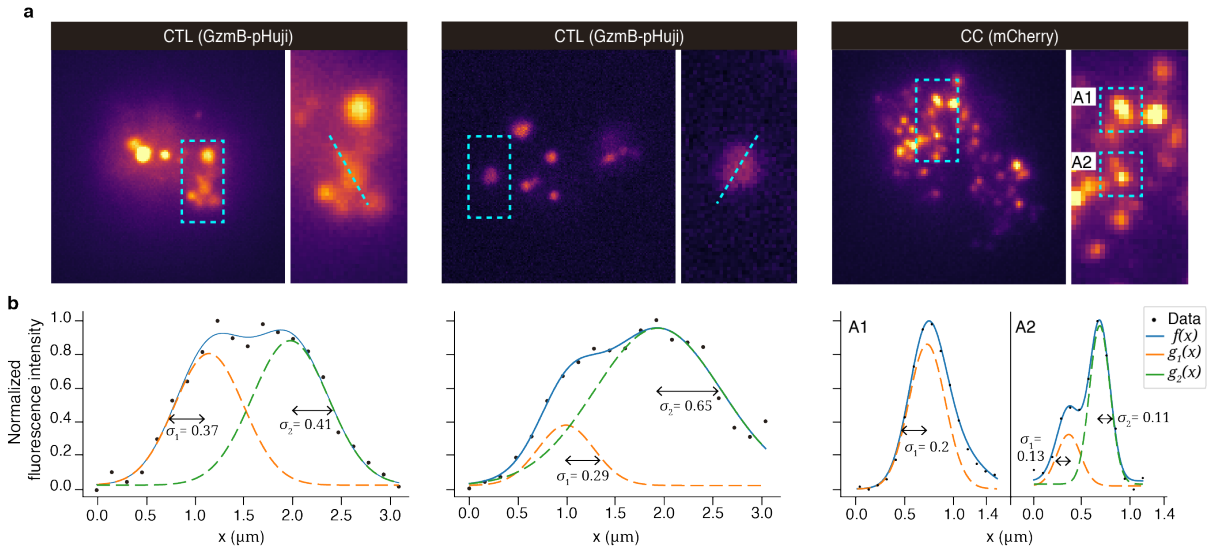


Figure 37. IVEA-Py separation of merge granules using DoG.
a. Examples of granule detection in challenging regions where vesicles are closely spaced or partially overlapping. Dashed rectangular boxes indicate areas of interest that are enlarged to the right, with additional dashed lines marking the pixel rows used for intensity profile measurements. IVEA-Py correctly separates granules in many cases where other methods fail, though instances remain where merging due to the point spread function (PSF) prevents separation. Dashed cyan segments refer to the measured intensity profile pixels. **c**. Fluorescence intensity profiles correspond to the highlighted regions in (**a**). For each case, double Gaussian fitting is shown (blue, fitted curve; green and orange, individual Gaussian components $g_1(x)$ and $g_2(x)$). The top panel shows a successful separation of two granules, while the middle panel illustrates a more difficult case in which IVEA-Py nevertheless detected and separated two partially merged vesicles. The bottom panel shows the limits of separation: in region A1, two granules separated by only one pixel (~110 nm for 3-pixel diameter vesicles) could not be resolved and were detected as a single object, consistent with the diffraction limit of ~200 nm in light microscopy. In region A2, two nearby granules were correctly resolved, demonstrating the effective range of IVEA-Py for separating closely spaced vesicles. Sigma values (σ = 0.11–0.13 µm for individual vesicles; ~0.2 µm for the merged profile) are indicated, underscoring the resolution limit of the method.

A further advantage of IVEA-Py lies in its preset parameter options, which simplify tuning for granule size. Users can select between presets optimized for small (2–3 pixels), medium (4–5 pixels), or large

(5–6 pixels) vesicles. These presets were particularly important when analyzing the smaller vesicles shown in **Figure 36** 3rd row, where the Euler method required switching to the small granule preset to maintain accuracy. For batch analysis of recordings with similar granule sizes, this preset system allows consistent parameterization without repeated manual adjustments, offering an advantage over methods that depend entirely on thresholding parameters. Advanced users can still access fine-grained parameter controls when needed, ensuring flexibility while preserving usability.

After the detection step, a tracking module is implemented in IVEA-Py to reconstruct vesicle trajectories over time. The objective of this module is not only to identify when and where granules undergo exocytosis, but also to follow their movement and behavior before performing fusion. Tracking is carried out in an "offline" fashion, which means that all detected centroids are first collected at the detection stage after DBSCAN clustering. These centroids are stored frame by frame and then processed to assign unique identifiers to vesicles across time.

The tracking pipeline is based on a combination of nearest-neighbor association and Kalman filtering. Nearest-neighbor matching was chosen as the primary association method because vesicle motion is often irregular and unpredictable, lacking a simple linear trajectory that could be modeled deterministically. Each detection in the current frame is assigned to the nearest centroid in the previous frame. This creates an initial correspondence that is efficient and reliable even for vesicles with heterogeneous dynamics. The trajectories are then smoothed with a Kalman filter, which also corrects for missing or noisy detections. In this hybrid approach, nearest-neighbor matching enables the tracing of complex or non-linear vesicle paths, while the Kalman filter applies statistical prediction to refine the trajectory and reduce jitter.

Low-confidence vesicles that appear in only a few frames are discarded to minimize false trajectories. By default, vesicles must be present in at least 10 frames (corresponding to 1 second at a 10 Hz acquisition rate) to be retained, though this threshold can be adjusted by the user. For vesicles that temporarily disappear due to noise or a weak signal, interpolation is used to fill in the missing segments of the trace. The default assumption is that if a vesicle disappears for fewer than 10 frames, its identity is maintained, and the missing positions are interpolated linearly. However, if the disappearance exceeds this threshold, a new vesicle ID is assigned. This approach preserves short-gap continuity and reduces erroneous mergers by grouping nearby detections with DBSCAN and reconnecting candidate trajectories based on their nearest centroids. It is not designed for long-term occlusions or complex trajectories. To do so, it would require more specialized tracking frameworks to recover optimal paths in those cases.

The tracking module also integrates seamlessly with temporal pooling when enabled. In this case, interpolation is performed across the pooling window, ensuring that granule positions are consistently

reconstructed even when frames have been merged for noise suppression. By combining offline centroid collection, nearest-neighbor matching, Kalman filtering, and interpolation, the tracking framework provides a reliable means to trace vesicle trajectories under a variety of imaging conditions.

The evaluation of tracking performance is summarized in (**Figure 38**). Here, trajectories from three representative recordings are displayed, with tracking limited to 500 frames for clarity. The combination of EDT, nearest-neighbor, and Kalman filter approach produced smooth and continuous trajectories, even in noisy recordings. Compared to existing plugins, which either lacked integrated tracking (ComDet) or use similar approaches and strategies but weaker detection methods (ExoJ). IVEA-Py offered a more consistent vesicle detection, which gave better tracking results due to advanced detection capabilities.

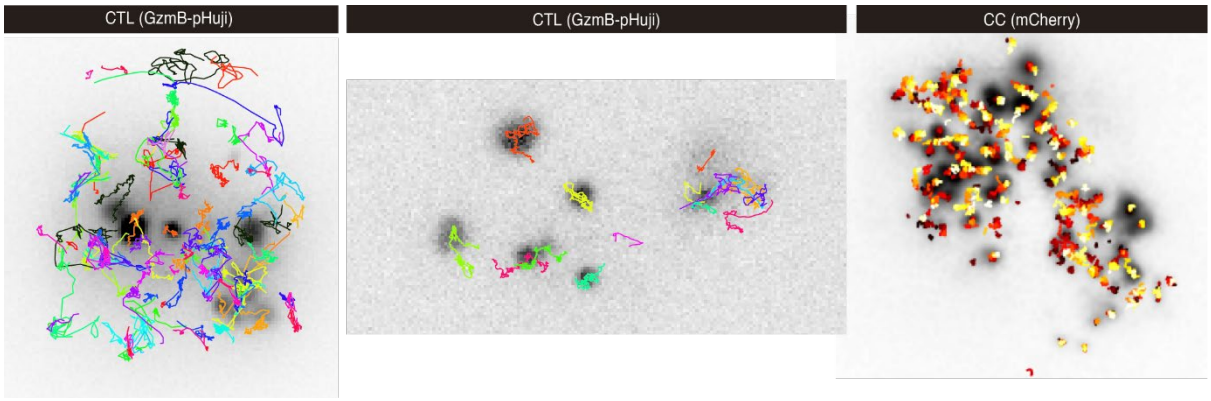


Figure 38. Granule tracking traces using IVEA-Py.
Granule tracking in IVEA-Py using a combination of nearest-neighbor association and Kalman filtering for trajectory smoothing. Traces are shown for three representative recordings, with tracking displayed over 500 frames for visual clarity. The approach yields smooth, continuous trajectories that reflect vesicle motion prior to fusion.

3.9 Hotspot area extraction.

Hotspot area detection was originally introduced in the AndromeDA project (Elizarova et al., 2022), which aimed to visualize dopamine release from axonal varicosities with high spatiotemporal resolution. AndromeDA is based on a near-infrared fluorescent nanosensor paint composed of single-walled carbon nanotubes functionalized with DNA. When dopamine binds reversibly to these nanosensors, their fluorescence emission increases, thereby enabling the direct optical detection of secretion events in cultured dopaminergic neurons. This approach represented a major advance over previous electrochemical and genetically encoded sensors, which lacked the necessary spatial resolution to resolve individual varicosities (Elizarova et al., 2022).

With AndromeDA, discrete release events could be distinguished from diffuse extracellular dopamine dispersion, revealing that hotspots occurred adjacent to a small fraction of varicosities only. Analysis demonstrated that dopamine release is highly heterogeneous: only about 17% of varicosities exhibited detectable release, while the majority remained functionally silent (Elizarova et al., 2022). Hotspot

events appeared transient, typically spanning only a few image frames. They exhibit a rapid onset followed by a slower decay, consistent with localized vesicular release and subsequent diffusion. Their occurrence was tightly coupled to neuronal activity and could be pharmacologically modulated.

To systematically identify hotspots, a machine learning-based tool called the Dopamine Recognition Tool (DART) was developed as a Fiji plugin. DART was designed to automatically detect regions of interest that show local increases in AndromeDA fluorescence and to correct for background signal fluctuations. This approach enabled high-throughput, unbiased detection of hotspots across large populations of varicosities. Manual analysis alone was insufficient, as these events are characterized by weak signals amid a highly noisy background (Elizarova et al., 2022).

When IVEA was developed to detect and analyze different types of exocytosis, the DART logic was integrated and expanded. Three significant improvements were introduced: An automatic parameter estimation module was utilized to reduce manual parameter tuning. A Multilayer Intensity Correction (MIC) method was developed to correct regional intensity fluctuations without distorting biological signals. An event temporal tracking technique is adapted to link hotspot dynamics across time. These additions transformed the hotspot module into a more robust and user-friendly system.

3.9.1 Performance and evaluation

In this module, events were detected using a combination of k-means clustering and iterative thresholding, followed by segmentation and temporal mean-intensity tracking (**Figure 39b**). Prior to detection, raw images underwent intensity fluctuation correction, a critical step since fluctuations could mimic or obscure genuine hotspots. Intensity fluctuations were not uniform across the image regions, which prevented normal methods from addressing these artifacts. The MIC algorithm (**Figure 40**) addressed this by segmenting the image into clusters (default, k = 5 after Gaussian filtering), resembling regions relative to their intensity value (Pham et al., 2000). Then, pixel intensities are corrected based on the mean intensity variation within each cluster. This localized correction effectively suppressed background fluctuations while preserving the true hotspot signals.

After correction, foreground detection was performed by calculating frame-to-frame intensity variation over a time window and binarizing the resulting difference image (**Figure 39c**). Standard Fiji thresholding algorithms were tested but produced inconsistent results due to sensitivity to noise statistics. Instead, an iterative global threshold was implemented. This threshold systematically increments the cutoff until the background noise is eliminated (**Figure 39c**). Detected hotspots were then tracked temporally by monitoring ROI intensity profiles (**Figure 39d**, **e**). Tracking ended when the ROI signal fell below half-maximal intensity, ensuring that only biologically relevant events were retained.

