Communicating Smartly in Molecular Communication Environments: Neural Networks in the Internet of Bio-Nano Things

Jorge Torres Gómez, *Senior Member, IEEE*, Pit Hofmann, *Graduate Student Member, IEEE*, Lisa Y. Debus, *Graduate Student Member, IEEE*, Osman Tugay Başaran, *Graduate Student Member, IEEE*, Sebastian Lotter, Roya Khanzadeh, Stefan Angerbauer, Bige Deniz Unluturk, *Member, IEEE*, Sergi Abadal, Werner Haselmayr, Frank H.P. Fitzek, *Fellow, IEEE*, Robert Schober, *Fellow, IEEE*, and Falko Dressler, *Fellow, IEEE*

***Abstract*—Recent developments in the Internet of Bio-Nano Things (IoBNT) are laying the groundwork for innovative applications across the healthcare sector. Nanodevices designed to operate within the body, managed remotely via the internet, are envisioned to promptly detect and actuate on potential diseases. In this vision, an inherent challenge arises due to the limited capabilities of individual nanosensors; specifically, nanosensors must communicate with one another to collaborate as a cluster. Aiming to research the boundaries of the clustering capabilities, this survey emphasizes data-driven communication strategies in molecular communication (MC) channels as a means of linking nanosensors. Due to the dynamics of MC environments, communication at the nanoscale faces new challenges where detailed modeling of the physical channel is often impractical. Relying on the flexibility and robustness of machine learning (ML) methods to tackle the dynamic nature of MC channels, the MC research community frequently refers to neural network (NN) architectures. This interdisciplinary research field encompasses various aspects, including the use of NNs to facilitate communication in MC environments, their implementation at the nanoscale, explainable approaches for NNs, and dataset generation for training. Within this survey, we provide a comprehensive analysis of fundamental perspectives on recent trends in NN architectures for MC, the feasibility of their implementation at the nanoscale, applied explainable artificial intelligence (XAI) techniques, and the accessibility of datasets along with best practices for their generation. Additionally, we offer open-source code repositories that illustrate NN-based methods to support reproducible research for key MC scenarios. Finally, we identify emerging research challenges, such as robust NN architectures, biologically integrated NN modules, and scalable training strategies.**

***Index Terms*—Machine learning, neural networks, deep learning, molecular communication, Internet of Bio-Nano Things**

J. Torres Gómez, L. Debus, O. Tugay Başaran, and F. Dressler are with the School of Electrical Engineering and Computer Science, TU Berlin, Berlin, Germany, Emails: {torres-gomez,debus,basaran,dressler}@ccs-labs.org. R. Khanzadeh, S. Angerbauer, and W. Haselmayr are with the Johannes Kepler University Linz, Austria, Email: {roya.khanzadeh,stefan.angerbauer,werner.haselmayr}@jku.at.

P. Hofmann and F. H.P. Fitzek are with the Deutsche Telekom Chair of Communication Networks, Technische Universität Dresden, Dresden, Germany; F. H.P. Fitzek is also with the Centre for Tactile Internet with Human-in-the-Loop (CeTI), Dresden, Germany, Email: {pit.hofmann,frank.fitzek}@tu-dresden.de.

B. D. Unluturk is with Michigan State University, 775 Woodlot Dr, East Lansing, MI, USA, e-mail: unluturk@msu.edu.

S. Abadal is with Universitat Politècnica de Catalunya, Spain, Email: abadal@ac.upc.edu.

S. Lotter and R. Schober are with the Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU), Erlangen, Germany, Email: {sebastian.g.lotter,robert.schober}@fau.de.

This work was supported by the German Federal Ministry of Education and Research (BMBF) through the project IoBNT, grant numbers 16KIS1986K & 16KIS1994, and the joint project 6G-life, grant number 16KISK001K. This work was also supported by the German Research Foundation (DFG) through the project NaBoCom, grant number DR 639/21-1, and as part of Germany's Excellence Strategy – EXC 2050/1 – Cluster of Excellence "Centre for Tactile Internet with Human-in-the-Loop" (CeTI). Further support has been given by the "University SAL Labs" initiative of Silicon Austria Labs (SAL) and its Austrian partner universities for applied fundamental research for electronic-based systems'

I. Introduction

Inspired by Schrödinger's thoughts about the question "What is life?" [1], the physics community joined biology to describe constituent components on the boundaries between inanimate matter and life. Today, more communities are joining the realm of biology, including electrical engineers and computer scientists for functional purposes; frameworks like the Internet of Bio-Nano Things (IoBNT) [2], [3], [4] can only be realized through truly interdisciplinary research between engineering, computer science, and life sciences. Focusing on recent advancements in artificial intelligence (AI) to realize IoBNT applications, this survey takes the journey one step further; we explore the research progress at the intersection of computer science and biology, specifically the use of neural networks (NNs) to enable molecular communication (MC) links and nanonetworks.

The constituent MC links of the IoBNT framework serve as artificial tools with promising applications in healthcare and industry but pose new challenges for practical deployment. Similarly to wireless links, communication over MC channels faces the lack of reliable connections as a consequence of different effects, though: The intricacies of the mobility of molecules in air or aqueous media,[1] the chemical interactions with surrounding reactants, or the geometry of transmitters, channels, and receivers, make it infeasible to derive analytic models in most of the practical cases.[2] As such, accurate estimators of MC channels are infeasible in practice, and

[1]See historical developments in describing the mobility of molecules in [5].

[2]In MC, "molecules" are generally the information carriers, although other classes of carriers exist, such as calcium ions and magnetic nanoparticles [6]. In this paper, we indistinctively refer to "molecule" and "nanoparticle" as information carriers.

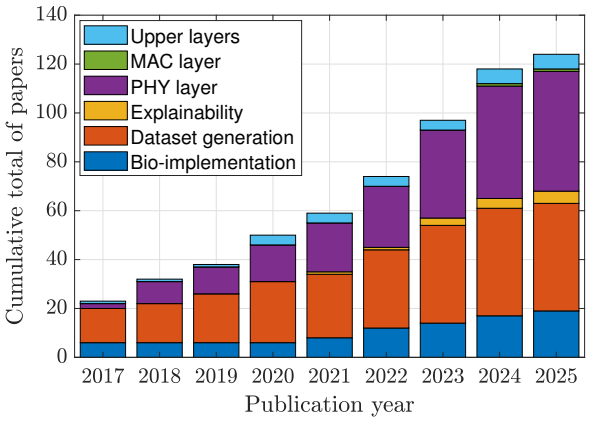


Fig. 1: Trends in contributions related to NN research for IoBNT per area and publication year. The areas refer to the communication layers, such as the upper layers, MAC layer, PHY (detectors, encoders, and synchronizers), and the bio-implementation of NN architectures.

receivers must be developed with self-learning mechanisms that account for unknown parameters. This is where machine learning (ML) plays a significant role, and specifically NN architectures as universal approximators to learn the channel characteristics along their training.

Such an approach is not entirely new in the bio realm, ML-based methods have already been explored and provide valuable inspiration for applying NNs to MC.[3] The research community is already investigating NN architectures as enablers of artificial communication mechanisms in MC channels. Examples include NNs as sequence decoders for air-based transmissions and experimental testbeds as in [10], and MC channel modeling in multiple-input multiple-output (MIMO) setups as in [11]. Furthermore, in large-scale links, i.e., transmission distances larger than a meter, NNs have also been deployed in the IoBNT framework to localize potential health conditions in the human body as in [12].

A. Prelude

In contrast to the development of intelligent wireless networks, where ML tools were layered on only after the technology had matured, by contrast, IoBNT is being engineered based on an already well-established ML foundation. Lessons learned from the communication performance in wireless links also apply to the IoBNT's physical layer (PHY), e.g., modular designs combining modulators, encoders, or channel estimation blocks are generally suboptimal, and a similar situation arises in MC links; see [13, Chap. 1]. The joint design of encoders and decoders has been shown to outperform focused, isolated blocks in MC links, as illustrated in [14].

Today, the research community contributes to the joint field of ML and IoBNT networks in various directions. As indicated in Fig. 1, a growing number of works focus on the PHY layer of IoBNT; other contributions research solutions in the MAC and higher layers, as well as dataset generation for training, the explainability of NN operation, and bio-compatible means for NN deployment. These ongoing research activities lay the groundwork for numerous new developments, as we unveil in this survey.

[3]Notably, ML has been proposed to model turbulent behavior [7] or to track particles in diffusion trajectories [8]. Furthermore, ML for the modeling of *active matter*, which includes processes such as swarming, chemotaxis, and other individual and collective phenomena, has been studied [9].

B. Literature Review Strategy

The literature review strategy to prepare this survey involved three stages: (i) Searching for the keywords "Neural Network" `AND` ("Molecular Communication" `OR` "IoBNT") in the papers' abstracts, (ii) looking for the papers referenced by the ones in stage (i) and for the papers referencing those as well, and (iii) listing papers in our curated set, which mostly reference topics on the biocompatible implementation of NNs and dataset generation. For stage (i), we conducted a comprehensive search in the leading databases IEEE Xplore, ACM Digital Library, Science Direct (Elsevier), Springer, and Wiley Online Library. In all cases, our inclusion criteria refer to: (i) The use of NNs as enablers of communication over MC links, which includes the communication design through all layers of the MC channel, i.e., from the PHY layer to the application layer, (ii) biocompatible implementation of NN models, (iii) explainable methods for NNs as applied to MC, and (iv) dataset creation. The search yielded a total of 260 papers for stage (i), and we manually added another 124 references in stages (ii) and (iii), yielding a total of 384 references. Adhering to the inclusion criteria outlined above, we filtered these references to 144 published papers, which span from 2017 to early 2025.

C. Contributions

Focusing on NN architectures, this survey aims to outline the potential of NNs in IoBNT applications from various facets: Architecture, implementation, explainable approaches, and dataset generation. Our main contributions can be summarized as follows:

- Deployment of NN architectures for MC links: We discuss in detail the NN architectures reported for the various IoBNT layers. The need to introduce NNs is clearly motivated for each layer based on the problem definitions reported in the literature. This is discussed taking into consideration various NN architectures, e.g., feedforward, recurrent, autoencoders, and convolutional. We aim to provide readers with a holistic view of problems, environments, and NN architectures in each communication layer.
- Comparative discussion of performance versus complexity of the state-of-the-art NN architectures: We thoroughly discuss the performance and complexity of the reported NN architectures as decoders in MC environments. We evaluate the decoding performance in terms of bit error rate (BER) and the complexity in terms of the number of learnable parameters. We found that recurrent architectures exhibit the highest performance while requiring the least complexity.
- Code developments for NN-based designs in MC links: We provide illustrative code examples for training and using NNs. Training and testing of the modules are based on synthetic and testbed-generated datasets in MC

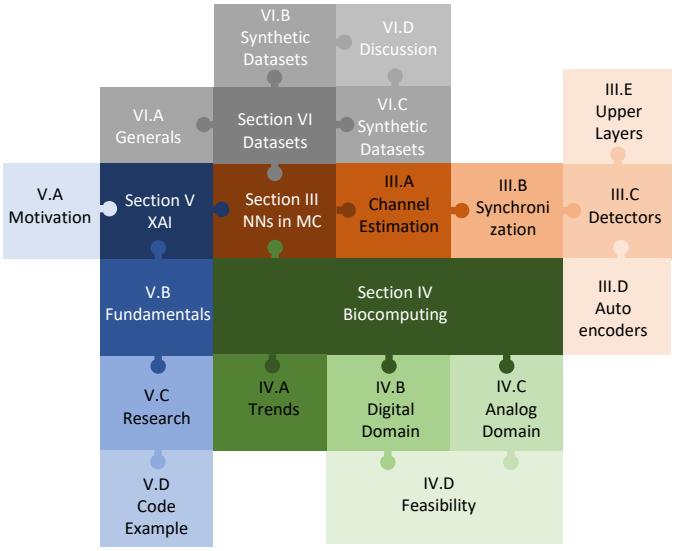


Fig. 2: Mosaic representation of the survey content.

channels. This open-access code encompasses a range of NN architectures in various MC environments, including free diffusion, flow-based, and vessel-like channels.

- Review of potential implementations at the nanoscale: We summarize state-of-the-art technologies for implementing NN architectures in the nanoscale domain. We describe bio-compatible technologies such as micro-fluidic circuits and deoxyribonucleic acid (DNA) chemical reactions, which provide means to run NN modules.
- Summary of explainable approaches to describe the operation of NNs: We introduce some of the most recent explainable methods with a focus on MC environments. In light of the physical nature of MC channels, we summarize means to interpret the operation of NNs.
- Comprehensive summary of synthetic and testbed-based generation of datasets for training NNs: We elaborate in detail on MC-related datasets for training NN modules. We also review dataset accessibility, documentation, and usability based on the published code and documentation per dataset.
- New upcoming challenges: We finally summarize new research directions related to the convergence of NN architectures and MC. We identify open problems for developing robust links, deployment, and training of NNs, as well as challenges towards explaining their operation.

D. Reader's Itinerary

This survey aims to give a detailed overview of NN concepts for MC. We continue in Section II summarizing recent developments in MC following the IoBNT framework. Within the next sections, we structured the content to guide the reader into the joint topic and help to dive deep into selected aspects as follows: (1) NNs enabling MC communication links, as the central theme in Section III; (2) Synthesis of NN modules with bio-components in Section IV; (3) Explainability approaches specific for AI in MC channels in Section V; and (4) Synthetic and testbed-generated datasets for training and testing NN modules in Section VI. The most extensive part is Section III, which examines standard communication components (supported by NNs) such as channel estimation, synchronization, detection, and autoencoder mechanisms.[4] We also list a glossary of terms used throughout this survey in the appendix for the reader's convenience.

This survey can be read in various ways to benefit the reader's expertise.[5] Fig. 2 shows a mosaic representation of the paper's content. As this figure illustrates, the paper can be read section by section, but more experienced readers can also jump directly to specific topics.