To evaluate the performance of IVEA in hotspot detection, we applied it to the same recordings originally analyzed with DART (**Figure 39a**) Using the default automated parameters, IVEA consistently detected the same hotspot regions as DART, typically within a margin of one ROI, but with a reduction in false positives, increasing its precision. This demonstrated that IVEA preserved the sensitivity of DART while enhancing specificity, highlighting its robustness for analyzing neuronal activity (**Figure 39f**).

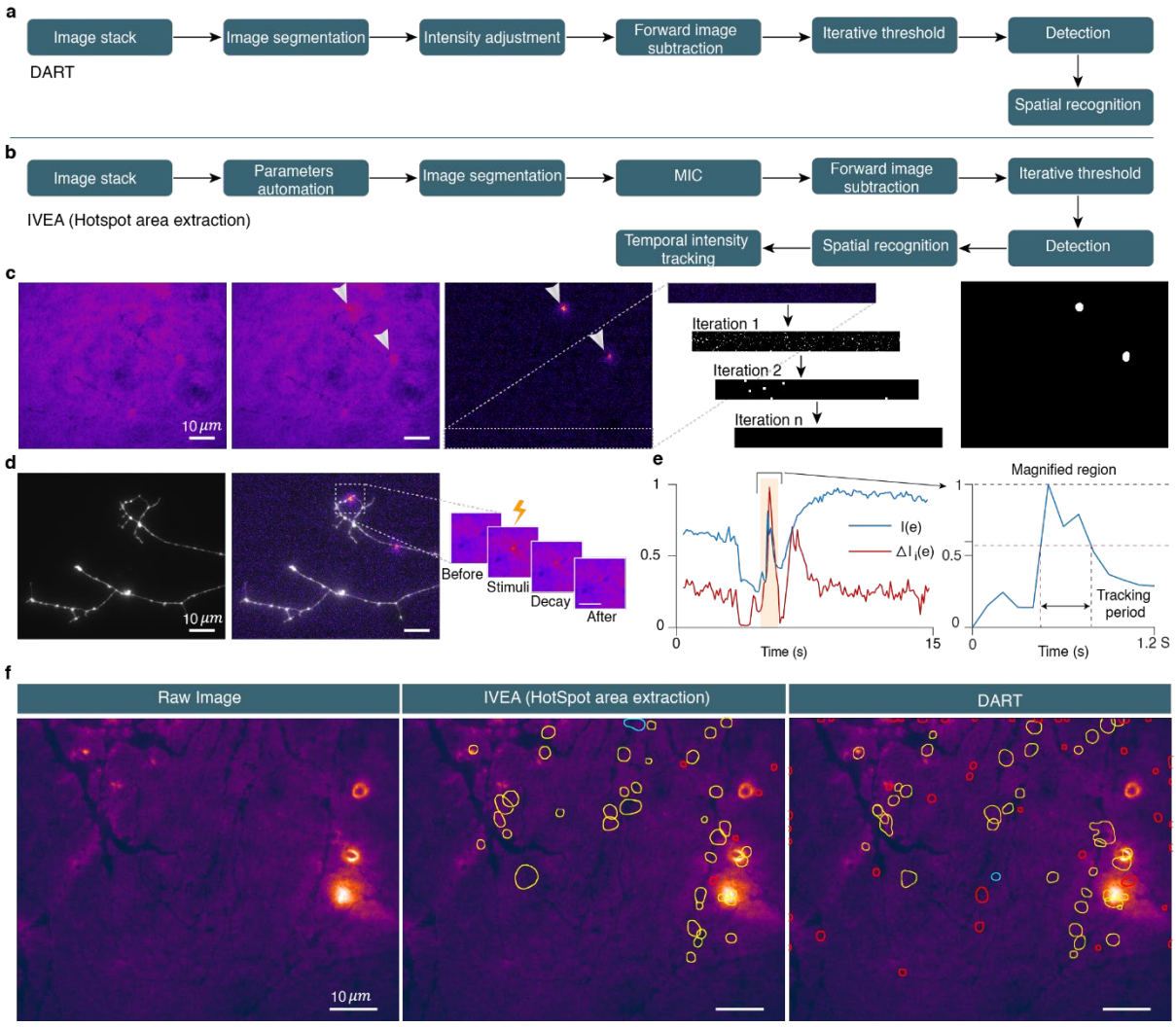


Figure 39 Performance of the sensor-based exocytosis detection algorithm (adapted from (Chouaib et al., 2025)).

a. Flowchart of the DART workflow. **b**. Flowchart of the IVEA hotspot extraction module. Compared with DART, IVEA introduces three major improvements: automated parameter estimation, multi-layer intensity correction (MIC; see Fig. 37), and temporal tracking of hotspot activity. **c**. Example of the detection process from left to right: the raw image; the same frame highlighting two candidate hotspots; the corresponding intensity-variation map, where hotspots are clearly visible; and the final segmented image used to define event ROIs. The iterative thresholding procedure operates on cropped subregions, progressively estimating the noise level and discarding spurious signals. **d**. Temporal hotspot tracking illustrated from left to right: raw neuron image, intensity-variation image with raw overlay, and a sequence of snapshots showing the tracked hotspot region over time. **e**. Representative intensity traces for a hotspot ROI. The mean signal extracted from the raw image sequence $I(e)$ is compared to the processed intensity-variation sequence $\Delta I_i(e)$. The inset magnifies the time window of the hotspot, showing the tracking period. **f**. Comparison of hotspot area detection by IVEA and DART. Yellow ROIs denote hotspots identified by both methods, red ROIs indicate likely false detections, and cyan ROIs mark true hotspots detected by one algorithm but missed by the other.

To examine the contribution of the MIC algorithm, it was directly compared to the correction strategies previously employed (**Figure 40**). Applying a simple ratio correction to the entire image proved inadequate, leaving strong residual background fluctuations that often produced spurious regions. The DART correction improved upon this by incorporating mean intensity variation between consecutive frames within pixel clusters, which partially suppressed noise but still attenuated or distorted genuine signals. This previous method partially suppresses noise, yet alters or attenuates some genuine signals. In contrast, MIC extended the ratio-based approach by utilizing it over each cluster (image layer), enabling layer-based correction. This refinement resulted in a clear separation of true activity from background, effectively eliminating fluctuations while preserving the temporal dynamics of active regions. The improvement was evident both in the fluorescence intensity traces and in the segmented hotspot regions (**Figure 40a–d**). These results demonstrate how IVEA, by combining MIC with automated parameter estimation and temporal tracking, enhanced the earlier DART framework. The IVEA hotspot area extraction module reduces user intervention, improves reproducibility, and provides more accurate detection of biologically meaningful hotspots.

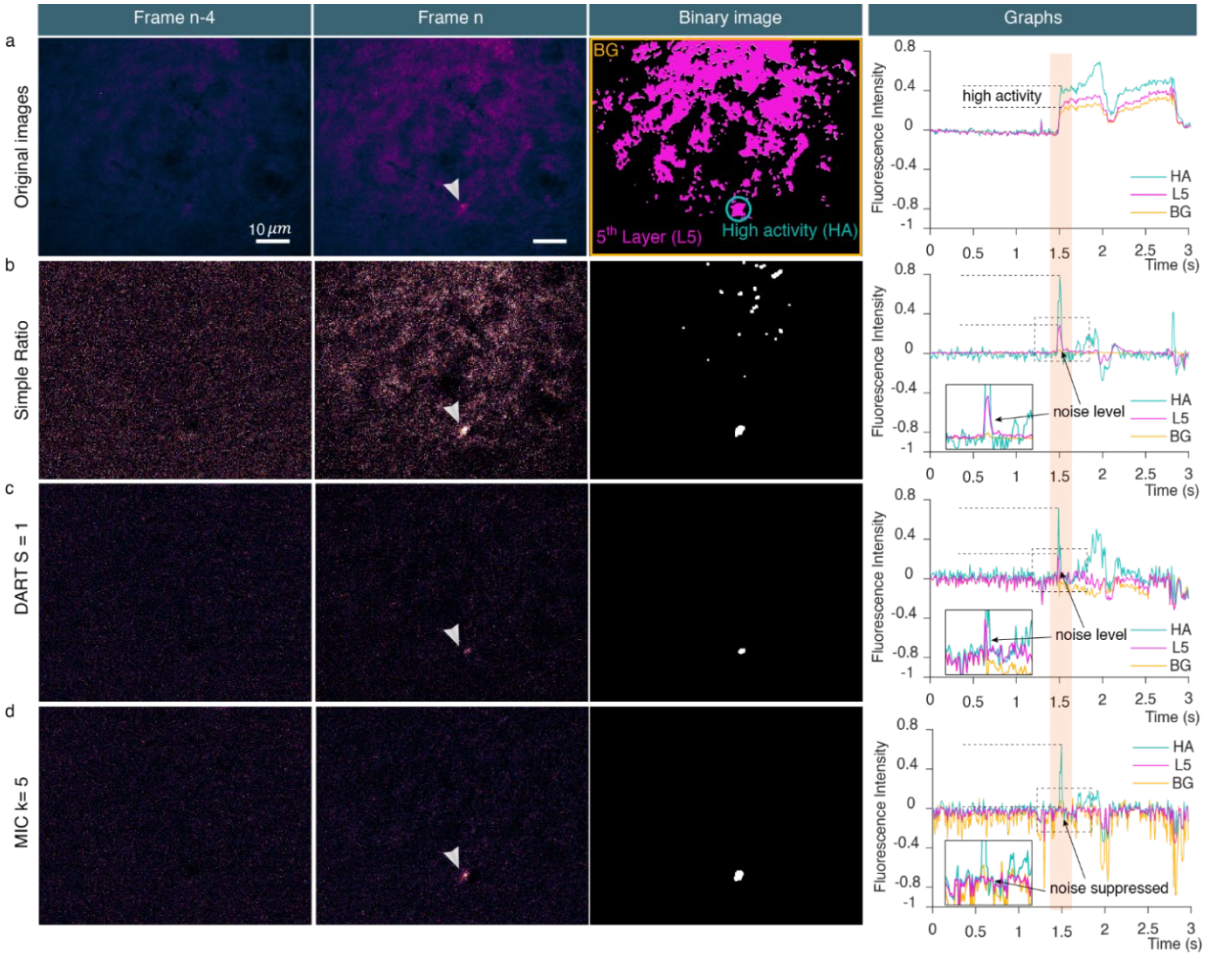


Figure 40. Multi-layer intensity correction (MIC (adapted from (Chouaib et al., 2025))).
The figure demonstrates how MIC addresses spatially non-uniform background fluctuations in fluorescence imaging and compares it with alternative approaches. The first column shows reference images without fluctuations. The second column shows an image containing both a high-activity (HA) region and background with

intensity fluctuations. The third column shows binary images derived from the second column by applying triangular thresholding and morphological reconstruction (erosion, median filtering, and dilation). The final column presents fluorescence intensity traces across three regions defined in the first binary image: the fifth k-means cluster layer (L5, magenta), the HA region (cyan), and the background (BG, orange). Insets magnify the period of hotspot activity. a, Raw images clustered with k-means (k=5), where L5 is taken as reference; the plot illustrates the elevated activity of the HA relative to L5 and BG. b, Results after applying a simple ratio correction, which fails to remove spurious regions; the corresponding trace shows residual noise contaminating the HA measurement. c, Results of the DART approach, which partially corrects non-uniform fluctuations but does not fully suppress noise. d, MIC with k=5 used to split the image into five layers to eliminate background noise while preserving the HA signal, enabling accurate temporal tracking.

4 Discussion & outlook

4.1 Introduction to the discussion

The study of exocytosis not only requires the visualization of vesicular fusion events but also a reliable framework for their systematic detection, quantification, and classification. Traditional approaches in this field have relied heavily on manual annotation or on rule-based image processing algorithms. These methods have proven beneficial in specific scenarios, but they are limited in objectivity, scalability, and reproducibility by nature. As imaging technologies such as TIRF-M, confocal, and super-resolution microscopy continue to advance, the datasets generated have grown in both size and complexity. Thus, there has been a pressing need for analytical tools that can handle these large-scale datasets with accuracy and efficiency, while reducing the dependence on human input and biases.

The IVEA platform was developed to tackle these challenges and push the exocytosis field forward. The design of IVEA was driven by a key realization that vesicle fusion events not only have diverse characteristics, but also that they take place within a complex background of heterogeneous cellular signals, motion artifacts, and fluctuating noise levels. Due to these complications, simple thresholding techniques are insufficient because they either misclassify background fluctuations and produce false positives or fail to capture weak biologically relevant events. By combining machine learning techniques with sophisticated image analysis, IVEA overcomes these constraints. This enables the platform to detect, classify, and interpret vesicle fusion events in a robust and scalable way.

The main goal behind the development of the IVEA platform was to create an automated, data-driven framework capable of identifying and classifying exocytosis events according to their biological relevance. The software was developed through a stepwise, problem-driven process, where each new version was designed to address limitations identified in previous iterations or to accommodate new analytical capabilities required by experimental data. Early implementations focused on temporal-based classification of candidate regions using recurrent neural networks, whereas later versions incorporated transformer-based architectures to jointly capture spatial and temporal features of vesicle fusion. Through this continuous refinement, IVEA evolved from a conventional image-processing tool into a comprehensive analytical platform capable of learning from complex imaging data and adapting to a wide range of experimental conditions.

For the Python implementation, IVEA was extended beyond event classification to incorporate granule detection and tracking (IVEA-Py). The addition of detection and tracking capabilities gave IVEA-Py the ability to provide dynamic information about vesicle behavior by tracing the path of vesicles as they move, dock, and prime prior to release. The IVEA-Py classification and tracking were evaluated on different datasets and compared with other existing software and methods. The following section

discusses these results and highlights the practical implications of IVEA-Py's granule detection and tracking framework.

4.1 Methodological considerations

The segmentation in the IVEA hotspot area detection module was fundamentally different from conventional object-based workflows. As shown in the Results section (**Figure 39–40)**, the structures of interest lack stable spatial features; instead, they exhibit brief fluorescence changes distributed across space and time. These signals lack consistent morphology and often display low contrast. This renders methods that rely on spatial features, particularly deep learning models developed for semantic or instance segmentation, not well-suited to our AndromeDA data (Elizarova et al., 2022). Even spatiotemporal neural network architectures would require reproducible event signatures, which our recordings do not reliably provide. Another practical limitation was the absence of training data. As we noted in the Results section, dopamine release events in AndromeDA are extremely weak and often difficult to distinguish from the background by eye (Elizarova et al., 2022). Under these circumstances, constructing meaningful ground-truth labels or assembling a representative dataset for supervised learning was not feasible either. Without a reliable definition of what constitutes an actual event, a learning-based approach would likely converge on biased patterns rather than biological signals. For these reasons, a classical image-processing strategy was the most appropriate and transparent choice in our case. The iterative global thresholding method described earlier in the Methods and Results sections was selected because standard Fiji thresholds assume the presence of a clear foreground distribution. AndromeDA recordings frequently consisted of a uniform background signal without an event, for which later threshold assumptions fail. Unlike these methods, the iterative threshold implemented in our case treats each frame primarily as a noise-estimation problem. It increases the cutoff until background fluctuations are removed, ensuring that a threshold is produced even when no event is present. This made the approach more stable and reliable under the low-contrast and highly variable conditions of our data.

A major challenge, however, was the strong and spatially heterogeneous intensity fluctuations across the field of view. As demonstrated in the Results section, global normalization approaches suppressed some fluctuations but often attenuated or distorted genuine signals. Several histogram-based corrections, including equalization and gamma adjustment, were also tested, but these methods usually suppressed weak events or altered their temporal shape. This motivated the introduction of a cluster-based correction strategy, MIC, which has demonstrated effectiveness in stabilizing intensity variation within local intensity groups (**Figure 40**). Notably, the MIC approach is tailored to datasets in which fluorescence is distributed across broad intensity ranges and where fluctuations disproportionately affect brighter pixels (**Figure 40a,** 3rd column). While this strategy proved effective for the AndromeDA recordings, it may need to be adjusted and adapted for other imaging contexts. In rare cases, cluster

assignments may still suppress true events if they fall within a region exclusively associated with actual events.

Considering developing a new non-maximum suppression algorithm in IVEA was based on different factors. Clustering and radius-based suppression algorithms can effectively handle static point distributions, but they are less suitable for dynamic fluorescence events that evolve continuously in time. Methods such as DBSCAN or Mean Shift operate on discrete coordinate sets and depend on user-defined neighborhood radii or bandwidths. In contrast, GNMS formulates suppression analytically within a continuous Gaussian field, integrating both spatial and temporal proximity through a single mathematical framework.

This distinction is important: exocytotic events do not manifest as discrete, stationary points but as temporally localized intensity diffusions. A radius-based or clustering approach could merge temporally distinct but spatially close events, leading to the loss of biologically meaningful information. The spatiotemporal Gaussian weighting used in GNMS resolves this issue by evaluating overlap in continuous (x, y, t) space, ensuring that detections remain independent unless they are truly concurrent in both space and time. The development of such an algorithm is needed to generalize the concept of non-maximum suppression beyond static geometry and provide a physically interpretable means of resolving redundancy in time-resolved microscopy data.

Regarding the random and stationary burst events modules, a methodological consideration for IVEA entailed the selection of a neural network architecture for classification that involved a range of tradeoffs. Convolutional neural networks, such as ResNet (He et al., 2016), U-Net (Ronneberger et al., 2015) and others have been widely applied to biomedical image analysis. However, IVEA does not use these architectures. To manage computational limits, a custom lightweight CNN encoder was created to precede the transformer stage of eViT. The goal was to have an encoder capable of efficient, compact spatial feature extraction without the computational overhead of a deeper CNN model. By embedding these features into a transformer framework, IVEA retained the ability to capture long-range dependencies across both space and time while remaining computationally tractable. This hybrid architecture was established to serve as an intermediary between traditional CNN-based models and a fully transformer-based model, capturing efficient accuracy.

Considering the vision radius for the eViT classifier for better classification. Expanding the vision radius enables the model to take in more contextual information surrounding a candidate event, which may improve classification by capturing cell morphology, neighboring vesicles, and background structures in the image. However, larger vision radii come with the risk of potentially introducing confounding signals, especially in dense areas, and the model may distance its attention from the vesicle of interest. Ultimately, it is possible to tune the radius to optimize the balance between specificity and

sensitivity; smaller radii favor isolated vesicles, and larger radii increase robustness in predicting more heterogeneous areas.

Beyond random and stationary burst modules, the next analytical step in IVEA involves a segmentation algorithm utilized in the hotspot area extraction module. The choice of classical segmentation algorithms in IVEA was guided by the physical characteristics of the data rather than by model complexity. Deep-learning segmentation frameworks (semantic, instance, panoptic) require well-defined object morphology and dense ground-truth labels. These conditions are difficult to apply to transient fluorescence bursts, which vary widely in intensity, duration, and spatial extent. In contrast, simpler intensity-based algorithms (iterative thresholding, DoG, k-means) are agnostic to object shape, robust to noise, and preserve temporal precision, which is essential when events last only a few frames. These methods also minimize computational overhead, enabling on-the-fly analysis of large microscopy datasets without GPU acceleration.

Finally, platform selection had a critical role in the compositional elements of design in IVEA. Initial development was performed in MATLAB ("MATLAB," 2023), which allowed quick prototype development of algorithms and visualization of intermediate results. The first implemented version to be publicly propagated was developed as a Fiji plugin, with Java as the programming language. This ensured the IVEA implementation would have a vast audience in the imaging community, but it also limited the ease of integrating new machine learning libraries and limited flexibility for development. The transition to using Python for IVEA-Py addressed these limitations. This implementation supported the inclusion of vectorized computation for granule detection and allowed deep learning frameworks like Google TensorFlow (Abadi et al., 2023; "TensorFlow for Java," 2019) to be integrated. Nevertheless, implementing IVEA as a Fiji/ImageJ plugin written in Java was a design choice aimed at ensuring accessibility for researchers familiar with the ImageJ software. This decision prioritized user accessibility, as Fiji/ImageJ plugins written in Java integrate directly into existing workflows with minimal setup. However, it also increased development complexity, since implementing and maintaining such functionality in Java is more demanding than in Python. Development and community support are broader and more flexible in Python, but the resulting tools are generally less accessible to non-technical users.

4.2 IVEA comparison with existing tools (exocytosis analysis)

To highlight IVEA's contribution, it is essential to place it in the context of previous analytical tools developed for studies of exocytosis. Over the last decade, various Fiji ImageJv2-based plugins (Rueden et al., 2017), and standalone rule-based pipelines were developed. ExoJ (Liu et al., 2024), pHusion (O'Shaughnessy et al., 2024), and SynActJ (Schmied et al., 2021) each introduced different advances and/or addressed different challenges. These tools have yielded pertinent and iterative contributions to the study of vesicle dynamics, facilitating engagement and adoption by interested and capable members

of the imaging community. However, limitations described below emphasize the necessity for more adaptable, scalable, and learning-based methodologies, such as IVEA (Chouaib et al., 2025).

Starting with ExoJ, this software implemented a comparatively sophisticated approach for vesicle detection and exocytosis analysis, combining temporal linking and Gaussian fitting to approximate event dynamics. While such methods represented a step toward a more integrated detection framework, ExoJ remained largely restricted to identifying a single class of events, specifically, the latent vesicle fusion. In contrast, IVEA demonstrated reliable performance across a broad range of exocytosis modalities, maintaining accuracy under diverse labeling and imaging conditions. These results highlight that even advanced rule-based systems, despite their algorithmic refinements, lack the adaptability and contextual learning capacity exhibited by deep learning-based models, which can generalize effectively across variable biological and optical conditions.

pHusion represented another rule-based method aimed at characterizing vesicle fusion events by prompting the user to define a series of scripts to be executed. These scripts run within a specific platform called ImageTank, each with user-adjustable parameters. It was difficult to adapt to working with pHusion, as these scripts execute sequentially. While the approach offers flexibility, it is inherently fragile: errors in one step can halt the entire process. Installation was non-trivial as well, and debugging failures required both coding literacy and logical reasoning, even though the interface does not explicitly require programming. pHusion could be useful for certain datasets, but was limited in its capabilities, as it was also bound to a specific operating system, macOS. In my opinion, pHusion is a nice project that has potential, as it has its own environment that can be later expanded to become something similar to the ImageJ software. However, at its current stage, the software remains difficult for the average user to operate effectively.

Finally, SynActJ ImageJ plugin compared with IVAE's stationary burst events module. This software is a Fiji-based plugin for automated analysis of synaptic activity that applies classical image processing (LoG filtering, watershed segmentation, and thresholding) in combination with an R Shiny app for quantitative analysis of fluorescence traces. SynActJ provides an analysis workflow for the study of synaptic boutons, but it still relies on user-defined global thresholds, which can be different across datasets. However, when tested on our datasets, SynActJ failed to detect relevant exocytosis events and therefore could not be applied effectively for our analyses (**Figure 32**).

These tools collectively demonstrated impressive ingenuity but shared common limitations: reliance on global thresholds, fixed morphological rules, and rigid processing pipelines. Such designs work under controlled conditions but fail in heterogeneous datasets. IVEA framework merges the accessibility of Fiji-based tools with the flexibility of modern AI methods (**Table 10**).