II. Most Recent Developments in Molecular Communications: A Survey-based Perspective

This section outlines the most recent developments in MC and nanonetworks, building upon related surveys published over the years 2019 to 2024. When integrating theoretical communication aspects into MC schemes, the research community is increasingly adopting AI-based approaches to overcome the complexity of theoretical concepts and practical implementations. Tracing this trend, we can group the most recent survey papers into four categories: i) IoBNT frameworks and applications, a field that also integrates MC and electromagnetics;, ii) Theoretical and technical developments in MC, which links communication theory with MC schemes; iii) Cross-disciplinary approaches between the life sciences and communication-engineering communities, which discuss the much-needed integration between the two communities, and iv) Recent AI innovations in the MC field as summarized within the first surveyed materials on the topic.

1) IoBNT/MC Frameworks and Applications: Grounded in applications for the early detection and treatment of diseases in the human body, surveys here elaborated on the IoBNT architecture to fulfill a futuristic paradigm: To connect human cells to the internet, as portrayed in [4]. The IoBNT architecture comprises a nanonetwork within the human body, where nano-biological functional devices are the main components, e.g., engineered bacteria, human cells, nano biosensors, that perform sensing and actuating tasks in the MC domain [16]. The nanonetwork is connected to wearables attached to the skin surface, which are then linked to healthcare providers via the internet; see an illustration in [17, Figures 7 and 9]. These conceptual developments and early-stage technologies assess IoBNT as a part of 6G beyond networks, see [18, Sec. XI], and embody a symbiotic relationship with the collective intelligence of the body [19].

Towards this vision, surveys have introduced a variety of interfaces between nanodevices and the internet through heterogeneous links in the biological [20], [21], electrical, acoustic, and electromagnetic domains; see [22, Sec. VII], [23, Sec. III], [24, Sec. 4], and [16], [25]. Such interfaces enable

[4]We provide the code on two platforms: (i) In the Ocean Code platform, we provide access to the cell-to-cell example developed in Section III-A4 under the link https://codeocean.com/capsule/6777864/tree/v1, and (ii) in the GitHub platform we provide access to all code in this paper related to synchronization, decoding, and autoencoding under the link https://github.com/tkn-tub/NN_molecular_communications. Furthermore, we provide the database for training and testing the reported NN modules in [15].

[5]Here, we invite the reader to follow the collage model for reading this survey, as first proposed by Julio Cortázar within the novel *Rayuela*.

groundbreaking healthcare applications, such as reducing the detection time for bacterial infections, as illustrated in [4, Sec. V] and discussed in [17], [20], [25]. Applications also span smart agriculture and environmental monitoring to track the health status of animals and plants, as summarized in [16], [26], [27]. In industrial environments, nanodevices can be deployed inside underground pipes to detect corrosion and damage [26, Sec. III.b].

In the electromagnetic domain, surveys have addressed analog front-end units in the terahertz (THz) band as the interface between implanted nanosensors and the outside world, see [22]. Due to the plasmonic effect observed in graphene materials, the distinctive relevance of this particular frequency band resides in the existence of signal generators (see [28]) and antennas (see [29], [30]) that can be miniaturized to a few microns long and wide. Different from the THz band, recent surveys also pointed to optics with nano-lasers and nano-antennas, MHz system frequencies with magnetoelectric antennas based on nano-electromechanical devices, or even to non-radiative techniques, either coupling-based or wave-based, such as galvanic coupling or ultrasound [31].

In particular, some surveys have focused on technological advances in signal generation and antennas, which become enablers of this approach [29], [31], while other survey materials focused on protocol and system design aspects [32]. Furthermore, several works have summarized potential healthcare-related applications of THz networks [29], [33], including (i) ultra-precise detection and localization of diseases with nanomachines flowing in the human blood system [32], [34], and (ii) brain-computer interfaces enabled by a wireless interaction with the human brain [35] even down to single neurons [36]. Notably, however, none of the existing works on the electromagnetic side of nanonetworks point to AI/ML as a possible tool to aid in the modeling of communication or the design of protocols.

Recent surveys on IoBNT also highlight the self-power capabilities of nanodevices in maintaining the nanonetwork in continuous operation. Micro-batteries built with micro-electromechanical and nano-electromechanical devices are powered by energy harvested from the environment or through wireless power transfer, as described in [16], [23]. Besides, due to the highly sensitive applications of IoBNT networks, especially in healthcare, surveys also remark on the challenges and risks of MC [37] and wireless channels, see [23, Sec. IV]. Likely attacks include eavesdropping, spoofing attacks, and jamming, see a full description in [23, Table 6]. These attacks exploit the limited computational capacities of nano nodes to implement sophisticated protection algorithms. Mitigating strategies use fingerprinting based on channel identification (a domain well-suited for AI) or cryptographic mechanisms to avoid eavesdropping.

2) Theoretical and Technical Developments Regarding MC Channel Models: MC channel models are a widespread research topic in the scientific community across different scales: (i) Human vessels, organs, and tissues on a larger scale; (ii) on the cellular scale, cell-to-cell communication and information processing; and (iii) on the molecular scale-free diffusion or junction MC channels between cells; see [20]. These theoretical models underpin the feasibility of applications such as drug delivery, disease monitoring (e.g., cancer initiation and progression), and the deployment of body network infrastructures [26].

Theoretical developments in [38] summarize the end-to-end channel impulse response (CIR) derivation for various MC geometries at the molecular scale. MC channels include free diffusion, advection, and chemical degradation channels, see [38, Table 1]. At the tissue scale, mobility models are summarized in [39] for passive and active bionanomachines that carry information. Passive mobility models include random walks in a fluid and with external forces, whereas active mobility models exhibit a bias towards a preferred direction, as seen in the case of chemotaxis.[6] Besides, theoretical works also target more functional problems in MC links. Examples include methods for channel parameter estimation [42], modulation and demodulation [43], [44], coding [45], and the formulation of spatial domain resources through MIMO schemes as in [46]. These works illustrate the capabilities of actual communication schemes at the nanoscale.

Not only does theoretical work drive the study of MC links, but also the development of simulators; see [38, Sec. V.A], [26, Sec. II.5]). Surveys also illustrate experimental testbeds as tracks of technological readiness in the near future, see [47], [48], [49], [38, Sec. V.B], [22, IV B.2], and [20, Sec. IV]. This research field focuses on interfaces between the biological and electrical domains, such as synaptic links in [50], optical links in [51], electrical domain connections in [52], and radio frequency (RF) signals in the THz domain, as described in [22].

3) Recent AI Innovations in the MC Field: A prevalent viewpoint expressed in some MC-related surveys is that data-driven detectors may overcome the limitations of model-based ones, which prevent them from performing optimally in real-world MC channels, as seen in [18], [53], [54], and [38, Sec. V.B.2.b].[7] Advancing the field, the community is beginning to embed feedforward NN models within DNA circuits in a cell, as summarized in [55], [56], [57], and remark on researching explainable methods for the operation of NN modules, as indicated in [53]. However, only a few surveys, such as [53], [58], [59] address ML methods suitable for developing MC-based schemes. The work in [53] mostly focuses on a performance comparison between model and data-driven detectors, the work in [58] primarily centers on the application layer, while the work in [59] succinctly summarizes biocompatible technologies for running NN modules.

4) Cross-Disciplinary Routes: Towards the interdisciplinary understanding among researchers in communication engineering, synthetic biology, and bioengineering, a hierarchy is proposed in [60] to map communication concepts to the biological behavior of cells. Unlike biologists, who study natural system interactions guided by holistic views, the engineering community is developing modular designs, i.e.,

[6]The chemotaxis process describes the mobility of bacteria towards the higher concentration of attractant molecules, see models in [40], [41].

[7]See also [54] for a detailed explanation of ML architectures and their application in other research areas such as therapy development and nanomaterial designs.

defining communication functionalities by layers as application, data abstraction, molecules-data interface, the interaction of molecules, and their propagation. A second perspective toward interdisciplinary research lies in the most recent application of information semantic concepts, as presented in [61]. MC-engineered systems, with a goal-oriented approach, may support the design of biochemical applications, as goals can be mapped to biological processes.

5) Summary Remarks: The variety of topics in the above-related surveys introduces a joint appeal: Scenario awareness must be part of MC deployments and dynamically adapt to real-world environments; see, for instance, the channel dependency of adaptive threshold detectors in [44, Eq. (17)], adaptive coding strategies to reduce inter-symbol interference (ISI) as in [45, Sec. IV.C], and the challenges to design high data-rate transmissions due to the non-stationary nature of MC noise, as discussed in [53]. A recent trend is that NN architectures are becoming relevant for applications in channel estimation, synchronization, detection, and related solutions such as localization and disease detection, as illustrated in Fig. 1. NNs are primarily implemented external to the MC environment, as we discuss in Section III, although embedding the NNs within the MC pipeline is very appealing, as we consider later in Section IV. In the following sections, we provide an overview of the literature groundwork on NN architectures, MC environments, and comparative performance.

III. Neural Networks as Enablers of IoBNT Networks

In the context of IoBNT, where communication and computation are deeply embedded into MC environments, this section details the reported NN-based solutions for the various network layers. We thoroughly discuss reported NN architectures, deployments, and results for the PHY and upper layers. As we will encounter in this section, early studies primarily focus on the PHY layer. Although these studies do not directly mention their application to IoBNT networks, they can be easily adapted as techniques for facilitating node connectivity within IoBNT. Within the PHY layer, we refer to the MC model illustrated in Fig. 3, where the MC model is primarily divided into three blocks for their separate study: transmitter, physical channel, and receiver. This section is organized into the following modules: Channel estimation, synchronization, and detection (included within the Receiver block in Fig. 3), as well as solutions in the MAC and upper layers. Each subsection follows the same structure: (i) Problem description for MC channels, where the demand of NN modules is discussed, followed by (ii) the MC environments where NNs have been deployed, (iii) description of reported NN architectures and their performance, (iv) illustrative examples of implemented code using NN modules, and (v) concluding remarks.

A. MC Channel Estimation

End-to-end MC channel estimators are essential for optimally tuning the decoding tasks at nodes. In the literature, we find the application of NN-based estimators for various purposes, such as evaluating communication performance [62], designing

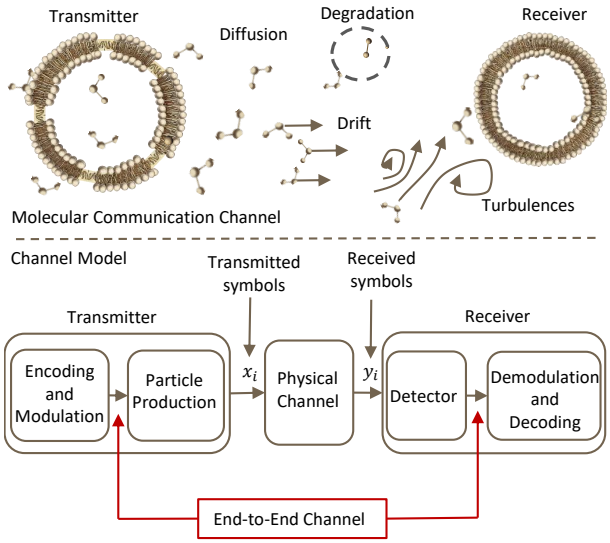


Fig. 3: Representation of a point-to-point MC link.

macro-scale receivers [63], and drug delivery for healthcare applications, as discussed in [64]. In this section, we categorize these reported estimators as solutions that contribute to parameter estimation and modeling of MC channels separately.

1) Problem Definition: The CIR in MC links is defined as the probability of finding a molecule at the receiver after its release from the emitter; see [38, Def. 2]. It is given as a function of time, here denoted as $h(t)$ with given parameters $\{h_0, h_1, \ldots, h_n\}$. Channel estimation and modeling are the two main problems, referring to finding the set of parameters $\{h_i\}$ and the sequence $h(t)$ itself, respectively. For instance, in the case of free-diffusion channels, where the transmitter node is modeled as a point and the receiver node as an absorving sphere, the CIR is given as $h(t) = \frac{V_{\text{rx}}}{(4\pi D t)^{\frac{3}{2}}} e^{-\frac{d}{4Dt}}$ [65, Eq. (6)], where D is the diffusion coefficient of the particles, t denotes time, d is the communication distance, and V_{rx} is the volume of the receiver. In this case, $h(t)$ is known, but parameters such as the distance may be unknown in practical scenarios, resulting in a channel estimation problem.

Besides, in many situations, evaluating the CIR sequence is not feasible due to the complexity of the MC channel's geometry and topology. Examples include the channel illustrated in Fig. 4, where a reflector introduces a non-linearity in the channel, preventing the development of a closed-form expression for $h(t)$. In that case, the CIR itself must be estimated, which refers to the channel modeling problem.

Either of these two problems unfolds into a challenge due to the unknown variables that need to be estimated, such as the communication distance, the impact of the geometry of the emitters and receivers, or the fluid velocity in vessel-like media. Due to the difficulty of these problems, literature reports NN-based solutions leveraging on their capacity as universal approximators. NNs can be used either to estimate the parameters for a given model or as a model-independent method to estimate the CIR. In the following subsections, we detail various MC scenarios and the reported NN-based estimators.

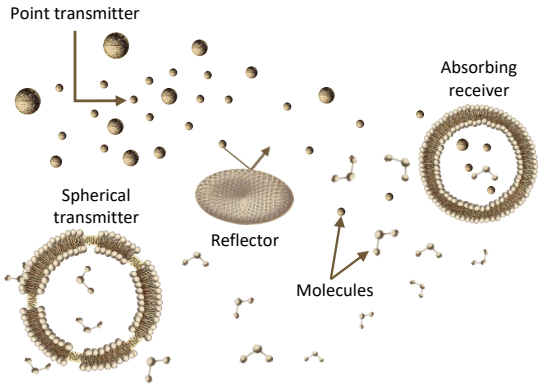


Fig. 4: Conceptual illustration of a free-diffusion channel for a complex topology. This environment comprises absorbers, reflectors, and multiple emitters.