4.3 Evolution of IVEA's event detection and classification

Initially, the focus was on detecting exocytosis events focused on intensity changes, using analysis and rule-based detection approaches. The early methods relied on detecting sharp deviations in fluorescence intensity within a region of interest and producing their segmentation and classification based on the temporal characteristics of the signal. Although these methods were successfully able to identify obvious, high-intensity fusion events, a more nuanced approach was needed for less obvious fusion events, especially in noisy or heterogeneous imaging conditions. Additionally, rule-based approaches are challenging to incorporate new datasets and require significant modifications to their prior approach for changes in conditions.

To address these issues, machine learning was introduced into IVEA, which is a major shift from the past reliance on pure-reliability methods. The first significant application using machine learning was the classification of events using the LSTM network. The advantages of LSTM included the ability to capture time-dependent sequences in fluorescent intensity traces (Karim et al., 2019). In our case, it was able to learn the structure of a vesicle fusion event related to exocytosis against other fluctuations, each of which is regarded as stochastic in nature. This was a major improvement over static classifiers as it included the spatiotemporal dynamics of vesicle release into the analytical approach. However, there were also limitations to LSTM. It was very sensitive to local intensity noise, and many times misclassified complex events when local spatial context (e.g., overlapping vesicles or cell morphology) was significant.

Building on this foundation, our classification module was later improved by employing an enhanced version of the vision transformer network (Dosovitskiy et al., 2021). There were two main reasons for adopting a transformer-based approach. First, we know that transformers are adept at spanning both the local and global relationships present in structured data (Vaswani et al., 2017) (or in our case, both local and complex relationships in spaces that incorporate the temporal dimension alongside the spatial). This allows the eViT model to effectively capture the temporal information incorporated with spatial features from the surrounding cellular context. Second, this design approach simplifies the way human experts process their data by categorizing the spatiotemporal appearance of the event as it exists in a raw image during visualization. The eViT extracts richer representations of vesicle fusion events than what is possible from an LSTM alone. This is achieved by embedding fluorescent traces into spatiotemporal patches all at once, which improves accuracy and generalizes better across various cell types and multiple imaging conditions.

The shift from LSTM to eViT improved classification accuracy, highlighting the need to combine both spatial and temporal features for better classification (**Figure 25**). The eViT model, for example, made it easier for the platform to analyze more complex events, including those occurring near bright structures or in varied backgrounds where LSTM struggled to a greater extent (**Figure 41**). However,

utilizing eViT introduced other practical issues, mainly related to computational challenges. The transformer-based framework required much more GPU resources, and model performance depended on the video memory available and patch size options. This meant that not all aspects of eViT could be further explored due to hardware limitations, and it will likely be a field to pursue as computational resources improve. To mitigate these hardware constraints, local optimization was implemented at the inference stage. Instead of loading the entire dataset into memory, the data were processed in sequential chunks, reducing both RAM and CPU overhead. This approach improved system stability and ensured compatibility across standard x86 architectures, allowing inference to be executed efficiently even on machines with limited memory. However, this modification did not increase computational speed, as it primarily addressed memory management rather than processing throughput.

Overall, IVEA's classification framework illustrates a gradual transition from rule-based intensity analysis approaches to data-driven spatiotemporal modeling. The past methods of rule-driven segmentation tools, and the works of Li et al, with their Hierarchical CNN (H. Li et al., 2017) were a demonstration of both the promise and challenges of past methods. Despite the improvements made to classification rates by the HCNN, similar rule-driven segmentation methods proceeded with considerable pre-processing operations, including Gaussian Mixture Model fitting, for engineered "GMM images". Although these pipelines can be very effective with carefully curated datasets, they are always fragile to vesicle motion, noise, intensity fluctuations, and out-of-focus artifacts, and do not lend themselves to the diverse and complicated nature of real experimental datasets. IVEA software was intended to provide a purposeful alternative to those pipelines. Rather than compressing temporal sequences into deep/engineered representations and separately examining candidate events. In this way, we allow IVEA to directly ingest the raw fluorescence intensity variations of the images. IVEA learns both spatial and temporal dependencies end-to-end with the use of a Vision Transformer. Therefore, it does not require any manual feature design or exhaustively engineered methods, allowing IVEA to potentially robustly generalize to varied datasets and perform well in batch analysis.

4.4 Granule detection and tracking module (IVEA-Py)

Exocytosis is associated with proximal trafficking, docking, and priming steps; therefore, capturing some of these upstream dynamics is also important. To utilize this functionality, the Python implementation IVEA-Py included a module for granule detection and tracking. This module is not revolutionary, but it certainly presents complementary information by connecting events back to vesicle trajectories, thereby moving away from a purely event-based paradigm to incorporate vesicle movement as well.

Adopting granule detection required careful search of available algorithms. Multiple methodologies were explored, ranging from simple intensity thresholding and morphology-based methods to more complex plugin architectures. Ultimately, three approaches stood out as potential candidates: the

ComDet ImageJ plugin, the ExoJ ImageJ plugin, and the gradient vector field flux method (G. Li et al., 2007). Each included merits and drawbacks that otherwise precluded their implementation alone into the IVEA-Py platform. ComDet was computationally straightforward and easy to use, but it required the use of global histogram thresholding and morphological reconstruction techniques, which could prove difficult for distinguishing less intense vesicular features or separating multiple granules in close proximity. ExoJ provided the à trous wavelet transform (Olivo-Marin, 2002) combined with a median absolute deviation threshold that enhanced the sensitivity of detecting vesicle-like features across scales (Liu et al., 2024). However, its reliance on a global MAD affected the ability to recover less intense vesicles in a heterogeneous background. The granule separation likely arises from the smoothing nature of the à trous transform implemented with the B3-spline kernel (Olivo-Marin, 2002). This can blur adjacent intensity peaks and hinder the separation of overlapping vesicles (**Figure 36**, 3rd row). On the other hand, the GVF flux method depended on image gradients and elastic diffusion to find blob centers, thus demonstrating robustness even for crowded or noisy regions. Its reliance on flux thresholding and radius-based suppression limited the flexibility of data sets with vesicles of varying sizes (Olivo-Marin, 2002).

Based on these observations, IVEA-Py implemented a hybrid framework that utilizes features of each of the three approaches, along with an added technique. The initial stage of detection involves the implementation of differences of Gaussian preprocessing or the Laplacian of Gaussian algorithms, enhancing a vesicle-like structure while suppressing background variation. The DoG/LoG images are then filtered by low MAD thresholding, eliminating spurious detections, after which gradient vector diffusion is applied. The implementation of a low MAD threshold in this context is not intended to identify regions of interest. Instead, its primary function is to reduce noise in images and simplify computation for determining GVF convergence points. Rather than using the original GVF flux approach (G. Li et al., 2007), which utilizes flux thresholding, a different methodology is proposed. The Euler integration and iterative voting were utilized to find local convergence points across candidate regions, which was helpful to capture true vesicles even when the maximum threshold sensitivity is low. As illustrated in **Figure 35 (**4th image), the use of a low threshold initially yields numerous candidate regions. Afterwards, Euler-based integration combined with a trajectory voting scheme refines these candidates (the voting threshold was set to 10 for this example). Even with the low thresholding, the resulting detections in the final ROIs were accurate and precisely overlapped the vesicle structures in the raw fluorescence image (**Figure 35,** 6th image).

Using both the Laplacian of Gaussian and Difference of Gaussian filters produces similar outputs, but their practical behavior differs at small spatial scales. For small granules, typically only a few pixels in diameter, the discrete sampling of the image makes the exact kernel size and σ value critical. In this regime, the LoG offers more precise control since its parameters can be directly tuned (e.g., a 5×5 or

7×7 kernel with $\sigma \approx 1$) to match the expected granule dimensions, producing clean, circular responses with minimal effort. Despite its theoretical approximation, the DoG does not necessarily offer computational benefits in practice; benchmarking 200 frames (200×200 pixels) showed an average processing time of approximately 1.9081 sec for LoG (7×7 kernel) and 2.1173 sec for DoG, likely due to the two sequential Gaussian convolutions required.

Experimentally, the LoG performed better for small, near-Gaussian granules, while the DoG proved more flexible for larger or heterogeneous structures, where tuning the σ_2/σ_1 ratio allows smoother adaptation across scales. For the matter of choice, both filters were retained in IVEA: the LoG is recommended for fine-scale detection of small vesicles, whereas the DoG remains a versatile alternative for broader or irregular morphologies.

Once the candidate centroids have been identified, density-based spatial clustering of applications with noise is implemented to further refine the centroids (Ester et al., 1996). This clustering step was a better idea than using a fixed radius, as it is more dynamic and merges clusters of points into one. There are also some optional modules that enhance detection in difficult scenarios.

The Euclidean Distance Transform step was introduced to refine centroid estimation. When applied to the masked image $h(x, y)$, the EDT leverages the approximately circular geometry of granules to estimate their centers more consistently (Strutz, 2021). Although this method provides an approximation rather than the exact convergence point, it offers the practical advantage of suppressing the apparent centroid oscillation observed across frames. This "wobbling" of vesicle centers can arise from a combination of biological and physical factors, including bidirectional transport along microtubules (Qu et al., 2025) driven by opposing motor proteins (kinesin and dynein) (Zhang et al., 2021), stochastic Brownian motion, and minor optical or mechanical fluctuations during imaging. For users primarily interested in stable trajectory tracking rather than fine-scale displacement analysis, enabling the EDT step is recommended to obtain smoother and more robust centroid trajectories. A final option is to use temporal smoothing across three or four frames to help suppress random noise and reinforce true signals when detecting. This is useful for higher frequency acquisitions ($\geq$ 10 Hz), which are typically noisy, since biological dynamics are slower than the frame rate.

After detection, I thought of implementing an offline tracking approach to reconstruct the vesicle trajectories. In our case, it is a batch analysis, which means live tracking is not crucial, and in the same way, I can speed up the process by using parallel looping or vectorization over all the detected and stored coordinates. These coordinates were stored frame by frame, then IVEA-Py assigned identities to vesicles using the nearest neighbor matching algorithm. This approach was preferred because vesicle paths are unpredictable and often non-linear, making deterministic motion models unsuitable. The

nearest neighbor assignments were subsequently refined with a Kalman filter, which smoothed the trajectories and corrected for short periods without detection. However, this step is optional at the user's discretion, as Kalman filtering would smooth trajectories by predicting motion across uncertain frames. In cases where precise localization of rapid movements is critical, users may choose to disable this step to avoid potential loss of fine motion details. Any low-confidence vesicles appearing in only a few frames were discarded, and short periods of disappearance (default is 10 frames) between detection intervals were bridged using linear interpolation. This ensured continuity in trajectories without erroneously merging distinct vesicles. In the case of temporal pooling being enabled, interpolations are extended throughout the pooling window to retain the granule locations. The combination of nearest neighbor matching, Kalman filtering, and interpolation produced smooth trajectories (**Figure 38**).

A key advantage of IVEA-Py compared to other methods is the balance between flexibility and usability. While the addition of other parameters (e.g., ratios of DoG, granule size, etc.) made using the feature more complicated, the decision was made to add preset options to make it more user-friendly. For users who are more advanced, manual parameter adjustments remain available to obtain more precise control over the parameters used for detection and tracking. This dual-layer design ensures that IVEA-Py can be used as a user-friendly tool and as a highly customizable platform for specialized datasets.

In summary, the granule detection and tracking module in IVEA-Py extends the platform from event classification to detailed studies on vesicle dynamics in terms of detection, tracking, and analysis. The IVEA-Py module has been developed to address the limitations of existing technologies. This is accomplished by integrating and enhancing multiple detection methodologies, utilizing optional techniques for challenging datasets, and tracking methodology using nearest neighbor with Kalman filtering. This module is designed to detect, identify, and track vesicles across a range of imaging conditions, thus offering a significant advancement in the field. Besides the improvement of measurement in studying exocytosis, this extension of IVEA-Py provides a needed opportunity to expand studies on vesicle trafficking to membrane dynamics.

4.5 Scope and limitations

While IVEA provides considerable improvement over existing methods for detecting, classifying, and tracking exocytosis events, it is vital to understand its scope and limitations. Like all analytical platforms, IVEA was built to answer a specific subset of biological questions and imaging conditions, and its performance must be interpreted in that context.