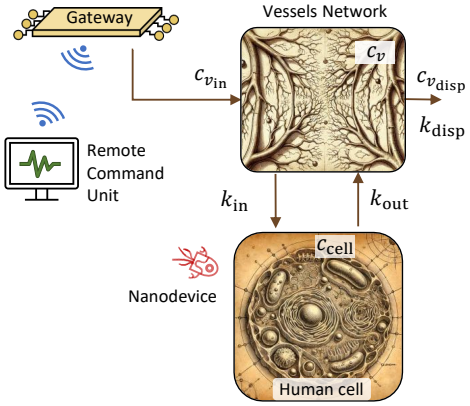


Fig. 5: Two-compartmental model of the cardiovascular system from [64]. Parameters k_{in}, k_{out}, and k_{disp} denote molecule rates following the compartmental model. Variables $c_{v_{\text{in}}}$ and $c_{v_{\text{disp}}}$ represent the concentration of molecules at the input and output of the corresponding compartment, while c_v and c_{cell} are those within the compartment.

2) Environments for MC Channel Estimation: In the literature, NN models for channel estimation and modeling have been deployed in two main MC environments: In simulated free-diffusion MC channels as in [62], [66], [67], [68], [69], [70], [71], [72], within the network of the human circulatory system (HCS) as in [64], and in realworld testbeds for open air channels, as in [63].

Free-diffusion environments: The more commonplace environment of free diffusion poses challenges in evaluating the channel model for complex topologies. Examples include scenarios where multiple senders, reflectors, and absorbers are situated between the emitter and the receiver, which hinder the analytical development of closed-form solutions for the CIR; see Fig. 4. The NN-related literature considers free-diffusion environments on the (μm) scale, which include spherical reflectors and absorbers placed at arbitrary locations between a point emitter and a spherical absorbing receiver. This is the case reported in [67], [71] in 2D and also extended to 3D in [68]. A less complex topology is considered in [62] in a 3D environment, including a perfect spherical receiver and point and spherical transmitters. MIMO links in MC are also gaining increasing interest because of their potential to enhance transmission rates. Examples are the 2×2 MIMO scheme in free-diffusion MC channels for [66], [70], 4×4 MIMO in [72], and asymmetric MIMO channels comprising 2 emitters and 4 receivers in [69]. These MIMO channels lay out the geometries of point transmitters alongside spherical absorbing receivers.

Literature also evaluated testbed measurements accounting for realistic channel effects, which include the various sizes of droplets, evaporation, and unsteady flows as in [63]. The testbed deploys an air compressor sprayer to release ethyl alcohol molecules and an MQ-3 alcohol sensor to detect the droplets from distances of up to $200\,\text{cm}$. See a summary of MC environments and their parameters in appendix, Table II, specifically for the entries related to channel estimation.

Human vessel networks: For drug delivery applications within the human body, the model reported in [64] includes three main components. As depicted in Fig. 5, these components are: 1) A photo-triggered transducer interface (Gateway), which is placed at the skin surface, 2) the human vessels network as the information channel, and 3) a network of nanodevices located at the target cell.[8] The photo-triggered interface is attached to the skin surface and injects specific molecules into the human circulatory system. This release mechanism is activated upon the wireless reception of a command from a remote unit, which the care providers can directly access. The released molecules travel through the vessel segments to the target cell, where a nanonetwork is activated to release the corresponding drugs upon detecting this particular molecule. Molecule concentration levels are evaluated using a two-compartmental model, one for the circulatory network and another for the target cell. The evaluation of the concentration level is based on a system of two differential equations, which incorporate the parameters k_{in}, k_{out}, and k_{disp}; refer to [73, Eqs. (14) and (15)]. These parameters refer to the rates of molecules flowing in and out from the compartment, as represented in Fig. 5.

3) NN Models for MC Channel Estimation: Various models are reported for free-diffusion channels aiming to estimate CIR parameters, as in [63], [66], [69], [70], [71], and more broadly, to estimate the CIR sequence as in [67], [68], [72]. These models implement the architectures feed forward NN in [62], [63], [64], [66], [69], [70], [72], convolutional neural network (CNN) in [67], and the renowned Transformer in [68], [69], [71].[9] The NN weights and bias coefficients are evaluated using the backpropagation algorithm, as in [62], [66], [70], and the Bayesian regularization approach, as in [62], [63], [66], which implements the Levenberg-Marquardt optimization algorithm [74], avoiding overfitting.

NN-based solutions for MC channel estimation: The aforementioned NN architectures are used to estimate a variety of MC channel parameters, including distances and the orientation of emitters and receivers in the channel. For instance, a feedforward NN model is developed in [63] to estimate the distance between the emitter and the receiver. The model employs a single hidden layer with a single node and is trained on a predefined set of features, which includes the peak of the

[8]The sketches for the bio-components in Fig. 5 do not mean to be accurate but are for concept illustration purposes only. The figures were generated with the assistance of ChatGPT, utilizing a Da Vinci style for the drawing. Similarly, we generated all illustrations of bio-components within this paper.

[9]Further details for the Transformer architecture are provided in Section C.

received sequence and the rise time from low to high amplitude in the sequence of received molecules. Results exhibit relative errors between the estimated and actual distances in the range of 2 to $20\,\%$ at communication distances of 100 to $200\,\mathrm{cm}$.

NNs are also reported in [62], [66] to evaluate the parameters of the first-hitting time probability; see [66, Eq. (2)]. The deployment consists of single and two NN machines, trained with the recorded number of molecules at the receiver. The single NN is deployed through an fully connected (FC) hidden layer and a total of 30 nodes. The two-machine model is proposed in [66], where each NN estimates a separate set of the CIR parameters. Both NNs are implemented with a single FC layer and 15 nodes each. The results in [66] indicate that the single-machine model performs better than the two-machine model. The models achieve high accuracy with an absolute error of less than 10^{-2}.

In free diffusion MIMO environments, NNs are trained to directly estimate the parameters of a given closed-form expression for the CIR, as indicated in [66], [69], [70]; see the model and the parameters in [66, Eqs. (2) and (3)]. These solutions [66], [70] follow a two-stage approach to train the NN. In the first stage, the authors fit a CIR model to a 2×2 MIMO MC channel, utilizing the non-linear least square method and with samples generated through simulations. Next, the NN is trained to predict the fitted CIR, utilizing as inputs MC channel parameters such as distance, the molecules' diffusion coefficient, and receiver radius. The trained NN is used later to predict the BER performance from the MC channel parameters.

Continuing with MIMO channels, a more typical deployment trains NN models using received molecule counts, as investigated in [69], [72]. The work in [69] comparatively analyzes the performance of Transformer (see appendix, Section C) and feedforward NN architectures for channel estimation, where the latter yields better performance. The feedforward NN comprises one FC layer and 5 hidden layers of 10, 20, 40, 20, and 10 neurons. The encoder and decoder components of the Transformer implement two FC layers and the same number of hidden layers as the feedforward NN. The inputs for both architectures are the number of received molecules, and the outputs are the parameters of the theoretical CIR expression. Particularly, the work in [72] estimates the distance and the rotation angle between the emitter and the receiver planes of a 4×4 MIMO setup, where the rotation refers to the missalignment of the emitters and receivers location in the range of 0 to $45°$, see [72, Fig. 2]. The solution deploys a feedforward NN of 14 connected layers, where the inputs are the numbers of molecules and the outputs are the coefficients.

Finally, in the environment of the human vessels network (see Fig. 5, [64]), feedforward NN models are reported to estimate the set of parameters $\{k_{\mathrm{in}},\, k_{\mathrm{out}},\, k_{\mathrm{disp}}\}$ for the compartmental scheme, see Fig. 5. The model is trained with the concentration of drugs at the nanodevice location, achieving an estimation error of less than $30\,\%$.

NN-based solutions for MC channel modeling: NN models are also trained to predict the CIR sequence in free diffusion channels as researched in [67], [68]. The solution develops two branches, one of which takes as input the RGB image of the MC environment. Point transmitters in the MC channel are depicted with dots, while absorbers, reflectors, and receivers are depicted as circles of different colors and radii. A CNN is used in [67], and a Transformer (without the positional encoders) is used in [68] to extract the relevant image features. A multilayer perceptron (MLP) block is included within the architecture to normalize the sequence for the CIR. The second branch of the design in [67] deploys an FC layer and an MLP block to estimate the maximum amplitude of the number of received molecules. The inputs to the FC layer are the diffusion coefficient, the sampling time, and the spatial granularity. In this two-branch design, the normalized mean square error (MSE) between the ground truth and the estimated CIR sequence has an order of magnitude of 10^{-2}.

4) Illustrative Code Example to Estimate the Distance Among Cells: This code example develops a distance estimator between immune and cancer cells based on the recorded number of vesicle molecules.[10] Cancer and immune cells exchange vesicles in the proximity of each other, and the concentration level of vesicles recorded at the immune cell can be readily used to estimate their distance to the cancer cell; see the sketch in Fig. 6 and model details in [75].[11] However, the concentration level's dependence on the distance parameter hinders the derivation of a closed-form expression for the inverse relation, i.e., distance versus concentration of vesicles.[12] Resolving these intricate dependencies, a trained NN can model the relation between the number of induced vesicle releases by the immune cell and its distance to the tumor cell.

To model the environment, we utilize the code provided by the authors in [75], which calculates the number of released vesicles by the immune cell; see the various curves plotted over time and for different distances in Fig. 6(a). We first extract two features from the raw data to train the NN: (i) The peak amplitude of the slope of the received vesicles; and (ii) the time location of the peak; see the corresponding slope curves in the embedded plot in Fig. 6(a), and the two features in Fig. 6(b). We implemented in Matlab a feedforward NN comprised of a single-layer and two nodes, using the hyperparameters listed in appendix, Table IV.[13] As depicted in Fig. 6(c), the NN accurately estimates the distance to the tumor cell in the range of 2 to $10\,\mu\mathrm{m}$ with a relative error of $3.3\,\%$. In medical applications, the estimated distance can trigger the release of drugs from engineered immune cells when they are close enough for effective treatment; refer to the proposed methodologies in [39, Section V].

5) Concluding Remarks: The use of NNs for MC channel estimation has mostly focused on feedforward NN and CNN architectures. Feedforward NNs are used to estimate specific parameters in the channel, and CNN architectures are adopted for the more ambitious goal of estimating the CIR. This subject still requires further development of NNs to target more realistic scenarios. Particularly in IoBNT contexts, both short-range channels, such as cell-to-cell, and long-range channels, like

[10] See further details on cell-to-cell communications in appendix, Section H.

[11] We provide access to the evaluation of the number of vesicles based on the distance and time in the dataset in [15].

[12] See the position-related terms r^D and r^T in [75, Eqs. (21) to (24)].

[13] We provide the distance estimator code at https://github.com/tkn-tub/NN_molecular_communications/tree/main/Section_III_A_distance_estimator.

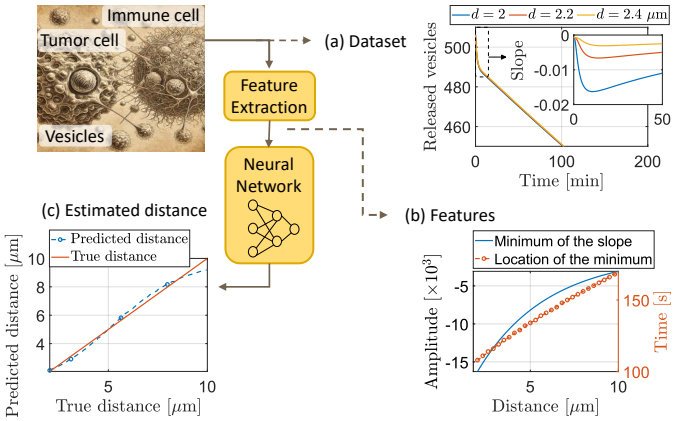


Fig. 6: Overview of the schematic and findings for calculating the distance between cells using a feedforward NN as an estimator.

those through the human circulatory system, are lacking from modeling with NNs.

B. Synchronization

Successful communication among nodes relies on all participants being aware of the timing of the other entities involved; they have to be synchronized. In many works related to MC, synchronization is assumed to exist by default, allowing for the separate study of receiver sensitivity to noise or the optimization of detection parameters. Yet, in real-world systems, the communicating nodes must explicitly synchronize with each other to achieve successful communication. The following sections will define the problem, summarize reported approaches, and describe an example implementation.

1) Problem Definition for Synchronization: The problem of symbol synchronization concerns the question of when a symbol is considered to start at the different communication participants. In the simplest case, i.e., one transmitter and one receiver, the transmitter sends out the symbols, and according to the delay produced by the MC channel, they arrive on the receiver side. It is the receiver's challenge to determine the start of the transmitted symbol.

Traditionally, symbol synchronization in MC is often addressed using the maximum likelihood estimation (MLE) as an optimal method; see [76]. Due to the high computational requirements for implementing MLE, it is often not a viable option for implementation in experimental MC systems. Out of this need, low-complexity versions of MLE that rely on the use of separate signaling molecules for data transmission and synchronization were proposed in [76]. Additionally, based on the use of two different signaling molecule types, the solution in [77] creates an artificial synchronization clock by exploiting the differences in diffusion coefficients. Due to their higher diffusion coefficient, the synchronization signaling molecules arrive at the receiver first and signal the start of a transmission for the slower data signaling molecules. A blind synchronization approach, based on MLE, was proposed in [78], where a predefined sequence of molecules is transmitted beforehand for MLE calibration.

In most synchronization approaches reported above, one problem is ignored: The variability of the MC setup. The system is assumed to be static and well-known. However, this assumption does not hold for real-world scenarios where transmitter-receiver distances, background flow, and the behavior of signaling molecules change over time. This variability triggers the need for synchronization to enable adaptation to the system's changing state. These problems, often untackled in the current literature, are well-suited for ML-based synchronizers because they can adapt continually to the fast-changing MC channel.

2) Relevant Environments for Synchronization: The reported ML-based solutions to synchronization constitute a first step towards including more complex environments in their setup and evaluating two different types of dynamic transmission scenarios. In the reported implementations, two extensions to the simple static MC channel were evaluated:

Mobile Setup: In many expected MC applications, the communication participants move around a more or less defined space and must be able to communicate with each other from different positions. The synchronization strategy must, therefore, be able to adapt to a changing number of molecules being received over time as a result of the varying CIR.

Relay Setup: Depending on the environment, the senders may need to extend each other's communication ranges by relaying messages. In this case, a single receiver will likely detect molecules sent for transmission by the last relay, together with the interference from previous relays. Combined with the possible mobility of communication participants, a synchronizer must be able to synchronize reliably in dynamically changing interference scenarios.

Still, these scenarios do not fully describe all expected application environments of MC. Synchronization strategies for MC must additionally consider more realistic setups like multiple transmitters and receivers or multi-path propagation. Here, again, the synchronization method must be able to adjust to the resulting dynamic environments, and ML could be highly beneficial.

3) NN Models for Synchronization: As synchronization in a dynamic MC setup relies on continuous adaptation to an ever-changing environment, reinforcement learning (RL), which is based on interaction with the environment, is especially fitting. With its ability to estimate complex mathematical models [79], RL offers the possibility of solving the synchronization problem in MC with little a priori knowledge.

Synchronization with the help of ML was first approached in [80]. An RL agent was designed for a simulated mobile MC setup of an air-based MC testbed [81]. Rewarded on the correctness of the decoded bit, the agent learned to adjust the decoding threshold to detect the synchronization sequence `[11001]`. While the agent showed potential, the necessity of knowing the correct bit value during training significantly impacted the applicability of the approach in a real-world setting. The synchronizer has also been integrated into a relayed mobile setup consisting of a transmitter, a relay, and a receiver, as described in [82]. The new RL agent's reward was based on the difference between the counted number of molecules and the threshold set by the agent. The synchronizer was reported to achieve a true positive rate (TPR) of over $80\,\%$ and a false positive rate (FPR) of

below 5 % for all transmitted synchronization frames. The results compared particularly well to the filter-based maximum likelihood estimation (FBMLE) synchronizer [76], which struggled to cope with the additional interference caused by the original transmission on the link between the relay and the receiver. In both RL-based synchronizers, a proximal policy optimization (PPO) agent was used. Their actor and critic networks included long short-term memory (LSTM) layers to enable the consideration of past observations and actions in the evaluation of the current step.

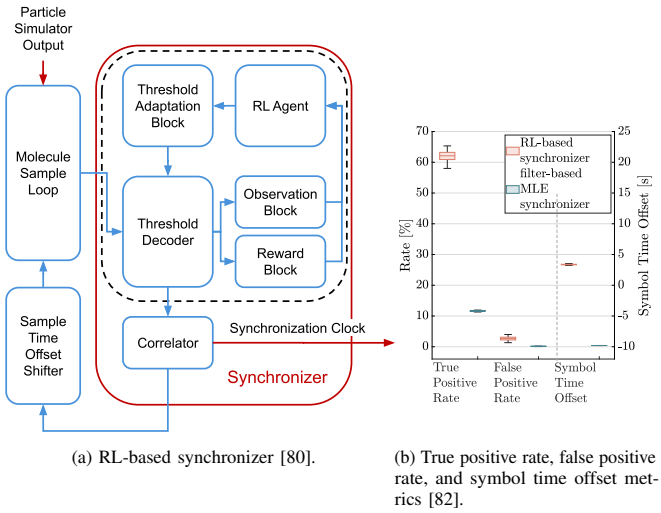
Synchronization and detection via deep neural networks (DNNs) is implemented in [83]. Two setups, featuring a one-dimensional CNN and a gated recurrent unit (GRU) recurrent neural network (RNN), are evaluated for their synchronization and decoding performance, using a padded Barker code as a synchronization frame. The first setup uses one NN for both synchronization and decoding, and the second uses separate NNs for the two tasks. In the evaluation, different static signal-to-noise ratio (SNR) scenarios are considered in an unbounded free-diffusion channel. Both setups successfully synchronize the transmissions at SNRs of 45 dB or higher.

While the reported results demonstrate the usability of ML to synchronize MC systems, they only scratch the surface of the approach's advantages. Especially in more complex environments that might include multi-path and multiple transmissions simultaneously, significant benefits are expected from employing an inherently reactive ML-based synchronizer.

4) Illustrative Code Example to Synchronize the Receiver and Emitter Symbol Time: Following the example in [82], we implemented an RL-based synchronizer in Matlab and Simulink.[14] For our implementation, we utilized a particle simulation of the media modulation (MM) testbed introduced in [84], demonstrating that the reported approach can be transferred to liquid-based closed-loop MC scenarios. In the testbed, the traditional transmitter and receiver structure of an MC testbed is extended by adding an eraser positioned in the loop after the receiver and also before the transmitter. Switchable signaling molecules [85] are used for communication in this setup. The transmitter can turn the molecules "on"; after they pass the receiver, the eraser turns them "off" again.

Our synchronizer follows the structure displayed in Fig. 7a. A loop through the environment consists of: (i) Sampling the current number of molecules (Molecule Sample Loop), (ii) decoding the current bit value (Threshold Decoder), (iii) having the RL agent adapt the threshold according to the observed state and the reward, and (iv) synchronizing the sampling offset if the synchronization frame was detected (Correlator and Sample Time Offset Shifter blocks). The presented system uses a PPO agent [86] with a single LSTM layer in both the actor and critic networks. We evaluated several different network and layer sizes and found that the agent performed best with 128 cells per LSTM layer. Additionally, we performed hyperparameter tuning for the actor and critic learning rates, mini-batch size, experience horizon, entropy loss rate, discount factor, and reward scaling factor, as described in [87]. For the other parameters, we found that their influence on the agent's performance was best left at the default parameter settings.

[14]We provide the synchronizer code at https://github.com/tkn-tub/NN_molecular_communications/tree/main/Section_III_B_synchronizer.



(a) RL-based synchronizer [80].

(b) True positive rate, false positive rate, and symbol time offset metrics [82].

Fig. 7: System model of the RL-based synchronizer and its performance.

We produce data for our implementation with a particle simulator of the testbed. To vary the repeating data slightly while not distorting the simulated channel behavior too much, additive white Gaussian noise was added to create an SNR of 30 dB, as described in [43]. The RL-based synchronizer was then trained with a time series dataset for 245 000 5-bit-frames in 35 000 episodes.

We evaluate our synchronizer by comparing it to the FBMLE synchronizer [76]. Both synchronizers ran 100 times 1000 5-bit-frames taken from the evaluation dataset. To judge the synchronizer's accuracy in detecting the transmitted synchronization frame `[11001]`, we evaluate the TPR, the FPR, and the symbol time offset with regard to the synchronization frames for both synchronizers. The results in Fig. 7b reveal that the RL-based synchronizer achieves a higher TPR than the FBMLE synchronizer. At the same time, it also detects more synchronization frames wrong, which causes it to have a higher FPR than the FBMLE approach. Taking both detection rates together, the RL-based synchronizer performs better. Regarding the absolute value of the symbol time offset, both synchronizers accurately detect the symbol's start.

5) Concluding Remarks: The reported solutions show the potential of using ML-based approaches for synchronization in MC. Their ability to adapt to the highly dynamic environments of MC channels allows smart synchronizers to outperform other synchronization approaches in real-world settings. While existing solutions employ only RL approaches and RNN structures, research integrating different network structures and attention models is rife with opportunities. As we integrate MC into the HCS for precision medicine applications, ML-based approaches offer the next step in synchronizing the workflow of different parts of the application within the ever-changing environment of the human body.

C. Detectors

This section examines NN architectures used for information decoding at the receiver nodes, which is the most studied

topic in the literature. A key benefit of NNs is their ability as universal approximators, enabling adaptable tuning of decoder parameters in unknown environments. In the form of Occam's razor, NN models are conceived to operate by learning the minimum possible MC parameters of the CIR. Architectures using feedforward NN, RNN, and CNN, as well as incorporating attention models, are combined to learn from past and future samples when decoding the current symbol. In the following, we provide details on the detection problem definition, environments, deployed NN architectures, and an illustrative code example.

1) Problem Definition to Decode Incoming Symbols: Decoding digital transmissions directly refers to a detection problem, where each transmitted symbol is identified in comparison to a given alphabet of possible symbols. Introducing our notation, we represent the transmitted symbols as $\{x_i\}$, as illustrated in Fig. 3. These symbols are distorted and contaminated by noise within the physical channel, yielding $\{y_i\}$. Decoding aims to retrieve the original transmission x_i from the observed data y_i.[15]

Similarly to wireless communications, the problem of decoding is formulated based on a probabilistic description of the MC channel. MC channels are abstracted with the conditional probability $P_{\mathrm{ch}}(y_k|x;H)$, where H refers to the set of channel parameters. Having a known probability density function for the transmitted symbols, denoted as $P_X(x)$, the optimal estimation of the emitted symbol follows the maximum a posteriori (MAP) principle as [10, Eq. (2)]

$$x_k = \underset{x}{\operatorname{argmax}}\, P_{\mathrm{ch}}(y_k|x;H) \times P_X(x), \tag{1}$$

which requires the a priori knowledge of both the end-to-end channel model, i.e., $P_{\mathrm{ch}}(y_k|x;H)$, and the set of parameters H to decode sequence y_k.

Solving Eq. (1) is typically infeasible due to the unknown distribution of $P_{\mathrm{ch}}(y_k|x;H)$ and its parameters H. For instance, time-variant channels, such as those found in drift fluidic environments of human vessels, make it challenging to conceive an estimator based on the formulation in (1). Furthermore, the problem becomes more complex in experimental testbeds, as the end-to-end channel model also encompasses the particular geometry of emitter and receiver, which is infeasible to detail in practice. Tackling these impediments, data-driven methods using NN models are reported to learn the channel probability distribution, thereby sidestepping the prior knowledge of the end-to-end MC channel model as a requisite.

Decoding may also require determining the optimal detection thresholds (λ_i) for concentration shift keying (CSK) transmissions, which is a second problem formulation stated in the literature; see [88, Eq. (26)]. For optimal performance, the λ_i's must be adjusted to the MC channel parameters, such as the distance between emitter and receiver, and the emission pattern. Evaluating all these parameters is not a trivial problem, and a closed-form expression of the optimal λ_i does not exist, see, for instance, the formulation in [88, Eq. (13)] for the simplest case of a zero-bit memory receiver. Following the threshold-based detection mode, this second problem formulation is stated to find the optimal set of threshold values to maximize performance, e.g., minimize the BER metric, as follows (see [88, Eq. (26)])

$$\lambda_i = \underset{\lambda}{\operatorname{argmin}}\, \mathrm{BER}, \tag{2}$$

which is also solved through NN models due to the lack of suitable closed-form expressions.

2) Environments for NN Detectors: NN models as detectors are trained in MC free diffusion environments, and the literature introduces multiple instances of these geometries. The training sequence is constructed in MC channels where the emitter follows a point transmitter and the receiver a spherical absorbing geometry as in [89], [90], [91] or a ligand receptor setup as in [92], [93]. More complex environments follow multi-hop links as in [94], 5×5 MIMO channels with ligand receptors in [93], and a 8×1 multiple-input single-output (MISO) channel as described in [95]. The mobility of the transmitters and receivers is also modeled in [96], [97], [98], [99] as a component that shapes the receiver sequence.

The randomness of the MC channels is primarily modeled using the Poisson distribution, as in [10], [88], [89], [96], [97], [100], [101], and it is less frequently modeled with the Gaussian distribution as in [95], [102]. More realistic MC models also incorporate the degradation of molecules due to chemical reactions, as evaluated in [90], [96], [97].

Training sequences are also recorded from experimental testbeds in vessel-like channels where binary emissions are performed with acid and base signals in water in [10], [11], [102], [103], [104], [105], and using magnetic nanoparticles in [106], [107]. Pulses for binary transmissions are also shaped using E. Coli bacteria as generators, as reported in the *in vivo* testbed in [108], [109]. E. Coli bacteria are stimulated with light to release protons and increase the medium's pH level. A pH sensor is used to record the pH level, serving as the receiver [108]. As another example, binary transmission is also performed with the release of flagellated bacteria, which are guided by a magnetic field towards a spherical receiver, as described in [109]. The receiver, which was previously filled with luminescent non-motile bacteria, starts to collect the quorum sensing (QS) molecules released by the arriving ones and produce light in response. The observed intensity and the number of collected bacteria are used to decode the transmitted sequence later.

To summarize, the communication methods outlined above utilize pulse-based modulation techniques, i.e., on-off keying (OOK); refer to the Modulation column in appendix, Table II. The one bits are encoded with the number of released molecules or bacteria, and the zero bits are encoded by their absence. Additionally, emissions occur with bit durations in the millisecond (ms) range and the channel noise varies significantly from a considerable level (SNR $= 0\,\mathrm{dB}$) to a nearly negligible one (SNR $= 60\,\mathrm{dB}$).

3) NN Architectures to Detect Incoming Symbols: Literature presents various NN network architectures used as detectors, primarily aimed at dynamically learning the MC channel parameters. Examples include feedforward NNs, a general-purpose

[15]Within this section, we assume that emitter and receiver are already synchronized.

network that takes slight advantage of a priori knowledge, such as channel memory. CNN architectures are also reported, which are more robust to channel delays. Moreover, the bidirectional recurrent neural network (BiRNN) architecture, which is more advantageous concerning the ISI in MC channels, is also discussed in the literature. This architecture utilizes past and future samples to decode the current symbol, aiming to recover knowledge from the channel's ISI. Additionally, more recently, popular attention models (see [110]) have been incorporated into the architecture to learn dependencies in the received sequence over more extended periods.

These architectures are reported to perform detection either on a symbol-by-symbol basis or in a sequence fashion. Symbol-by-symbol is the simplest one, where the NN is trained with samples corresponding to a single symbol only, like in [88], [91], [92], [100], [103], [111]. More robustly, in the sequence detection mode, the NN is trained with several symbols, where detection of the current symbol develops from observing past and future ones, like in [88], [105], [111]. The architectures that follow these two approaches, as developed in the literature, are described next, and a detailed summary of their achieved performance is provided in the appendix, Table II.

a) Symbol-by-symbol detection mode: In the symbol-by-symbol detection mode, feedforward NNs are employed to solve the MAP formulation in (1); see [90], [91], [92], [96], [103], [108], [109], [111], [112] and to find the optimal decoding threshold, as in Eq. (2); see [88], [100]. RNNs and CNNs are also reported to solve the MAP formulation as described in [108] and [95], respectively. Symbol-by-symbol detectors are also trained more efficiently using distinctive features from the input sequence. Features are evaluated based on the concentration difference of received molecules [10], [90], [96], [103], [108] or by taking the coefficients of a fitted polynomial curve to the incoming symbol, see [108]. Other examples extract features from the received sequence using one [103], [108], [109] and three [106], [107] hidden layers CNN architectures.

b) Sequence detection mode: The reported NN architectures are more diverse in the sequence detection mode. Reported methods not only develop feedforward NNs models [88], [100], but also CNNs [93], RNNs encompassing LSTM cells [93], [103], and BiRNNs [10]. Following the feedforward NN architecture, the authors in [88] develop a cascade connection as threshold decoders aimed at solving the decoding problem in (2). As depicted in Fig. 8a, decoding the current bit (b_k) accounts for the previous decoded ones (b_{k-1} and b_{k-2}). In this way, the NN inherently learns the impact of ISI in the MC channel.