One of IVEA's strengths is its ability to generalize across various cell types and experimental systems. During development, the platform has been successful with neuronal, immune, and endocrine cells, each presenting unique imaging challenges due to vesicle size, density, and background heterogeneity.

The combination of rule-based preprocessing and machine-learning-based classification has enabled IVEA to remain robust across these contexts. However, IVEA is optimized for burst-like fusion events, in which vesicle release is observed as a discrete change in intensity over a short time frame. Long-lasting non-burst events, such as those attempting to be quantified with specific probes (e.g., Synaptobrevin-SEP) (Chi et al., 2001; Fernandez-Alfonso et al., 2006), remain more difficult to measure reliably. These events often lack the sharp temporal dynamics that IVEA's classifiers were trained to recognize, underscoring the need for future adaptations to broaden the platform's event repertoire.

Another limitation to be discussed includes vesicle detection and tracking. Although IVEA-Py's hybrid detection approach improves the separation of closely spaced vesicles, very dense regions still lead to occlusion, making some detections difficult to resolve. Even when using a more advanced algorithm, such as GVF-Euler integration prior to DBSCAN clustering, vesicles may still be too closely spaced, with separations smaller than the diffraction limit of light microscopy. When vesicles are too close together, IVEA may either classify two vesicles as one or underestimate the number of unique vesicles. These errors not only compound in the detection stage but also propagate into the tracking stage as well. If two vesicles are classified as one during detection, IVEA will be unable to track the motions of the two vesicles correctly since the tracking algorithm will lose the unique identification of each vesicle (**Figure 43**). Since we know that vesicles move in a highly irregular and non-linear manner, trajectories can easily be switched among neighboring objects, preventing the recovery of the true path. This limitation further complicates the creation of robust modeling for statistical estimates of trajectory prediction, as the underlying motion is not always simple, linear, or deterministic. These limitations demonstrate the physical limitations associated with optical resolution, as well as the inherent difficulty of tracking vesicles in densely populated regions.

There are also some trade-offs from a computational perspective. The LSTM classifier is a reasonable baseline for stationary event classification (due to its relative computational efficiency), but the eViT module offers improved spatiotemporal representations while using significantly more GPU resources. If the graphical video memory is insufficient, the maximum allowed patch size and maximum batch size are limited to half-scale outputs. Thus, the potential of the transformer model cannot be fully captured on standard hardware during the training phase. The hybrid detection pipeline developed in IVEA-Py also adds additional detection model parameters, such as DoG ratio, and even granule size presets, which also require tuning during application. To address this possibility, there are preset parameter profiles (small, medium, large vesicles), but even so, if the datasets are different and do not reflect the standard dataset granule size, advanced users may have to perform fine-tuning of the parameters again.

The Java-based implementation of IVEA in Fiji is also a compromise because it is more convenient for the wider imaging community, since the software can be used without any coding experience. However, the limitations of Java for integrating with modern machine learning libraries and for efficient detection and tracking routines are a significant consequence of needing Java. The Python implementation addresses these issues by improving computational efficiency and flexibility. However, such a platform requires a separate environment, which some users may find less appealing.

As a final point, while IVEA reduces user bias compared with manual annotation or rule-based pipelines, it does not eliminate it entirely. The software includes an automated parameter estimation module that adapts thresholds and prominence levels to identify ROI candidates. However, in low-signal datasets with weak or irregular intensity profiles, this automation may fail to detect suitable local maxima. For instance, when the optimal prominence lies within a narrow range (e.g., 10–20) that cannot be reached through the default incremental search, it would require manual adjustment of sensitivity parameters. In such cases, reproducibility is maintained through automatic logging of all applied settings, allowing users to reapply or verify parameters across analyses.

In summary, IVEA extends the capabilities of vesicle analysis through an adaptable, machine-learning-based approach that accommodates diverse datasets and cell types. However, its limitations in accommodating long-lasting events, pinpointing extreme vesicle aggregation, and balancing computational demands offer opportunities for future improvement. These constraints do not diminish the platform's productivity but instead delineate the space in which it operates and provide a basis for the next iteration of methodological advances.

4.6 Outlook

The development of IVEA has established a versatile framework for the detection, classification, and tracking of exocytosis events; it also presents numerous options for refinement and expansion. Multiple limitations discussed in this work directly address future improvement potential, both in terms of algorithmic sophistication and applicability across a range of biological systems.

One potential avenue would be to integrate adaptive parameterization. Currently, parameters including the DoG ratio or LoG kernel and sigma size, granule size, and detection thresholds must either be manually defined or rely on preset guidelines. While the inclusion of preset profiles greatly simplifies use, fully automated parameter learning would further enhance reproducibility and accessibility. The future IVEA could use Bayesian optimization, reinforcement learning (Mnih et al., 2015), or meta-learning (Finn et al., 2017) to allow the platform to learn and adapt parameterization settings from the datasets it processes to decrease manual intervention and bias.

Detection strategies could also be improved by exploring the hybrid pipeline in place. For example, learned flow-field representations (L. Li et al., 2025), could be utilized to replace the fixed GVF

framework (G. Li et al., 2007). This would facilitate the model to learn directly from data how vesicle movements converged. These approaches would use the same principles as optical flow estimation in computer vision (Horn & Schunck, 1981), but adapted for the scale of vesicle trafficking and dynamics. Also, integrating multi-object probabilistic tracking frameworks could help alleviate some of the challenges posed by overlapping vesicles, which can lead to ID assignment switches. Adding a probabilistic approach may help resolve the ambiguity over a longer trajectory that might improve tracking in these challenging conditions.

A second major outlook involves types of events being examined. IVEA is currently optimized to examine transient burst-like fusion events, but it can be extended to examine long or non-burst-associated events. Events such as those associated with Synaptobrevin-SEP probes, or in combination with FM dyes (Ryan, 1999) in neurons or even other biological activities. Presumably, this would involve retraining or fine-tuning the classifiers on datasets that have been selectively enriched with the target types of events.

Looking further ahead, inspiration can be drawn from recent advances in foundation models such as the Segment Anything Model (SAM) (Kirillov et al., 2023) and its adaptation for microscopy, Segment Anything for Microscopy (μSAM) (Archit et al., 2025). These architectures can generalize and provide robust performance across domains through combining large-scale pretraining and prompt-based interactivity, something similar to huge LLM models (i.e., ChatGPT OpenAI, Grok xAI, Gemini Google, and others). This allows the user to prompt the model with minimal information and obtain meaningful outputs. While SAM is primarily designed for segmentation, its fundamental principle could be adapted for classifying dynamic events in time-lapse microscopy. SAM has promptable, general-purpose representation learning with task-specific fine-tuning. These properties provide a conceptual model that could be adapted. In the context of IVEA, this could allow the user to mark a candidate region in a video where activity is occurring. Subsequently, this will allow the eViT module to refine the spatiotemporal characterization of the event directly and learn the new event's features.

The main challenges here are substantial; SAM's approach to generalizing through prompts was developed originally for segmentation. This needs to be translated into the classification of temporal activity detection. Here, vesicle fusion events appear as heterogeneous Gaussian-like intensity spreads rather than discrete object units. Additionally, the lack of large-scale annotated datasets for exocytosis events makes it difficult to train such models in the near future. Developing such a foundation for classification would likely require both self-supervised pretraining on large microscopy video collections and task-specific fine-tuning with curated annotations, a scale of effort beyond what is currently feasible but worth considering as a long-term goal. Developing such a foundation for classification will take a considerable amount of effort. This would combine both self-supervised pretraining on large microscopy video collections and task-specific fine-tuning with curated

annotations. In terms of effort, this is a significant human work that can be considered as a long-term goal.

Integration with segmentation modules is another avenue that could yield great benefit. The ability to specify the location of cellular structures, e.g., membranes, synaptic terminals, and immune synapses, provides a context for capturing vesicle detections and tracks in biologically relevant areas. This could be especially important for slow or diffuse synaptic exocytosis events, which propagate over a larger area of the membrane. In this regard, segmentation approaches similar to SAM or even Cellpose (Pachitariu & Stringer, 2022; Stringer et al., 2021) may be useful. Cellpose's flow-based gradient vector field approach is not ideally suited for segmenting neurons or other complex morphologies, but the iterative user refinement and common graphical output are helpful design cues. A similar semi-automated training loop in IVEA, where the user accepts/rejects the classification of events, would allow the model to improve using the users' own data. In this regard, these developments may further IVEA's usability towards a more community-engaged and adaptive framework. Implementing segmentation and user refinement, along with robust detection and classification, could further broaden the accessibility of the platform beyond exocytosis.

Tracking can also be further improved; this would be by incorporating deep learning or advanced mathematical models for trajectory prediction and association. Recurrent or Transformer-based networks could learn motion patterns directly from data, which can provide more consistent tracking across occlusions or irregular motion. These approaches offer a better chance for future development without committing to a single implementation strategy. Eventually, a prominent technical consideration has to do with platform integration. Similar to IVEA-Py, extending the tracking functionality in a Java-based Fiji plugin could encourage greater adoption within the imaging community. This dual-platform strategy would ensure that IVEA remains accessible to users accustomed to Fiji/ImageJ. Concurrently, it offers better capabilities in Python for those seeking maximum performance and flexibility.

In summary, the future of IVEA lies in extending its adaptability, expanding its biological scope, and lowering the barriers to its adoption. IVEA could move beyond its current focus on exocytosis activity by incorporating adaptive parameterization, probabilistic tracking, extended event types with related segmentation, and community-driven training. Such an evolution would position IVEA alongside tools like SAM and Cellpose as a widely adopted, community-standard solution for image-based biological discovery.

References

[1].Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., . . . Zheng, X. (2023). TensorFlow: Large-scale machine learning on heterogeneous systems (Version Version 2.10.1): Google Inc. Retrieved from https://www.tensorflow.org

[2].Apergis, I., Bayliss, D., Asimakoulas, L., Chote, P., McCormac, J., Mitchell, M. A., . . . Wheatley, P. (2025). High-Precision Photometry with a scientific CMOS Camera: I Lab Testing of the Marana camera. arXiv:2510.14484. doi:10.48550/arXiv.2510.14484

[3].Archit, A., Freckmann, L., Nair, S., Khalid, N., Hilt, P., Rajashekar, V., . . . Pape, C. (2025). Segment Anything for Microscopy. *Nat Methods, 22*(3), 579-591. doi:10.1038/s41592-024-02580-4

[4].Arefi, F., Mansourian, A. M., & Kasaei, S. (2024). Deep spectral improvement for unsupervised image instance segmentation. *PLoS One, 19*(10), e0307432. doi:10.1371/journal.pone.0307432

[5].Balzarotti, F., Eilers, Y., Gwosch, K. C., Gynnå, A. H., Westphal, V., Stefani, F. D., . . . Hell, S. W. (2017). Nanometer resolution imaging and tracking of fluorescent molecules with minimal photon fluxes. *Science, 355*(6325), 606-612. doi:doi:10.1126/science.aak9913

[6].Becherer, U., Pasche, M., Nofal, S., Hof, D., Matti, U., & Rettig, J. (2007). Quantifying exocytosis by combination of membrane capacitance measurements and total internal reflection fluorescence microscopy in chromaffin cells. *PLoS One, 2*(6), e505. doi:10.1371/journal.pone.0000505

[7].Chang, H. F., Schirra, C., Pattu, V., Krause, E., & Becherer, U. (2023). Lytic granule exocytosis at immune synapses: lessons from neuronal synapses. *Front Immunol, 14*, 1177670. doi:10.3389/fimmu.2023.1177670

[8].Chi, P., Greengard, P., & Ryan, T. A. (2001). Synapsin dispersion and reclustering during synaptic activity. *Nat Neurosci, 4*(12), 1187-1193. doi:10.1038/nn756

[9].Chouaib, A. A., Chang, H. F., Khamis, O. M., Alawar, N., Echeverry, S., Demeersseman, L., . . . Becherer, U. (2025). Highly adaptable deep-learning platform for automated detection and analysis of vesicle exocytosis. *Nat Commun, 16*(1), 6450. doi:10.1038/s41467-025-61579-3

[10].Dataset, Z. (2025). *IVEA dataset: Highly adaptable deep-learning platform for automated detection and analysis of vesicle exocytosis*. Retrieved from: https://doi.org/10.5281/zenodo.13153017

[11].Desai, S. S., & Ramaswamy, H. G. (2020). Ablation-CAM: Visual Explanations for Deep Convolutional Network via Gradient-free Localization. *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 972-980.