The long-term dependencies in the input sequence are more efficiently learned using LSTM cells within RNN architectures, as described in [10], [103].[16] The schematic of these decoders is depicted in Fig. 8b, where three layers are connected in cascade, each layer is of length 40 and trained with 120-bit sequences; see [103, Fig. 1 b)]. Besides, using LSTM networks, bidirectional architectures can be more effectively deployed to exploit the correlation of past and future samples for decoding; see [103, Fig. 1 c)]. As past emissions are the source of ISI, their knowledge can be used to cancel out their impact in the current detection. Meanwhile, as the current emission leaks into the next ones, future samples also carry helpful information to decode the current one. Using the available input information of past and future samples, the implemented BiRNN model in [103], [111] reduces the impact of ISI and is also reported to be more computationally efficient than the Viterbi decoder for long memory channels.

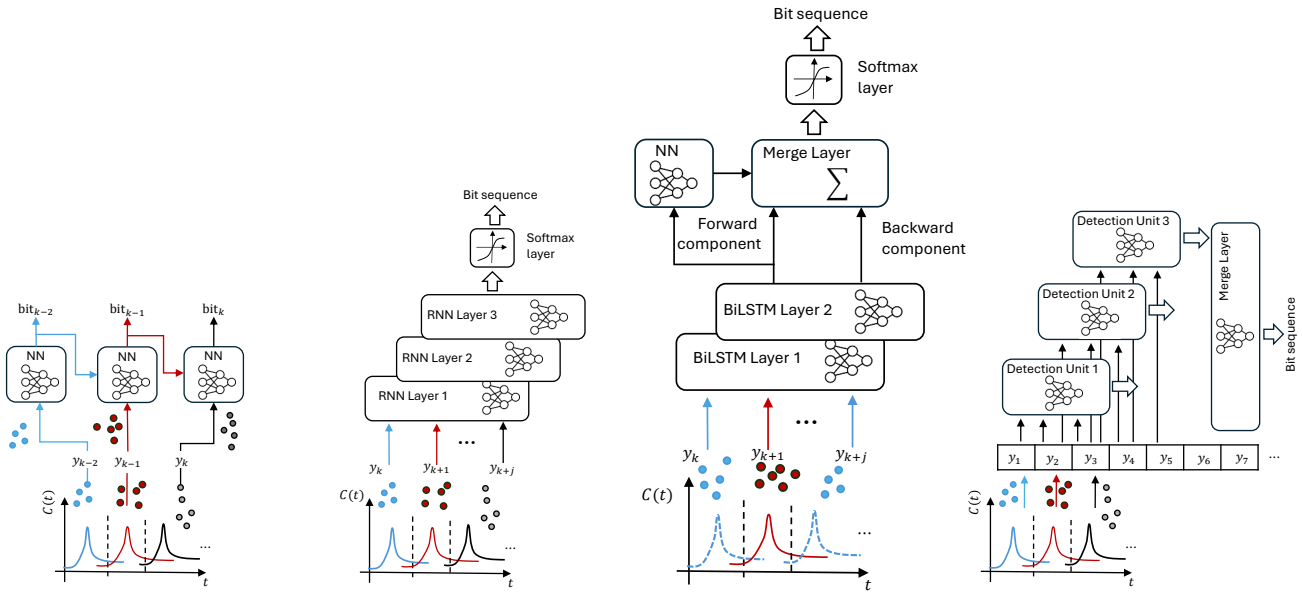
The above BiRNN architecture is further enhanced by adding a learning mechanism that merges the forward and backward pathways, as reported in [105, Fig. 1b]. As depicted in Fig. 8c, a feedforward NN adjusts the coefficients of the weighted sum for the merging layer. The model is trained using the Adam optimization algorithm (see [113]) and reports a lower BER (one order of magnitude lower) compared to detectors utilizing feedforward NNs or CNN architectures.

The above architectures can also be integrated into the sliding window architecture to further improve performance; see the schematic in Fig. 8d. Three detection units decode overlapped symbols within a sliding window, aiming to find larger correlation lags within the input sequence. For instance, in deciding the bit sequence within the symbol y_3 in Fig. 8d, not only the neighbor samples y_2 and y_4 are processed (within the Dectection Unit 2) but also y_1 and y_5 (within the detection units 1 and 3, respectively). In this way, the Merge Layer block in Fig. 8d is fed with a more extensive sequence. As reported in the literature, the detection units are implemented using BiRNNs and a Merge Layer block, which calculates the average of the RNNs outputs; see [10, Eq. (10)].

c) Attention models in the loop: There is an increasing reference to attention models aiming to enhance the learning capabilities of NN architectures in MC channels; see recent examples in [94], [98], [99], [102], [114]. As their main feature, attention models estimate the relevance of data within the input sequence, which enhances the decoding phase later; see further details in the appendix, Section C. Although the previous architectures, such as BiRNN, inherently include this feature, they are typically limited to small-range dependencies, while attention models achieve more extensive ranges within the input sequence; see [115].

The literature in the MC field reports the implementation of various architectures based on Transformers (see appendix, Fig. 17). Seeking to reduce complexity, the encoder component of the Transformer is implemented with a single self-attention unit, and the decoder component comprises only the NN unit, as described in [94], [99, Fig. 2]. Additionally, the authors in [114] further research the reduction of the Transformer hyperparameters (such as encoding vector length and input size) and components, resulting in a more compact architecture while maintaining standard operation. As an alternative variant, the encoder component of the Transformer can be replaced with a three-layer CNN module, which is connected to the input of the decoder component. Moreover, in search of a more robust architecture, the sliding window scheme in Fig. 8d is developed by implementing the Detection Units with a Transformer, as in [99].

[16]The major advantage of LSTM over RNN is avoiding the vanishing gradient problem, allowing to learn longer data dependencies; see details in appendix, Fig. 15.



(a) Cascade connection of feedforward NNs. (b) Cascade connection of RNNs. (c) Cascade connection of BiRNNs. (d) Sliding window detector.

Fig. 8: Feedforward and recurrent NN architectures as sequence decoders. The RNN modules implement LSTM cells; see [10], [103].

4) Illustrative Code Example for Symbol Detection: This section demonstrates the performance of the sliding BiRNN architecture using samples from the experimental testbed illustrated in Fig. 9a. The testbed sets a communication link over the distance of $1\,\mathrm{m}$ between the emitter sprayer and the receiver sensor. Using ethanol molecules as carriers, OOK emissions are performed with binary pulses of $T_b = 4\,\mathrm{s}$ duration each; see further details in [81, Sec. II]. We use the recorded received pulse as the expected sequence for the emissions of ones, and we modeled the randomness of the received sequence with the Poisson channel model; see [38, Eq. (87)]. Besides, we programmatically added noise molecules of concentration $10\,\mathrm{mg/L}$ yielding an SNR of $27.5\,\mathrm{dB}$. At the receiver, the observed sequence of pulses is sampled using the synchronization signal, which is assumed to be known.[17]

Due to channel memory, consecutive emissions result in ISI, which hinders the use of low-complexity schemes, such as the threshold decoder. For the decoder, we train the sliding BiRNN scheme reported in [10], [104] as a sequence detector. With the measured pulse duration from the experimental testbed ($28.6\,\mathrm{s}$ in total), we evaluate the channel memory as $L = \lfloor \frac{28.6\,\mathrm{s}}{T_b} \rfloor = 7$, which also defines the number of hidden states of the BiRNN model.[18] For training, we use the Adam algorithm [113] with a learning rate of 10^{-3}, along with 10 epochs of a batch size of 10, and with the emission of 10^5 bits.

The training loss decreases in value with increasing epochs and bit duration, as illustrated in Fig. 9b. Specifically, we observe stability after the second epoch, and losses decrease as the bit duration increases. The resulting BER falls within the range 10^{-2} to 2×10^{-6}, where the higher errors occur in the lower bit duration range of the plot ($T_b < 2\,\mathrm{s}$). Besides, as the peak of received pulses lasts for around $4\,\mathrm{s}$, the resulting BER dramatically decreases when transmissions are performed with a bit duration of $5\,\mathrm{s}$ or longer, i.e., the impact of ISI becomes negligible.

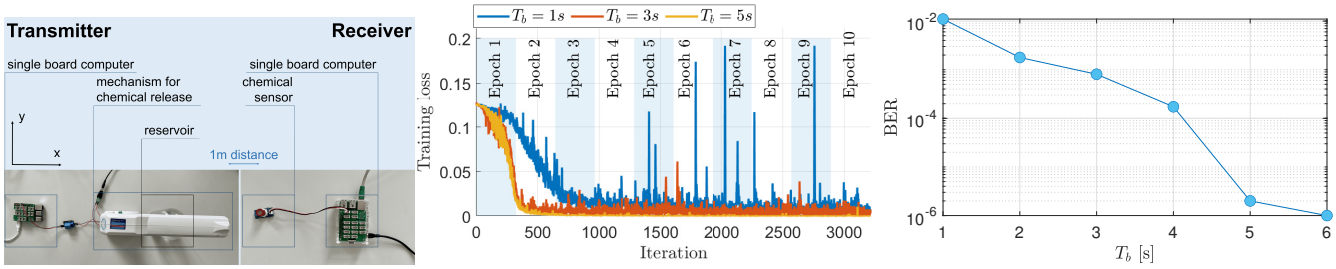
5) Concluding Remarks: NN models have proven to successfully cope with the challenging nature of MC channels as detectors, achieving BERs of less than 10^{-4} in testbed environments, see a detailed summary in appendix, Table II. The variety of environments and NN architectures is rich in the literature: Feedforward, recurrent neural networks, and attention models have been researched to decode sequences. These architectures have been tested in diffusion and drift channels, using molecules and bacteria as information carriers. However, upon reviewing the literature, little is found regarding realistic environments for precision medicine applications; the considered MC channels have mostly simplified geometries, such as free diffusion or point-to-point links, in testbeds. In future work, diffusion models can be extended to target cell-to-cell communications, including extracellular and intracellular channel models. Mobility models can be enhanced to replicate the complex communication environment in human vessels, presenting valuable opportunities for further NN-model research. Besides, the integration of these decoders with MC channel models, as introduced in the previous section, and noise suppressors, as in [117] will lead to improved performance. Although a balance between the complexity of a larger architecture, resulting from this integration, and performance should be the focus for further research.

D. Autoencoders

End-to-end learning using autoencoder (AEC) is an innovative approach in MC, which, on the one hand, mitigates the challenges associated with molecular channel modeling, and on the other hand, optimizes the entire MC system, including both the transmitter and receiver. Traditional MC systems typically divide the transceiver chain into distinct blocks such as modulation, channel estimation, synchronization, equalization,

[17]The dataset for the sequence of pulses is accessible in [116].

[18]The number of hidden states in the BiRNN corresponds with the amount of information to be retained; which in this case is not larger than the channel memory.



(a) Experimental testbed; adapted from [82]. (b) Training loss. (c) Resulting BER.

Fig. 9: Experimental testbed used for MC and measurements, as developed in [81].

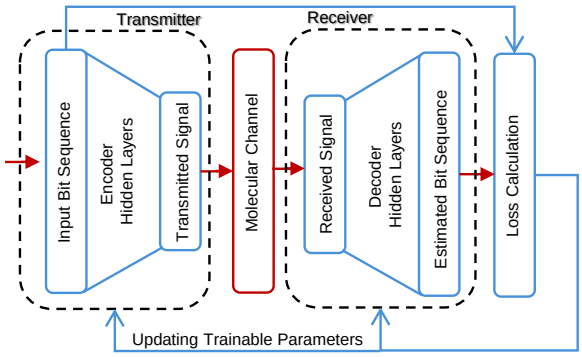


Fig. 10: AEC architecture for end-to-end learning of MC.

and demodulation. However, greater advantages can be achieved from a design that integrates the entire communication system (end-to-end), and AEC serves as the prime architecture to optimize the entire process.

An AEC consists of two NNs, known as the encoder and decoder, which are trained together in an unsupervised fashion. The encoder and decoder replace the transmitter and receiver parts of the communication system, and the entire communication pathway is optimized simultaneously in an end-to-end manner [13, Sec. 1.3.2 pp. 11]. This section summarizes the reported literature on AEC for MC, defines the end-to-end learning problem in MC, describes the deployed NN architectures, and finally, provides an illustrative code example.

1) Problem Definition for Autoencoders: The encoder and the decoder parts of the AEC are implemented as DNNs, with trainable parameters; see a representation in Fig. 10. The problem is defined as finding these trainable parameters so that the transmitted information from the emitter is restored at the receiver. Within the MC environment, the encoder generates an optimal number of emitted molecules (referred to as symbol generation or constellation design), ensuring that it can be accurately detected at the receiver side with minimal error, despite the stochastic nature and variations of the molecular channel. Therefore, the ultimate goal is to train the AEC in such a way as to minimize a desired loss function; see the corresponding block (rightmost) in Fig. 10.

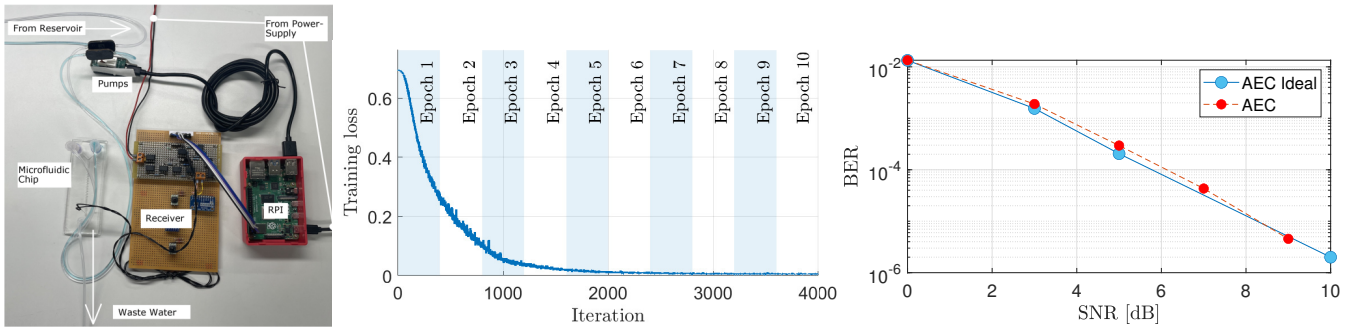
As for the loss function, an AEC adopts the cross-entropy as an effective metric for the end-to-end learning of systems. This metric measures the difference between the predicted and the actual probability distributions of the transmitted information. For symbol-based transmission, where data is encoded as individual symbols from a predefined alphabet, categorical cross-entropy is appropriate [118]. In contrast, binary cross-entropy is more effective for block-based transmission, which processes data in the form of larger binary sequences, as reported in [119].

2) Environments for Autoencoders: Advection-diffusion channels are the main type of MC channel considered in the literature for studying AECs, where a trained point-transmitter encodes information bits into the concentration of molecules and manages the release of molecules accordingly. The molecules are detected by a passive receiver, which decodes the signal and extracts the information using its trained NN, as reported in [14], [119], [120].

3) NN Models for Autoencoders: The AEC model must first undergo training before being deployed in a practical MC system. The literature presents several approaches for training AECs to facilitate end-to-end learning in MC environments, including:

a) Model-assumed training: Model-assumed training relies on full knowledge of the CIR in MC. Both transmitter and receiver are trained jointly using the estimated CIR from the testbed, which is modeled as a convolutional layer with fixed and non-trainable parameters placed between the encoder and the decoder parts of the AEC. An example is given in [14], where the CIR is obtained from an experimental salinity-based testbed as reported in [121]. This approach assumes that if the model from the testbed remains stable during runtime, with minimal variations, the transmitter will identify effective strategies for encoding input during training. Then, in the online phase, when actual transmission occurs in the experimental system, the receiver can be retrained to adapt to the slightly changed channel conditions. At the same time, the transmitter will maintain its previously established strategies. Thus, if the assumed model in the training phase is sufficiently accurate, the trained AEC will also work perfectly for transmission over the experimental channel.

Two CNNs are employed as encoder and decoder. The encoder consists of three convolutional layers, with 16, 32, and one filter respectively, each using a kernel size of 3 elements. Each convolutional layer is followed by batch normalization and an rectified linear unit (ReLU) activation function. A final normalization layer ensures a controlled number of transmitted molecules. The decoder begins with a convolutional layer featuring 16 filters and a kernel size proportional to the channel memory, effectively mitigating ISI. This is followed by an adaptive average pooling layer, a fully connected linear layer, and another convolutional layer, all of which, except the last



(a) Experimental testbed for salinity-based communication on a microfluidic chip.

(b) Training loss for the AEC.

(c) Resulting BER for the AEC when symbol duration is 300 ms.

Fig. 11: Experimental testbed developed in [121] and the results of the trained AEC in [119].

layer, incorporate batch normalization and ReLU activation. The final layer employs a sigmoid activation function. Having a differentiable channel model, the AEC is trained using the backpropagation technique (see appendix, Sections E and F).

b) Training via data-driven channel identification: The AEC can also be trained on a data-driven molecular channel representation, as proposed in [119]. The proposed training procedure involves three steps to (i) model the MC channel through an RNN, (ii) train the emitter and receiver components of the AEC, and (iii) fine-tune the trained model. A data-driven ML method is used in the first step to obtain a differentiable molecular channel representation. A specific type of linear regression, a so-called auto-regressive exogenous (ARX) method, is utilized as a promising alternative to the channel representation using NNs. This method models the channel function as an infinite impulse response (IIR) filter implemented by a trainable RNN, which is differentiable. In the second step, both the encoder and decoder parts of the AEC are jointly trained using backpropagation, while the RNN representing the channel remains fixed. Lastly, the AEC undergoes fine-tuning to address mismatches between the approximation of the channel model and the real model. The decoder parameters are adjusted using transmissions over the real channel model, while the encoder parameters remain unchanged.

c) Model-free training: In this mode, the AEC is trained without relying on any model (analytic or data-driven). To illustrate the methodology, let's consider the deep reinforcement learning (DRL) system with fully connected NNs, as proposed in [118], where information is transmitted through a stochastic, unknown channel. During each training iteration, the decoder is optimized while keeping the encoder parameters fixed, followed by the optimization of the encoder with the decoder parameters fixed. This iterative process enhances the overall system performance. Updating the decoder parameters is straightforward as it is a supervised task, and does not require backpropagation through the channel. However, updating the encoder relies on backpropagation through the channel, which requires a channel model to be available. To circumvent this challenge, the authors in [118] employ an RL model, treating the transmitter as an agent that receives the loss calculated at the receiver via an ideal separate feedback channel.

d) Transmitter-exclusive training: Transmitter-exclusive approaches perform the training on the transmitter side only, This training mode is well-suited for the IoBNT scenario, where the transmitter is usually easily accessible, as it is located outside the body, and the receiver is of low complexity and located inside, as seen in [120], [122]. The proposed asymmetric autoencoder (AAEC) in [120], [122] employs a CNN as the encoder for binary symbol transmission with adjustable transmission concentration levels, along with a simple threshold detector as the decoder, which is implemented by a single convolutional layer for training purposes. This approach seeks to mitigate the impact of residual molecules from prior transmissions by encoding symbols to counteract lingering interference, thereby reducing ISI.

Moreover, a DNN-based approach for optimizing the number of molecules released by each transmitter in a mobile molecular MIMO system is presented in [123]. While AEC is not utilized in the proposed structure, this DNN-based method can be integrated into transmitter-exclusive training for MC. The optimization is performed based on different transmitter-receiver distances to minimize the inter-link interference (ILI), thereby reducing the BER. The DNN is trained with constraints on the molecule release count, ensuring compliance with lower and upper bounds. The results demonstrate that the DNN-based approach outperforms the genetic algorithm (GA)-based approach as it achieves a lower average BER and a significantly reduced computation time. Additionally, the DNN-based optimization achieves a BER comparable to that of an exhaustive search while significantly reducing computational time.

Adaptive modulation and coding are other critical tasks to dynamically adjust transmission parameters in response to changing channel conditions. Adaptive modulation is essential for maintaining reliable and efficient communication in MC, where factors like diffusion and advection highly influence the channel. In this context, an RL-module has been proposed to optimize real-time transmission parameters, such as the modulation order and symbol duration, [124]. A gateway device connected to the RL model estimates the channel conditions based on heart rate data by leveraging a digital twin model of the human circulatory system. This solution improves the achievable raw bit rate and error performance. Additionally, to address the restricted capabilities at the nanoscale, this

architecture alleviates the computational burden on resource-limited nanodevices by offloading complexity to an external gateway.

4) Illustrative Example for Autoencoders: Here, we illustrate the performance of the AEC proposed in [119] for the salinity-based testbed in [121]. In this testbed, information bits are encoded as varying salinity levels, which are subsequently detected through corresponding electrical conductivity levels in water. Fig. 11a shows the overall setup for the testbed, where the distance from the release of the impulsive excitation is set to $3.6\,\mathrm{cm}$. During the channel identification step, we use real measurements from salinity-based communication in a microfluidic channel to train an RNN.[19]

The AEC comprises three convolutional layers, each followed by a batch normalization and a non-linear ReLU activation function at the encoder side. The decoder consists of a convolutional layer that mimics a linear equalizer, aiming to reduce ISI and noise. Two pooling layers (average and max) and a linear fully connected layer are included to downsample the features and match the size of the transmitted bit sequence. The final layer employs a sigmoid activation function to produce outputs in the range of zero to one, which is suitable for soft-input decoding. Training is conducted via simulations and over 2000 epochs, each with a batch size of 40 and a transmission sequence of $100\,\mathrm{bit}$; see the evolution in Fig. 11b. Each emission consists of binary pulses at the levels the encoder determines, with each pulse lasting $500\,\mathrm{ms}$. The Adam algorithm is used with a learning rate of 0.009 and a learning rate decay of 0.99 applied every 1000 iterations. As a result, the obtained BER ranges from 9×10^{-2} to 9×10^{-6} when the SNR is in the interval 0 to $10\,\mathrm{dB}$, which is also rather similar to the ideal case of full MC channel knowledge; labeled as AEC Ideal in Fig. 11c.

5) Concluding Remarks: AEC can inherently enhance the performance of MC by jointly optimizing the transmitter and receiver operations. Besides, this architecture effectively addresses the complexities of dynamic and stochastic MC channels. However, several practical challenges remain unresolved. For example, real-world molecular channels are time-varying, requiring continuous fine-tuning of NN parameters to adapt to unseen conditions. How to efficiently update the transmitter parameters during deployment without incurring high computational or energy costs remains an open question. Additionally, deploying resource-intensive NNs on small, biocompatible, and resource-constrained devices poses significant technical hurdles.

E. Higher Layers

So far, most of the summarized works in this section focus on point-to-point communication links. Fewer studies address higher layers, such as resource allocation (MAC layer) and localization (application layer). This section provides an overview of the reported research on integrating NN architectures into the higher layers of IoBNT networks.

1) Resource Allocation: Resource allocation problems arise when multiple users attempt to communicate using the same resource. Such a scenario is examined in free diffusion channels in [125], which investigates a transmission policy that minimizes the BER. The scenario considers an arbitrary number of emitters placed at arbitrary locations in a free-diffusion channel, and restricted to a maximum number of released molecules.

The transmission policy is devised by a feedforward NN with three hidden layers, each containing twice the number of neurons as there are transmitters, except for the first layer, which has two additional neurons. The NN is trained using the distances from the emitters to the receiver as inputs, and the model outputs the number of molecules released by each transmitter. As activation functions, the first layer uses the hyperbolic tangent sigmoid, the "purelin" (linear) in the second hidden layer, and the ReLU for the third hidden layer. This architecture achieves a BER on the order of magnitude 10^{-3} with three to five emitters positioned between 12.5 to $14.5\,\mu\mathrm{m}$ from the receiver. The network operates on a symbol time of $100\,\mathrm{ms}$, a total of 6×10^4 realeased molecules and a noisy source modeled with the a Gaussian distribution of variance 1000 units.

2) Localization: Localizing diseases and self-localizing nanosensors are two of the most expected applications of the IoBNT networks in precision medicine. Scenarios have been developed within the dynamic environment of blood vessels as in [12], [126], [127], [128], [129], [130] and in the less restrictive case of free diffusion environments as in [131].

Within the blood vessels, reported work assumes that the blood flow passively drives existing nanosensors within the vessels. In the HCS environment, the self-localization capabilities of nanosensors are particularly challenging to develop due to the absence of a reference system. A reference system for self-localizing nanosensors within the bloodstream can be anchored to the concentration level of nanosensors and their traveling time, as proposed in [126]. This work assumes that nanosensors contain an internal counter and an external device capable of resetting the internal counter of the nanosensor to zero when it travels through the heart. Then, as the nanosensor's traveling time increases as it flows through the blood vessels, the recorded traveling time per nanosensor can be used to distinguish shorter (central body) from longer (lower body) paths, thereby self-distinguishing the body region. Additionally, as the concentration of nanosensors is dependent on the particular vessel path (see the Markov model formulation in [127], [132]), it can be used as a second metric for self-localization. As such, a feed-forward NN can be trained to distinguish the traveling path of nanosensors based on these two metrics, i.e., traveling time and concentration level, as proposed in [12]. The NN is trained on data generated by the BloodVoyagerS (BVS) simulator, achieving approximately $85\,\%$ positive predictions.[20]

Localization is also based on establishing a communication link among nanonodes, as developed in [129]. This research

[19]We provide the autoencoder code at https://github.com/tkn-tub/NN_molecular_communications/tree/main/Section_III_D_autoencoders

[20]The BVS has been a popular option for data generation for localization; see [133], [134]. Access to the documented data generated with BVS and its processing is given in https://github.com/jorge-torresgomez/BVS_data.

assumes transmitter nodes with fixed positions within the human vessel and a receiver node, which is the intended one to be localized. Using the Transformer architecture, the prediction accuracy of the receiver coordinates ranges as 78 to 96 %, depending on the blood speed and the Transformer complexity.

Localization is also reported using graph neural networks (GNNs) in [130], which relies on the modeling of the HCS as a graph. This model is theoretically more expressive, leading to higher accuracies than the feedforward NN approaches. It also has the advantage of being able to generalize better to multiple anchors and their locations without having to retrain the NN [130].

Localization of nanonodes is also achieved in short-range free diffusion MC links, as developed in [131] for security applications. In a free-diffusion environment, molecules travel in all directions, and a communication link can be eavesdropped on just by passively recording the molecules in the surroundings. A trained NN can detect the presence of non-intended entities in the vicinity of the receiver by comparing the number of received molecules with the position of the eavesdropper node. The model is developed for a 2D free-diffusion environment and localizes an entity with an error of less than $4\,\mu\mathrm{m}$.

3) Data Fusion: The reported literature targets data fusion techniques in tasks related to detecting potential abnormalities. Applications include health-condition monitoring, environmental sensing, and the detection of toxic agents. Commonly implemented fusion rules are the `OR`, `AND`, or `MAJORITY` rules; see for instance [12, Sec. IV B]. However, these rules are complex to apply in practice, as they require evaluating channel-dependent parameters such as noise and interference levels, which are typically unknown. Overcoming this impediment, NNs are trained and deployed to effectively fuse data from various sensor nodes, utilizing feedforward NNs in [135] and RNN as developed by the same authors in [136]. These two architectures are developed in free-diffusion channels, where the achieved performance corresponds to a detection probability larger than 0.9 and a false alarm probability of 5×10^{-2}. Furthermore, the robustness of data fusion techniques is investigated in [137], where an RL agent is trained to optimize the observation window for collecting samples at the fusion node. The RL agent adjusts the duration of the time slot to ensure a detection probability within a confidence interval and minimize noise levels. This work reports the fusion of binary emissions from 100 nodes in free diffusion channels, utilizing the `OR` rule at the fusion node.