[12].Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., . . . Houlsby, N. (2021). An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. *International Conference on Learning Representations* 2010.11929. doi:10.48550/arXiv.2010.11929

[13].Elizarova, S., Chouaib, A. A., Shaib, A., Hill, B., Mann, F., Brose, N., . . . Daniel, J. A. (2022). A fluorescent nanosensor paint detects dopamine release at axonal varicosities with high spatiotemporal resolution. *Proc Natl Acad Sci U S A, 119*(22), e2202842119. doi:10.1073/pnas.2202842119

[14].Elman, J. L. (1990). Finding Structure in Time. *Cognitive Science, 14*(2), 179-211. doi:Doi 10.1016/0364-0213(90)90002-E

[15].Ester, M., Kriegel, H.-P., Sander, J., & Xu, X. (1996). *A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise.* Paper presented at the Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96).

[16].Fernandez-Alfonso, T., Kwan, R., & Ryan, T. A. (2006). Synaptic vesicles interchange their membrane proteins with a large surface reservoir during recycling. *Neuron, 51*(2), 179-186. doi:10.1016/j.neuron.2006.06.008

[17].Finn, C., Abbeel, P., & Levine, S. (2017). Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. arXiv:1703.03400. doi:10.48550/arXiv.1703.03400

[18].Ge, L., Shin, W., Arpino, G., Wei, L., Chan, C. Y., Bleck, C. K. E., . . . Wu, L. G. (2022). Sequential compound fusion and kiss-and-run mediate exo- and endocytosis in excitable cells. *Sci Adv, 8*(24), eabm6049. doi:10.1126/sciadv.abm6049

[19].He, K., Zhang, X., Ren, S., & Sun, J. (2016, 27-30 June 2016). *Deep Residual Learning for Image Recognition.* Paper presented at the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

[20].Hensel, T. A., Wirth, J. O., Schwarz, O. L., & Hell, S. W. (2025). Diffraction minima resolve point scatterers at few hundredths of the wavelength. *Nature Physics, 21*(3), 412-420. doi:10.1038/s41567-024-02760-1

[21].Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation, 9*(8), 1735-1780. doi:DOI 10.1162/neco.1997.9.8.1735

[22].Horn, B. K. P., & Schunck, B. G. (1981). Determining optical flow. *Artificial Intelligence, 17*(1), 185-203. doi:https://doi.org/10.1016/0004-3702(81)90024-2

[23].Hugo, S., Dembla, E., Halimani, M., Matti, U., Rettig, J., & Becherer, U. (2013). Deciphering dead-end docking of large dense core vesicles in bovine chromaffin cells. *J Neurosci, 33*(43), 17123-17137. doi:10.1523/JNEUROSCI.1589-13.2013

[24].Jahn, R., & Sudhof, T. C. (1994). Synaptic vesicles and exocytosis. *Annu Rev Neurosci, 17*, 219-246. doi:10.1146/annurev.ne.17.030194.001251

[25].Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., . . . Hassabis, D. (2021). Highly accurate protein structure prediction with AlphaFold. *Nature, 596*(7873), 583-589. doi:10.1038/s41586-021-03819-2

[26].Karim, F., Majumdar, S., Darabi, H., & Harford, S. (2019). Multivariate LSTM-FCNs for time series classification. *Neural Netw, 116*, 237-245. doi:10.1016/j.neunet.2019.04.014

[27].Kirillov, A., He, K., Girshick, R., Rother, C., & Dollár, P. (2018). Panoptic Segmentation. arXiv:1801.00868. doi:10.48550/arXiv.1801.00868

[28].Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., . . . Girshick, R. (2023). Segment Anything. arXiv:2304.02643. doi:10.48550/arXiv.2304.02643

[29].Li, G., Liu, T., Nie, J., Guo, L., Malicki, J., Mara, A., . . . Wong, S. T. (2007). Detection of blob objects in microscopic zebrafish images based on gradient vector diffusion. *Cytometry A, 71*(10), 835-845. doi:10.1002/cyto.a.20436

[30].Li, H., Mao, Y., Yin, Z., & Xu, Y. (2017). A Hierarchical Convolutional Neural Network for vesicle fusion event classification. *Comput Med Imaging Graph, 60*, 22-34. doi:10.1016/j.compmedimag.2017.04.003

[31].Li, L., Zhang, W., Li, Y., Jiang, C., & Wang, Y. (2025). An attention-enhanced Fourier neural operator model for predicting flow fields in turbomachinery Cascades. *Physics of Fluids, 37*(3), 036121. doi:10.1063/5.0254681

[32].Li, Y., Miao, N., Ma, L., Shuang, F., & Huang, X. (2023). Transformer for object detection: Review and benchmark. *Engineering Applications of Artificial Intelligence, 126*, 107021. doi:https://doi.org/10.1016/j.engappai.2023.107021

[33].Linkert, M., Rueden, C. T., Allan, C., Burel, J. M., Moore, W., Patterson, A., . . . Swedlow, J. R. (2010). Metadata matters: access to image data in the real world. *J Cell Biol, 189*(5), 777-782. doi:10.1083/jcb.201004104

[34].Lisanza, S. L., Gershon, J. M., Tipps, S. W. K., Sims, J. N., Arnoldt, L., Hendel, S. J., . . . Baker, D. (2025). Multistate and functional protein design using RoseTTAFold sequence space diffusion. *Nat Biotechnol, 43*(8), 1288-1298. doi:10.1038/s41587-024-02395-w

[35].Liu, J., Verweij, F. J., van Niel, G., Galli, T., Danglot, L., & Bun, P. (2024). ExoJ - a Fiji/ImageJ2 plugin for automated spatiotemporal detection and analysis of exocytosis. *J Cell Sci, 137*(20). doi:10.1242/jcs.261938

[36].Long, J., Shelhamer, E., & Darrell, T. (2014). Fully Convolutional Networks for Semantic Segmentation. arXiv:1411.4038. doi:10.48550/arXiv.1411.4038

[37].Lowe, D. G. (2004). Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision, 60*(2), 91-110. doi:10.1023/B:VISI.0000029664.99615.94

[38].Macqueen, J. (1965). On Convergence of K-Means and Partitions with Minimum Average Variance. *Annals of Mathematical Statistics, 36*(3), 1084-&. Retrieved from <Go to ISI>://WOS:A19656741200075

[39]. MATLAB (Version Version 9.15 (R2023b)). (2023). Natick, Massachusetts: The MathWorks Inc. Retrieved from https://www.mathworks.com

[40].Meyes, R., Lu, M., Waubert de Puiseau, C., & Meisen, T. (2019). Ablation Studies in Artificial Neural Networks. arXiv:1901.08644. doi:10.48550/arXiv.1901.08644

[41].Mildenhall, B., Srinivasan, P. P., Tancik, M., Barron, J. T., Ramamoorthi, R., & Ng, R. (2020a, 2020//). *NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis.* Paper presented at the Computer Vision – ECCV 2020, Cham.

[42].Mildenhall, B., Srinivasan, P. P., Tancik, M., Barron, J. T., Ramamoorthi, R., & Ng, R. (2020b). NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. arXiv:2003.08934. doi:10.48550/arXiv.2003.08934

[43].Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., . . . Hassabis, D. (2015). Human-level control through deep reinforcement learning. *Nature, 518*(7540), 529-533. doi:10.1038/nature14236

[44].O'Shaughnessy, E. C., Lam, M., Ryken, S. E., Wiesner, T., Lukasik, K., Zuchero, J. B., . . . Gupton, S. L. (2024). pHusion - a robust and versatile toolset for automated detection and analysis of exocytosis. *J Cell Sci, 137*(20). doi:10.1242/jcs.261828

[45].Olivo-Marin, J.-C. (2002). Extraction of spots in biological images using multiscale products. *Pattern Recognition, 35*(9), 1989-1996. doi:https://doi.org/10.1016/S0031-3203(01)00127-3

[46].Pachitariu, M., & Stringer, C. (2022). Cellpose 2.0: how to train your own model. *Nat Methods, 19*(12), 1634-1641. doi:10.1038/s41592-022-01663-4

[47].Pham, D. L., Xu, C., & Prince, J. L. (2000). Current methods in medical image segmentation. *Annu Rev Biomed Eng, 2*, 315-337. doi:10.1146/annurev.bioeng.2.1.315

[48].Qu, J., Li, J., Wang, H., Lan, J., Huo, Z., & Li, X. (2025). Decoding the role of microtubules: a trafficking road for vesicle. *Theranostics, 15*(11), 5138-5152. doi:10.7150/thno.110120

[49].Raghu, M., Zhang, C., Kleinberg, J., & Bengio, S. (2019). Transfusion: Understanding Transfer Learning for Medical Imaging. arXiv:1902.07208. doi:10.48550/arXiv.1902.07208

[50].Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2015). You Only Look Once: Unified, Real-Time Object Detection. arXiv:1506.02640. doi:10.48550/arXiv.1506.02640

[51].Rolnick, D., Ahuja, A., Schwarz, J., Lillicrap, T. P., & Wayne, G. (2018). Experience Replay for Continual Learning. arXiv:1811.11682. doi:10.48550/arXiv.1811.11682

[52].Ronneberger, O., Fischer, P., & Brox, T. (2015, 2015//). *U-Net: Convolutional Networks for Biomedical Image Segmentation.* Paper presented at the Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015, Cham.

[53].Rueden, C. T., Schindelin, J., Hiner, M. C., DeZonia, B. E., Walter, A. E., Arena, E. T., & Eliceiri, K. W. (2017). ImageJ2: ImageJ for the next generation of scientific image data. *BMC Bioinformatics, 18*(1), 529. doi:10.1186/s12859-017-1934-z

[54].Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning Representations by Back-Propagating Errors. *Nature, 323*(6088), 533-536. doi:DOI 10.1038/323533a0

[55].Ryan, T. A. (1999). Inhibitors of myosin light chain kinase block synaptic vesicle pool mobilization during action potential firing. *J Neurosci, 19*(4), 1317-1323. doi:10.1523/JNEUROSCI.19-04-01317.1999

[56].Schmied, C., Soykan, T., Bolz, S., Haucke, V., & Lehmann, M. (2021). SynActJ: Easy-to-Use Automated Analysis of Synaptic Activity. *Frontiers in Computer Science, 3*. doi:10.3389/fcomp.2021.777837

[57].Schneider, C. A., Rasband, W. S., & Eliceiri, K. W. (2012). NIH Image to ImageJ: 25 years of image analysis. *Nat Methods, 9*(7), 671-675. doi:10.1038/nmeth.2089

[58].Shaib, A. H., Chouaib, A. A., Chowdhury, R., Altendorf, J., Mihaylov, D., Zhang, C., . . . Rizzoli, S. O. (2024). One-step nanoscale expansion microscopy reveals individual protein shapes. *Nat Biotechnol*. doi:10.1038/s41587-024-02431-9

[59].Shaib, A. H., Chouaib, A. A., Chowdhury, R., Altendorf, J., Mihaylov, D., Zhang, C., . . . Rizzoli, S. O. (2025). One-step nanoscale expansion microscopy reveals individual protein shapes. *Nat Biotechnol, 43*(9), 1539-1547. doi:10.1038/s41587-024-02431-9

[60].Shaib, A. H., Staudt, A., Harb, A., Klose, M., Shaaban, A., Schirra, C., . . . Becherer, U. (2018). Paralogs of the Calcium-Dependent Activator Protein for Secretion Differentially Regulate Synaptic Transmission and Peptide Secretion in Sensory Neurons. *Front Cell Neurosci, 12*, 304. doi:10.3389/fncel.2018.00304