F. Concluding Remarks and Outlook

Within the reported literature, the deployment of NNs follows two main directions: (i) The most popular models are feedforward NNs, and (ii) deployment focuses primarily on point-to-point data communication. The comparative summary, illustrated in Fig. 15, identifies these two main directions, where we display the NNs per communication layer and MC scenario, and the size of the bubbles refers to the number of reports in the literature. This figure also identifies ways to research progress towards enabling IoBNT networks, highlighting that multipoint-to-multipoint links require additional solutions, while developments in the upper layers require more research work. Only

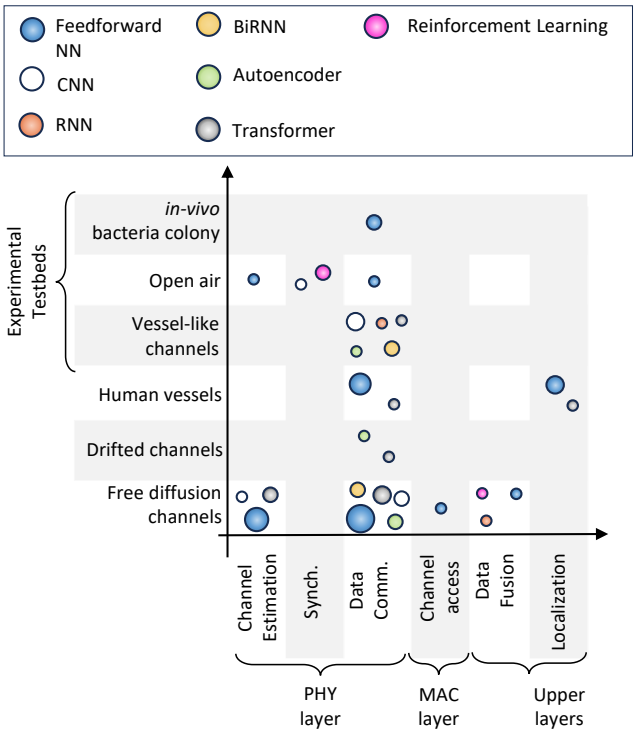


Fig. 12: Bubble plot for the number of NN architectures per application and environment as reported in the literature. The amount of reported research is reflected in the size of the bubbles.

a few works address resource allocation, localization, and disease detection, which makes this field an area for further contributions. From this figure, we also identify potential research opportunities across MC experimental environments; most of the research focuses on vessel-like channels, while less attention is given to *in-vivo* or open-air environments.

Furthermore, when examining the trade-off between performance and complexity for the various NN architectures, we provide a comparative summary in Fig. 13. This figure illustrates the performance of the architectures feedforward NN, RNN, BiRNN, and temporal convolutional neural network (TCN), those evaluated with the BER versus the transmission speed (bit time T_b).[21] The various bit times correspond to high and moderate ISI levels, as measured by the interference-to-total received molecule ratio (ITR) parameter and illustrated in Fig. 13, see [43]. This figure reveals an expected result: The BiRNN performs similarly to the other architectures, with fewer learnable parameters, including the coefficients and biases. As such, the BiRNN model captures a more accurate interrelation among samples, yielding improved decoding performance.

[21] In this evaluation, we developed the NN architectures following the complexity of the BiRNN, with five LSTM cells in the forward and backward directions to fairly cope with the channel memory. We implemented the NN architectures as decoders of pulses recorded with the experimental testbed in Fig. 9a (OOK emissions), as introduced in Section III-C4, and implemented a channel memory of $5\,\mathrm{samples}$. We did not implement the autoencoder architecture since it produces amplitude shift keying (ASK)-like emissions, which are not feasible within the experimental testbed shown in Fig. 9a. We also avoid introducing the Transformer models due to the poor performance we observed in simulations when implemented with the same number of parameters as the other NN architectures, i.e., around 300 learnable coefficients and biases. We provide open access code to the comparative analysis of the NN architectures in https://github.com/tkn-tub/NN_molecular_communications/tree/main/Section_III_Conclusions

Besides, this figure also displays an inferred approximation of magnitude orders regarding the NNs' performance and complexity. NN architectures with hundreds of learnable parameters are required in MC channels with low ISI, i.e., low-channel memory (two to five samples).

In the following, we identify various research directions regarding the application of NNs in IoBNT networks.

1) Alternative NN Architectures: Recent literature has considered mainly feedforward NNs and RNNs for MC; see Fig. 12. However, other architectures are attracting recent attention within the general literature on NN: Transformers and physics-informed architectures. Despite the examples introduced for the Transformers in Section III-C3, they are agnostically applied in MC channels. Attention models are reported in language processing for joint alignment and decoding with application to translation and text generation. This model processes time series to capture long-term dependencies. However, in the MC field, this architecture is still not fully configured to address the interdependencies in the received sequence produced by ISI. Transformer-based architectures, when applied to MC channels, require a thorough revision to fully leverage the effects of ISI channels in decoding the received sequence; refer to a comparable assertion in [138].

Furthermore, NNs are still ill-suited to non-linear scenarios (e.g., turbulent fluids) and require vast amounts of data to converge, often with poor accuracy. Nevertheless, given that the underlying *equations* governing the system are known, physics-informed NN models arise as a promising approach. This architecture can include turbulence models by integrating Reynolds-averaged Navier-Stokes or Fokker-Planck equations into the learning algorithm; refer to examples in this area in [139], [140]. A suitable balance of accuracy and computational complexity can be attained through physics-informed NNs, a topic for additional investigation.

2) Hyperparameter Tunning: Avoiding the manual selection of the NN hyperparameters, a group of techniques called hyperparameter optimization performs the automatic tuning of these; see [141]. Yet, when applying NNs to MC problems, there are no reported methods for optimizing hyperparameters such as the number of nodes, hidden layers, or the learning rate. Reported methods in the literature just selected these without further discussion; the impact of such hyperparameters and their optimization (see appendix, Section G) remains an open topic.

3) Channel Access in MC-based IoBNT Networks: Channel access strategies face the problem of the unknown number of nodes in the link, which is the case in nanonetwork formation in MC scenarios. Coordination of the nodes is pivotal to treating potential diseases effectively. Nanosensors are expected to combine their actions to simultaneously release the right amount of drugs at potential targets, such as cancer cells [142]. Protocols have been designed to develop these applications, as seen in [143]; however, the dynamics of MC channels in fluidic environments hinder their straightforward deployment. NN-based strategies developed in computer networks for node clustering, such as those in [144], can be repurposed in MC environments for nanosensors cluster formation as well.

4) Leveraging Cognitive Radio Concepts: Cognitive radio is a thoroughly explored subject in wireless environments, primarily focusing on creating intelligent communication systems that adjust their transmission methods, see [145] for details. This paradigm seamlessly integrates into MC environments; research introduced in Section III already leans toward this integration but is far from being complete. Further work on cross-layer designs and cooperative communication strategies is needed, to mention two fields where MC-schemes would benefit. Furthermore, from a system-level perspective, a cognitive unit can adapt its sensing and transmission strategies for goal-oriented applications, such as remotely mimicking a bioprocess. For example, in the development of digital twins at the cellular scale (see some developments in [146]), the interior of living cells and the interactions among them need to be reproduced, a challenge that inevitably demands a cognitive engine to integrate all levels of MC schemes.

IV. Enabling Biocomputing in IoBNT Networks

One particular challenge in nanoscale computing is the development of biocompatible nanoprocessing nodes.[22] Nanonodes are projected to perform the needed computation and host smart processing modules with biocompatible premises, as needed in IoBNT networks; see [27]. Bio-inspired micro/nano computer design is a relatively young research field; therefore, the literature reflects varying readiness levels of individual concepts, which we summarize in this section. Developments in this area lay the foundation to support the intelligent capabilities of the nanocomponents of the IoBNT networks. This section summarizes the most recent trends in technological development and identifies one of the most promising research areas for further advancements.

A. Trends in Biocomputing

Researchers have explored various approaches to enhance the intelligence of nanodevices and improve their computational capabilities, adhering to biocompatible premises. Adders, multipliers, and non-linear functions are the fundamental building blocks for developing NN components; see appendix, Fig. 15. Conceiving these arithmetic operations at the nanoscale will enable the integration of the aforementioned NN-based transmission and reception architectures within the operational nanodevices.

Already published solutions are developed along three main directions: (i) Extend the biological role of DNA-based chemical reactions within the cell for computing; see the evolution in [56, Fig. 2a] and [55, Fig. 1], (ii) develop chemical reaction networks (CRN) as in [149], and (iii) build microfluidic chips for computing as in [150]. Towards the development of NN models, a major trend in the literature is the use of engineered gene regulatory networks (GRNs) (see Section IV-C1) due to their programmability for complex calculations; see [55, Sec. 4.3]. Primarily developed within the communication-engineering community, a second trend

[22]Although this is a hardware-like requisite, much focus is also on new theoretical developments of computability, see [147], [148].

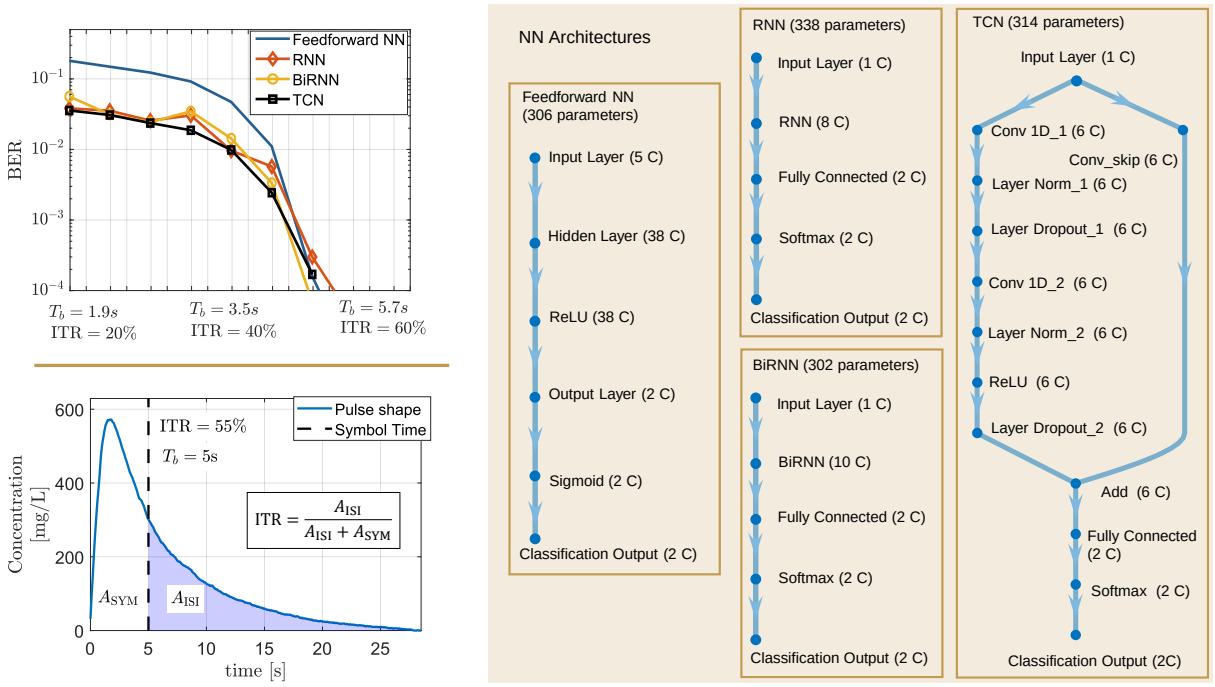


Fig. 13: Graph representation and performance of various NNs architectures with a similar number of learnable parameters (coefficients and biases). The "C" denotes the number of input channels per layer.

involves integrating chemical reactions within microfluidic chips, see Section IV-B2. Following these research streams, we trace the early developments of digital logic and digital signal processing (DSP) operations for NNs in the following sections.

B. Biocomputing in the Digital Domain

Logic gates and registers are the constituent elements to perform arithmetic operations in the digital domain; see [151, Sec. 5.2.1]. The adders and multiplier blocks needed to implement NNs can be digitally implemented as a combination of `AND`, `OR`, and `XOR` gates. Following this thought, we describe the early developments of logic and DSP operations in the digital domain in the following sections.

1) Logic gates and NNs with CRNs: The first description of CRNs realizing neuron components of NNs dates to the work in [152], where a neuron was prototyped through an enzyme reaction system.[23] Subsequent work develops neurons through the connection of logic gates, where a 2-neuron feedforward NN is realized in [149] to perform as a classifier of black and white images. The linear neuron operation, as shown in appendix, Fig. 15, results from mixing encoded bit-1s and bit-0s, analogous to acid and base compounds. The non-linear component of the neuron is realized by thresholding with the pH indicator. The classifier identifies 8×8 resolution images of handwritten digits with an accuracy larger than $95\,\%$.[24]

2) Digital signal processing in microfluidic chips: The literature also reports models for DSP operations in microfluidic channels, which can be also used for implementing NNs, see this trend within the fifth level in [154]. A finite impulse response (FIR) filter is modeled in [155, Sec. V] to implement lowpass, stopband, and bandpass filtering of molecule concentration levels. The impulse response (in the time domain) and transfer function (in the frequency domain) are derived for combining microfluidic straight channels of arbitrary length, turns, bifurcations, and junctions, and in drifted fluids where convection dominates diffusion. The filter coefficients are constructed by equating the transfer function of the microfluidic channel to a desired FIR impulse response equation. Then, the filter coefficients are determined using the least-squares formulation, see [155].[25] In this way, a multitap microfluidic circuit performs equivalent to a FIR filter having non-negative coefficients; see [155, Fig. 6].

The FIR design of the microfluidic circuit can also be extended to implement the neurons of an NN architecture. The linear components of a neuron can be evaluated with an FIR structure considering its coefficients as the learnable parameters (see appendix, Fig. 15).[26] The non-linear component of the neuron, i.e., the activation function, can be implemented through chemical reactions integrated to the microfluidic channel. Examples are given in [157], [158], where a CRN is attached to the output gate of a microfluidic channel, and similarly in [159] with a bacterial colony.