[61].Simonyan, K., & Zisserman, A. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv:1409.1556. doi:10.48550/arXiv.1409.1556

[62].Solovyev, R., Wang, W., & Gabruseva, T. (2021). Weighted boxes fusion: Ensembling boxes from different object detection models. *Image and Vision Computing, 107*, 104117. doi:https://doi.org/10.1016/j.imavis.2021.104117

[63].Staudt, A., Ratai, O., Bouzouina, A., Fecher-Trost, C., Shaaban, A., Bzeih, H., . . . Becherer, U. (2022). Localization of the Priming Factors CAPS1 and CAPS2 in Mouse Sensory Neurons Is Determined by Their N-Termini. *Front Mol Neurosci, 15*, 674243. doi:10.3389/fnmol.2022.674243

[64].Stringer, C., Wang, T., Michaelos, M., & Pachitariu, M. (2021). Cellpose: a generalist algorithm for cellular segmentation. *Nat Methods, 18*(1), 100-106. doi:10.1038/s41592-020-01018-x

[65].Strutz, T. (2021). The Distance Transform and its Computation. arXiv:2106.03503. doi:10.48550/arXiv.2106.03503

[66].Tan, C., Sun, F., Kong, T., Zhang, W., Yang, C., & Liu, C. (2018). A Survey on Deep Transfer Learning. arXiv:1808.01974. doi:10.48550/arXiv.1808.01974

[67]. TensorFlow for Java (Version Version 1.15.0). (2019): Google Inc. Retrieved from https://www.tensorflow.org/jvm

[68].Tian, Q., & Lipp, P. (2021). Apparent calcium spark properties and fast-scanning 2D confocal imaging modalities. *Cell Calcium, 93*, 102303. doi:10.1016/j.ceca.2020.102303

[69].Tran, D., Wang, H., Torresani, L., Ray, J., LeCun, Y., & Paluri, M. (2018, 18-23 June 2018). *A Closer Look at Spatiotemporal Convolutions for Action Recognition.* Paper presented at the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition.

[70].Tsuboi, T., & Rutter, G. A. (2003). Multiple forms of "kiss-and-run" exocytosis revealed by evanescent wave microscopy. *Curr Biol, 13*(7), 563-567. doi:10.1016/s0960-9822(03)00176-3

[71].Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., . . . Polosukhin, I. (2017). Attention Is All You Need. arXiv:1706.03762. doi:10.48550/arXiv.1706.03762

[72].Yosinski, J., Clune, J., Bengio, Y., & Lipson, H. (2014). How transferable are features in deep neural networks? , arXiv:1411.1792. doi:10.48550/arXiv.1411.1792

[73].Yuan, T., Lu, J., Zhang, J., Zhang, Y., & Chen, L. (2015). Spatiotemporal detection and analysis of exocytosis reveal fusion "hotspots" organized by the cytoskeleton in endocrine cells. *Biophys J, 108*(2), 251-260. doi:10.1016/j.bpj.2014.11.3462

[74].Zenodo, I. (2025). IVEA software: Highly adaptable deep-learning platform for automated detection and analysis of vesicle exocytosis: Zenodo. Retrieved from https://doi.org/10.5281/zenodo.15498139

[75].Zhang, M. L., Ti, H. Y., Wang, P. Y., & Li, H. (2021). Intracellular transport dynamics revealed by single-particle tracking. *Biophys Rep, 7*(5), 413-427. doi:10.52601/bpr.2021.210035

5 Appendix

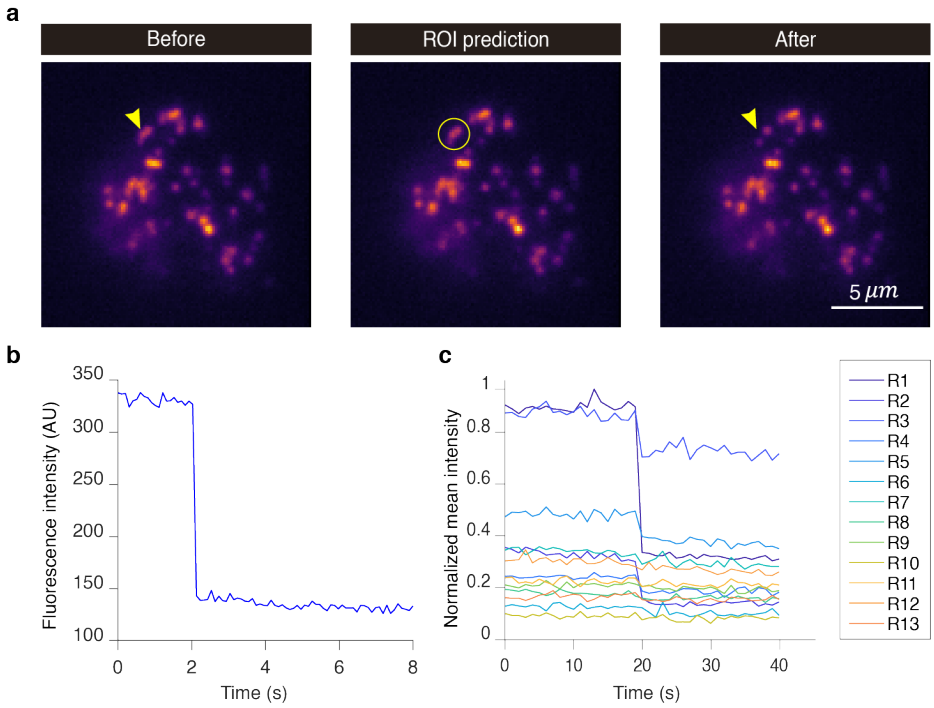


Figure 41 Chromaffin cell small clustered granules LSTM pattern display (adapted from (Chouaib et al., 2025)).

a. Images of a chromaffin cell, showcasing a granule exocytotic event occurring withing a granule aggregation. The granules were stained through over-expression of NPY-mCherry (pH-insensitive, data from*(Becherer et al., 2007)*). **b**. Displays the event fluorescence intensity profile over 80 frames. **c**. Displays the fluorescence intensity profiles of 13 spatial subregions extracted from a single detected ROI. Only one subregion exhibited a clear decrease in intensity corresponding to the fusion event, while the remaining regions showed minimal change. This illustrates the weak nature of the signal in chromaffin cells, which the LSTM model often misses to classify. If the event were stronger, all subregions would exhibit a concurrent intensity drop (**Figure 29i**).

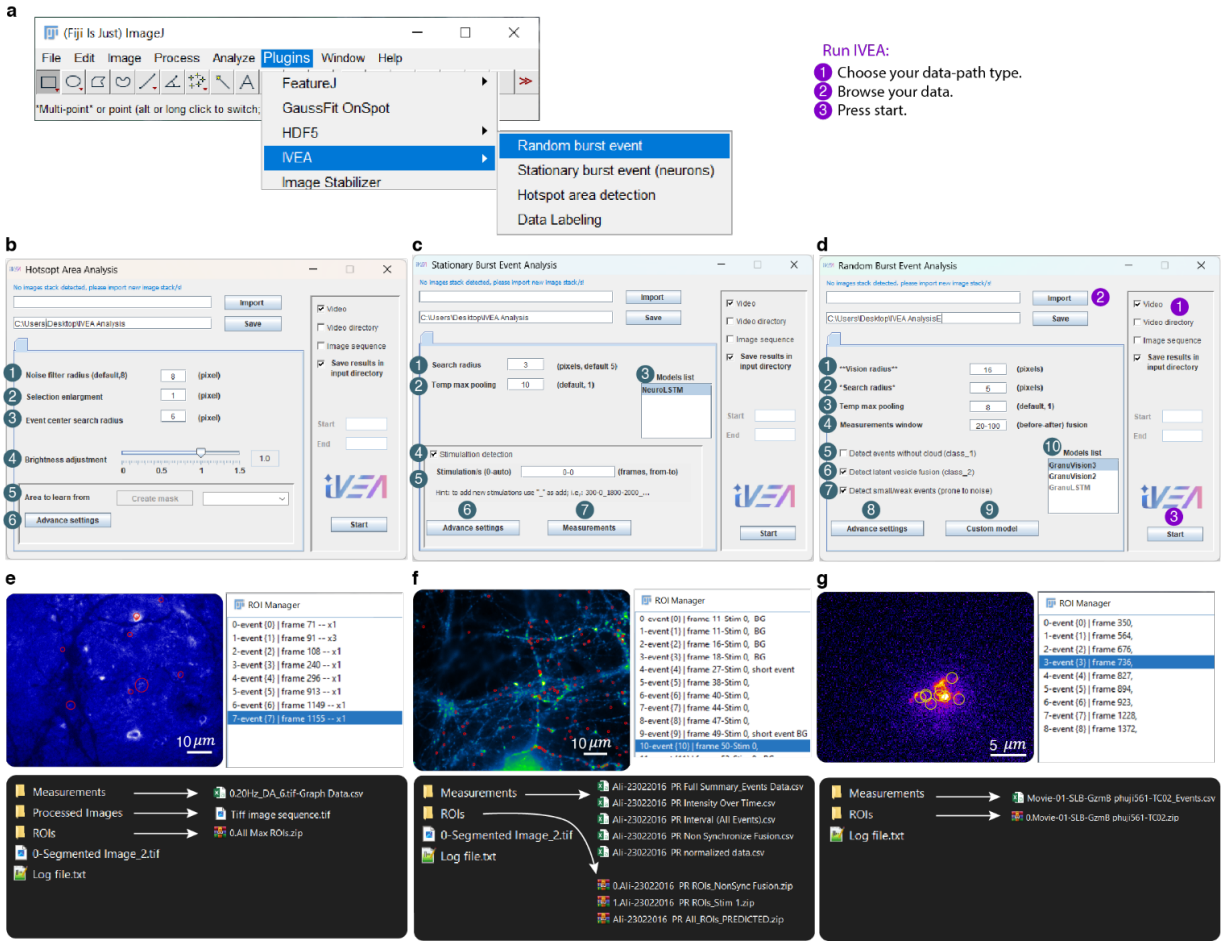


Figure 42. IVEA modules graphical user interface and output results (adapted from (Chouaib et al., 2025))
a. Location of the IVEA plugin within the Fiji environment. **b**. *Hotspot Area Events Analysis:* (1) Noise filtering used to extract candidate events from the image; increasing the filter value reduces detection sensitivity without influencing measurement results. (2) Option to expand the detected mask by *n* pixels. (3) Search radius parameter for identifying event identity. (4) Brightness adjustment control that compensates for image intensity fluctuations. (5) Region selection for adaptive learning. (6) Advanced settings tab containing control parameters automatically optimized by the software. **c**. *Stationary Burst Events Analysis:* (1–2) Search radius defining the ROI around each event. (2–3) Temporal max-pooling to reduce video length via a moving-maximum projection (a value of 1 disables frame reduction). (3–10) List of pretrained neural network models provided within IVEA. (4) Option to enable or exclude stimulation-related event detection. (5) Defines the stimulation interval (**a–b**), which is automatically determined when either value is set to zero. (6) Default parameter controls for stationary burst analysis. (7) Measurement configuration options for exported results. **d**. *Random Burst Events Analysis:* (1) Vision radius specifying the spatial window size used by the network. (4) Temporal measurement interval (**a–b**). (5) Option to include pH-insensitive events. (6) Option to include latent granule fusions. (7) "Small or weak events" mode disables the intensity filter to retain low-signal events close to background noise levels. (8) Default parameter controls for random burst analysis. (9) Export options for training data and use of custom neural networks. **e–g**, Representative output interfaces for hotspot area, stationary burst, and random burst event analyses, respectively.

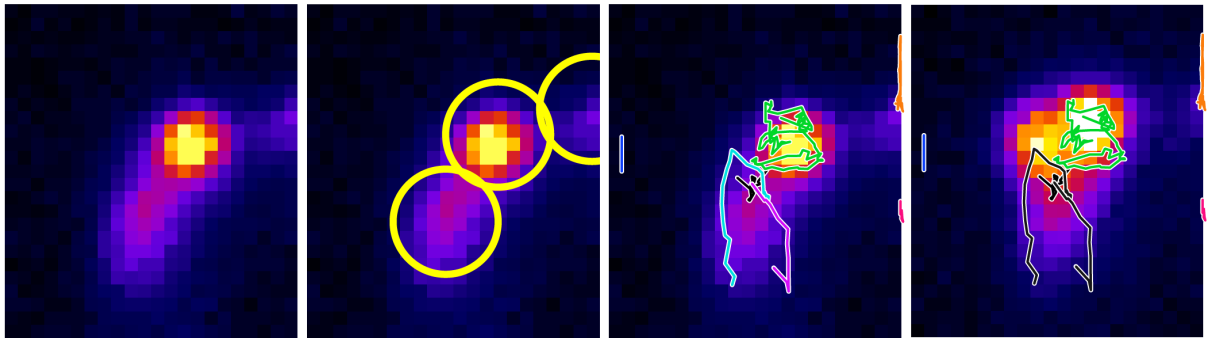


Figure 43. Granule occlusion artifacts in trajectory linking.
From left to right: (1) raw TIRF image showing two partially overlapping granules; (2) ROI overlay illustrating individual detection regions; (3) current tracking output, where a single granule trajectory (cyan, black, and magenta) is incorrectly segmented into three separate traces due to transient occlusion; and (4) corrected trajectory linking after occlusion handling, accurately representing the continuous motion of the same granule.

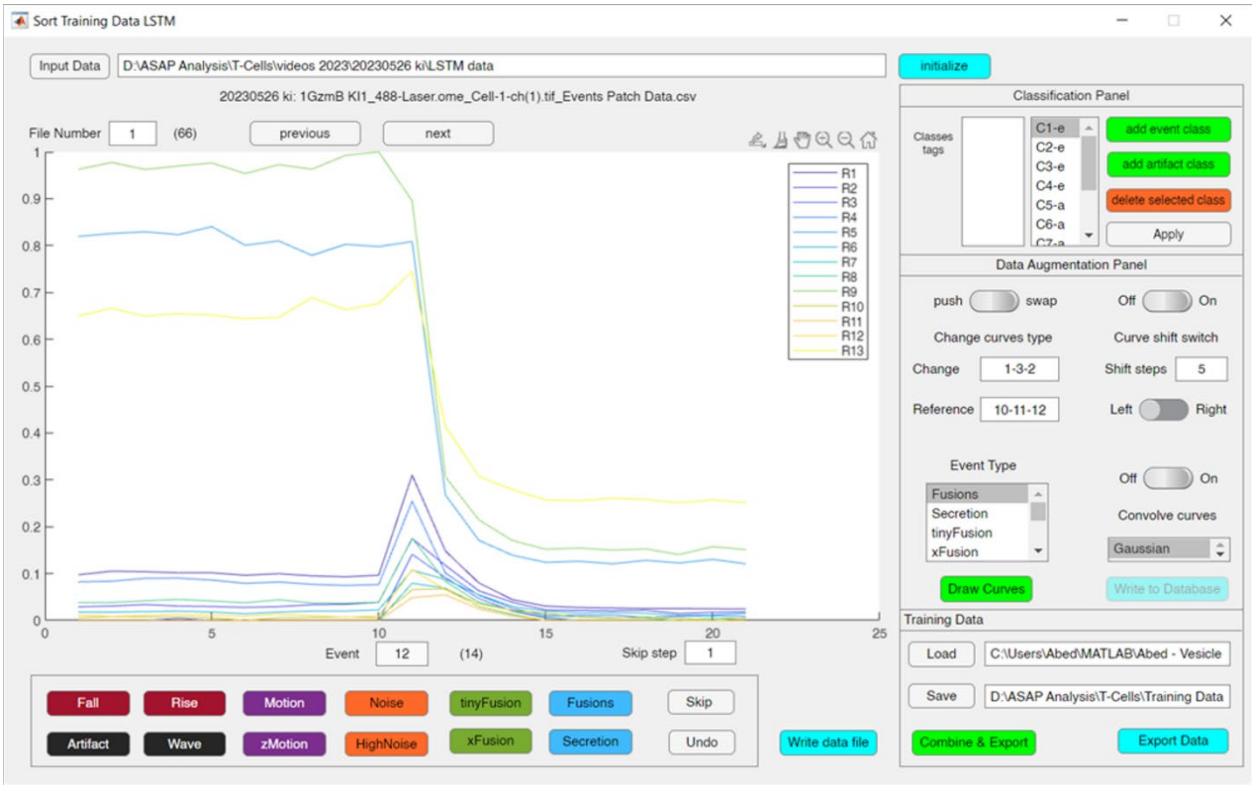


Figure 44 IVEA LSTM Network MATLAB Labeling Application.
Custom MATLAB graphical interface developed for labeling and visualizing temporal intensity profiles of candidate exocytosis events. The application displays the mean intensity of 13 spatial regions (R1–R13) as curves plotted over time, allowing visual inspection of event dynamics. Classification buttons at the bottom enable rapid assignment of event categories, while most interface functions are mapped to keyboard shortcuts to accelerate manual labeling. The side control panel provides tools for managing class labels and performing basic data augmentation operations, such as temporal shifting or curve flipping. This interface enabled efficient generation of large, labeled training datasets for the LSTM network within IVEA.

Table 10. Comparative overview of existing tools for fluorescence-based activity recognition.

The table summarizes software tools commonly used for analyzing dynamic fluorescence signals in live-cell microscopy. "Activity recognition" refers to the capability to detect dynamic or transient cellular activity rather than static morphology. IVEA demonstrates generalization beyond exocytosis, having also detected calcium sparks without retraining. PTrack II and TrackMate primarily perform particle or object tracking without classification of biological events. pHusion implements a mathematical model of fluorescence decay kinetics, rather than a data-driven recognition approach. ICY and other Fiji-based tools serve as general platforms for image analysis but are not domain-specific automated systems for exocytosis or comparable activity detection.

Software	Platform	Activity Recognition (General)	Specific to Exocytosis	Deep Learning	Automated Parameterization	Trajectory / Event Linking	Transfer Learning / Retraining	Cross-Platform Open Source
IVEA / IVEA-Py	Fiji / Python	✅ Yes (Exocytosis, Calcium Sparks, etc.)	✅	✅	✅	✅	✅	✅
TrackMate	Fiji	✅ (General object tracking)	❌	❌	❌	✅	❌	✅
ICY	Standalone (Java)	✅ (General image analysis)	❌	❌	❌	✅	❌	✅
ExoJ	Fiji	❌	✅ (pH -sensitive vesicles)	❌	❌	Limited	❌	✅
SynActJ	Fiji	❌	✅ (Synaptic activity)	❌	❌	Limited	❌	✅
pHusion	MATLAB / Python	❌	✅ (Mathematical fluorescence decay model)	❌	Partial (Manual)	❌	❌	⚠️ (Proprietary base)
PTrack II	Fiji	✅ (Particle tracking)	❌	❌	❌	✅	❌	✅

6 List of publications

Published

[1]: Chouaib, A.A., Chang, HF., Khamis, O.M. *et al.* Highly adaptable deep-learning platform for automated detection and analysis of vesicle exocytosis. Nature Communications **16**, 6450 (2025). https://doi.org/10.1038/s41467-025-61579-3.

[2]: Shaib, A.H., Chouaib, A.A., Chowdhury, R. et al. One-step nanoscale expansion microscopy reveals individual protein shapes. Nature Biotechnology (2024). https://doi.org/10.1038/s41587-024-02431-9. Nature Biotechnology.

[3]: RChowdhury, R., Mimoso, T., Chouaib, A.A. et al. Microtubules as a versatile reference standard for expansion microscopy. Nature Commun Biology 8, 499 (2025). https://doi.org/10.1038/s42003-025-07967-3.

[4]: Daniel, J. A., Elizarova, S., Shaib, A. A., Chouaib, A. A., . . . Brose, Nils, . . . (2023). An intellectual-disability-associated mutation of the transcriptional regulator NACC1 impairs glutamatergic neurotransmission. Front Molecular Neuroscience, 16, 1115880. doi:10.3389/fnmol.2023.1115880

[5]: Elizarova, S., Chouaib, A. A., . . . Brose, Nils., . . . Daniel, J. A. (2022). A fluorescent nanosensor paint detects dopamine release at axonal varicosities with high spatiotemporal resolution. Proc Natl Acad Sci U S A (PNAS), 119(22), e2202842119. doi:10.1073/pnas.2202842119.

[6]: Patrícia Santos, ... , Jeong Seop Rhee, Abed A. Chouaib, Silvio O. Rizzoli, James Daniel, and Tiago F. Outeiro, Glutamatergic synaptic resilience to overexpressed human alpha-synuclein. Nature Parkinson's disease, https://doi.org/10.1038/s41531-025-01085-x

[7]: Xuemei Li, ... , Meltem Hohmann, Nadia Alawar, Abed Chouaib, Ute Becherer, Varsha Pattu, Jens Rettig, Elmar Krause, Hsin-Fang Chang, bioRxiv 2025.01.29.635520; doi: https://doi.org/10.1101/2025.01.29.635520

ACKNOWLEDGEMENTS

First and foremost, I would like to express my deepest gratitude to my supervisor, Prof. Dr. Ute Becherer, for her unwavering support and guidance throughout my PhD journey. Her scientific insight, patience, and steady encouragement have been essential to both my research and my personal growth. Even though we come from different disciplines, she, from biology, and I, from computer science, she always made time to discuss technical details, listen to my ideas, and help me refine complex algorithms. I also deeply appreciated the effort she put into searching for relevant articles and sending me material to explore; these gestures meant a lot. To me, Ute has been more than a supervisor; she has been a mentor with a generous spirit, someone who truly looks after her students and guides them with kindness.

I am profoundly grateful to Dr. Ali H. Shaib from the University of Göttingen, my older brother and closest collaborator on different projects. His exceptional scientific vision and support have been pivotal throughout my journey. As the pioneer of the one-nanometer super-resolution expansion method, he entrusted me with the opportunity to develop the ONE-Platform for one-step nanoscale expansion, believing in my ability to bring this concept to life through algorithm development. His mentorship, trust, and example as a scientist have deeply shaped my work and determination. Thank you, Ali, for being a father, a guiding mentor, and a brother whose faith and support I will always carry with me.

I would like to sincerely thank Dr. Hsin-Fang Chang for providing data for the IVEA project and for her kind friendship and support throughout my time in the lab. Beyond our scientific collaboration, her encouragement, insightful conversations, and kindness both in and out of the lab made this journey more enjoyable. My heartfelt thanks also go to Dr. Meltem Homann, whose friendship and positive spirit brought warmth and joy into the daily lab life.

Special thanks to Dr. David Steven, Dr. Sushovan Chanda, and Mohammad Mahdi Alawieh for reading the thesis and providing invaluable input.

I want to thank Prof. Lauterbach for his valuable feedback and input on the early stage of the IVEA manuscript development.

To my fellow PhD colleagues and friends — Yasser Medlij, Ahmad Lotfinia, Mohammad Shakier, and Marie-Louise Wirkner, thank you for the stimulating discussions, collaboration, and companionship that made the lab a genuinely inspiring place to work. Your friendship and energy turned long research days into memorable ones.

I am deeply appreciative of all the individuals who provided data for my work, including my lab members and collaborators from Germany (Göttingen, Homburg), Sweden, and France. Your contributions have been vital to the success of my research.

Finally, I thank my family, whose unconditional love has been my foundation. Thank you for supporting me through every challenge and triumph. Special gratitude goes to my mother, whose endless care and faith have carried me through life; to my father; to my sister Rayan; to my younger brother Sami; and a heartfelt thank you to my brother Hesni, who, despite being critically injured while serving in the special forces against terrorism, remains a true hero and a symbol of resilience. Your courage and strength have inspired me to persevere and complete this journey.

The curriculum vitae was removed from the electronic version of the doctoral thesis for reasons of data protection.