Notably, DSP operations can also be implemented using graphene biofield-effect transistors integrated with microfluidic channels. These transistors are functionalized to detect DNA strands flowing in liquid channels (see [160], [161]), thereby providing an interface between the molecular and electrical

[23]CRN is an abstraction of the dynamics of a system of chemical reactions given by a finite set of differential equations; see [55, Sec. 3.1].

[24]The handwritten images are taken from the modified national institue of standards and technology (MNIST) database [153].

[25]The microfluidic channel parameters can also be equated to an equivalent FIR block using numerical results from fluid simulators, see [156].

[26]Although the coefficients of the microfluidic FIR filter are non-negative, the linear component of the neuron can be implemented after scaling the input and the coefficients.

domains. Once within the electric domain, additional transistors can be integrated to develop the DSP components of NNs such as adders, multipliers, and non-linear functions.

C. Biocomputing in the Analog Domain

Unlike the *in silico* technologies, where NNs are deployed digitally, the integration of NNs with chemical-based technologies is developed more effectively in the analog domain. This is primarily due to the difficulties of chemically wiring arithmetic blocks and the possibilities of engineering the already in place analog arithmetic blocks within cells. The following sections summarize several reported NN implementations, which are based on biological *in-vivo* systems, compartment-based models, and chemical reactions.

1) Wet Neuromorphic Computing: Computing can be realized in the analog domain using the already in place neuromorphic capabilities of biological systems [162]. For instance, within the cell, memory and computation are inherently integrated within the GRN (which refers to neuromorphic computing). Unlike traditional von Neumann computers, where memory and computation run separately, the integrated capabilities within the cells result in a computational platform with lower energy consumption. The concept also extends to offloading computation into tissues and entire organisms. The most vivid examples are the brain organoids, where NN modules are developed in a culture of neuron cells as in [163], [164].

Within the cell, arithmetic can be realized in the analog domain by engineering DNA circuits, as summarized in [55], [165], [166]. Additionally, arithmetic can be achieved through the assembly of different DNA tiles, as described in [167, Sec. 4.3] and [168]. Examples include adders, multipliers, subtractors, power-laws, and dividers realized by combining DNA circuits, as illustrated in the early work in [169]. To a greater extent in the literature, feedforward NNs are the focus within the natural interaction among the network of genes that naturally resides within the cell. For instance, NNs are identified with genes as the computing units, transcription factors as the stimulus, and the degree of influence of transcription factors in the genes as the weights, see [170], [171] and the early work in [172]. Using these analogies between DNA-chemical components and NN functions, the potential of operating NN structures already within the gene regulatory circuits is extensive; as a result of mining the connection in the gene regulatory network more than a hundred single-layer NNs have been identified [166, Fig. 3].

The first work demonstrating the potential of DNA-based engineering relied on the DNA strand-displacement technique, as developed in [173].[27] In this technique, neurons are realized through the reaction of input strands and toeholds (DNA segment) with three gates. The multiplier gates output a total number of DNA strands in proportion to the weight, i.e., the operation $\omega_i \times x_i$, where ω_i is the weight and x_i is the input. The integration gate is chemically related to the output strands from the multiplier gates in a way that a reaction product reflects a strand as the sum of the inputs, i.e., $\sum \omega_i \times x_i$. Finally, thresholding is realized through a chemical reaction at the last gate.

Instances of concrete developments synthesize NNs in the logarithmic domain for a more natural replication within gene regulatory systems. In the logarithmic domain, coefficient multiplication is transformed to exponentiation, and summation to multiplication, both operations implemented using two gene circuits, as illustrated in [175, Fig. 1a)]. These gene circuits are later connected to a third one to implement the non-linear function of the neuron, as illustrated in [175, Fig. 1j)]. Other examples for gene circuits develop single-layer NNs as: 2 inputs and 2 neurons in [175], 3 inputs and 3 neurons in [176], [177], [178], 2 layers with 2 inputs in [179], and a more complex CNN architecture in [180]. Testing their powerful computational capabilities, these DNA-based NNs are reported for imaging messenger ribonucleic acid (RNA) molecules and for regulating cell behavior such as cell-to-cell communication and adhesion; see [57, Sec. 4].

On a larger scale, other examples include combining multiple engineered cells as in [181] and the striking performance of brain organoids cultured cells [182]. A single-layer NN is conceived through cell-to-cell communications in a diffusive medium. The weights of the NN are set by the diffusive properties of the information molecules in the medium. The required non-linearity is realized by the gene expression inside the *E. Coli* cell. A different approach is developed in [183], where multiple bacteria serve as senders, and they release a number of molecules in proportion to the magnitude of the learnable coefficients of the NN. Another bacterium serves as a receiver and implements the nonlinear function. Altogether, the sender and receiver bacteria perform as a single-layer NN. The concept's applicability to ML-tasks was demonstrated in [183], where a classification task was solved using the proposed approach.

Brain organoids, which develop in cultures of human stem cells, are known to perform complex tasks in real-time. For instance, the work in [164] illustrates the training of living neuron cells to play the game Pong, and the work in [163] illustrates the prediction capacity of spatio-temporal data from hundreds of thousands of neural cells. The learning capability of these networks relies on the neuroplasticity of the neuron cells. The experimental work illustrates the response of surrounding cells to an input stimulus, accompanied by the development of new connections. This technology offers two key advantages related to scalability: Reduced energy consumption (the human brain uses approximately 20 W) and faster training (it takes about 4 epochs to match the performance of LSTM cells, which require 50 epochs).

2) Analog computation with CRNs: DNA regulatory circuits are also in the domain of CRN, as they provide the substrate for the reaction pathways.[28] However, other chemical solutions implement the computation without engineering DNA molecules. These solutions look to overcome limitations such as temperature sensitivity, long computation times, and a large number of chemical components, as stated in [184]. For

[27] DNA strand displacement regulates the gene expression and is capable of creating universal computation [174].

[28] Substrates refer to reactant compounds in a chemical reaction.

instance, the metabolic circuits of cells are engineered in [185] to implement 2 and 4 input NNs of a single layer. This work realized positive weights with the concentration level in a cell-free framework, and negative weights were identified with attenuating reactions. However, due to the large number of required reactions, these techniques are often combined with DNA-computing [186], [187], [188] for ease of implementation. Notably, CRN was used to implement a chemical NN that even allows backpropagation [189] and, thus, online in-situ NN training.

3) Compartment-based models: Nanoscale computing units can also be developed based on the propagation of molecules between various connected compartments and chemical reactions within these compartments, as in [190]. Matrix weights can be mapped to a nanoscale computing unit by adjusting the compartment volume [190]. While the work in [190] especially focuses on a mathematical theory of a compartment-based reaction-diffusion computer, a proof of concept has been provided in [191]. This work presents three chemical processes for implementing a matrix multiplication in a nanoscale computing unit: phase transition, precipitation, and acid-to-base reaction. Experimentally obtained measurements in a lab-scale implementation demonstrate a "reasonable accuracy" for all three proposed chemical reactions, providing a path towards a realization at the microscale level in future work. Further theoretical work in [192] extends the concept to molecular nano NNs and outlines practical applications. For instance, compartments are connected to realize a molecular nano NN, which mimics an artificial NN [192]. This work also validates the proposed molecular nano NN by applying classification and regression tasks.

4) Reservoir computing in MC channels: In pursuit of more powerful NN architectures, reservoir computing has also been reported in the MC domain to realize RNN models [193]. Reservoir computing utilizes a static RNN structure, placing the network's intelligent component in the output layer instead [194]. The solution in [193] implements the reservoir using the MC geometry: Point transmitter-free diffusion MC channel-ligand spherical receptor. This end-to-end MC channel implements the state update equation of the RNN architecture; see [193, Eq. (1)]. The output layer is designed as a vector comprising samples of the number of bound molecules at the receptor, collected over a time window; see [193, Eq. (9)]. The training of this model involves adjusting the vector component as weights in the output layer, and it is implemented in a silicon interface. The research in [193] also discusses the impact of MC channel parameters on network performance, illustrating ranges of preferred values for distances, diffusion coefficients, and binding coefficients; see [193, Fig. 4].

D. Resource Demand and Feasibility of Bio-AI Unit Implementation

The required arithmetic operations for NN operation, such as adders, multipliers, and non-linear functions, are highly resource-demanding to be implemented by logic units within cells or microfluidic devices. For instance, an eight-bit adder would require 16 `AND`, 8 `OR` and 8 `XOR` two-input gates; see [151, Sec. 5.2.1]. The NN-based decoder architectures require around 300 learnable parameters, see Fig. 13, thereby nearly the same number of adders, yielding a total of $300 \times (16 + 8 + 8) = 9600$ logic gates to be implemented. Implementing a gate per cell or microfluidic circuit yields a circuit physical dimension on the millimeter scale, preventing low-dimensional developments on the micrometer or nanometer scale.[29] Furthermore, hundreds of learnable parameters also lead to a high number of connections, which imposes a more significant challenge regarding the chemical wiring of the corresponding logic units.

By contrast, gene regulatory networks within the cell allow a larger space density of calculations. Following the entries in [166, Fig. 3] related to feedforward NN implementation, we can find networks of 26 learnable parameters maximum.[30] Therefore, 14 connected cells can reach the complexity reported for decoders, i.e., approximately 300 in Fig. 13, and still fit within the micrometer range. Furthermore, reported examples of brain organoids [182] and reservoir computing [193] are also candidate technologies for developing NN architectures at the micrometer scale. Reported research illustrates designs in the range of 10 to $100\,\mu\mathrm{m}$, although not limited to further scaling down their dimension. These advancements demonstrate the practicality of NN architectures for IoBNT applications; however, a design for NN module interfaces (both bio and electric) is still not available.

E. Concluding Remarks and Future Outlook

In summary, there are numerous approaches for implementing NNs in a biological and/or chemical manner. Current challenges can be summarized as follows: The scalability of NNs is limited in all concepts due to geometrical [191], chemical [189], or biological [175] constraints. While numerous concepts have been shown to work in practice, a unified framework for the design and standardization of molecular NNs is currently missing. The development of such a unified scheme is anticipated as current research in the field is increasingly progressing.

Besides, we identify the following research directions related to the implementation of NNs in the biological domain:

1) MC Networks as ML Platforms: Inspired by paradigms in WiFi and ML joint development [197], MC networks can also constitute a running platform for ML development (see [13, Chap. 12]). Various examples exist in wireless networks where intelligence is distributed among nodes. Such ideas can be extended to MC networks for deploying large NN architectures challenging to fit on a single node.

2) Deployment at the Nanoscale Level: The current integration of MC and ML relies on macro-scale external computers hosting the NN deployment; see the examples

[29]We assume the dimension of a cell is in the range of 1 to $2\,\mu\mathrm{m}$ as for E. coli [195] and for microfluidic circuits in the range of 1 to $10\,\mu\mathrm{m}$ [196].

[30]We perform this calculation assuming the number of hidden nodes follows the number of inputs. For instance, a network of 4 inputs and 3 outputs comprises 4 nodes in the hidden layer. As the total number of weights corresponds to the total number of connections among nodes, and the biases the total number of nodes, the calculation yields $4 \times 2 + 4 \times 3 + 3 \times 2 = 26$ learnable parameters.

in Section III. Examples of calculations at the micro-scale are already accessible, as introduced in Section IV; yet, little research has been published on integrating such solutions with NNs. An approach to develop NNs in the digital domain can be attained by the digital representation of the molecular signals and the arithmetic operations performed at the bit level, as standard electronic circuits work. In this way, a digital-like biocomputer can be implemented by integrating a molecular analog-to-digital converter; see developments in [198], [199], [200], [201] for digital adders and multipliers, as well as examples in [167]. This integration would require developing a bio-framework to interface the components. On the other hand, this development would require low-precision logic within the circuit due to the limited capacities at the nanoscale level, i.e., adders and multipliers with a low-bit representation of less than a byte. This restriction leads us to another challenge: The impact of quantization errors also increases. The impact of these errors needs to be investigated, and mechanisms to mitigate them must also be implemented; see, for instance, the delta-sigma modulation in [202, Sec. 12.3] as used to decode single-bit encoded transmissions.

V. Early Work on Explainable Neural Networks

Over the past few decades, MC has evolved from theoretical concepts to practical implementations, leveraging advances in nanotechnology and bioengineering. Concurrently, the integration of AI has transformed molecular communication by enhancing the efficiency and robustness of signal processing, encoding, transmission, and decoding processes, as discussed in [53]. Despite these advancements, the complexity and opacity of AI models present a significant barrier to their full realization in MC systems. The solutions reported in this section tackle these challenges, introducing methods to make the application of NNs transparent. This section outlines the motivation behind this research direction, followed by a brief description of the fundamental tools, a summary of the reported research, and an illustrative code example.

A. Motivation

The application of NNs in MC has traditionally focused on optimizing various aspects of the communication process, see examples in Section III. However, the "black-box" nature of these models often leads to a lack of transparency and interpretability, which is problematic in critical applications where understanding the decision-making process is crucial. This is where explainable artificial intelligence (XAI) comes into play, offering a suite of techniques designed to elucidate the inner workings of AI models, making them more transparent and comprehensible.

The motivation behind XAI research in MC is twofold. Firstly, there is a pressing need to bridge the gap between the sophisticated NN algorithms used in MC and the requirement for transparency and explainability. XAI can help identify and mitigate biases and errors within NN models, leading to more accurate and reliable communication. Moreover, XAI can facilitate interdisciplinary collaboration by making NN models more accessible to researchers and practitioners from various fields, including biology, chemistry, and engineering.

B. Explainable and Interpretable Molecular Communication

The interpretability of a model is a combination of factors related to its complexity and transparency. Models with fewer parameters and near-linearity become more interpretable, albeit with limited performance. Performance and explainability are counteracting; balancing these two aspects poses significant challenges for designing high-performance methods for sensitive use cases. Within the scope of this challenge, several critical metrics help in evaluating the effectiveness of explanations.

1) Explainability Metrics: The following metrics evaluate to what degree NN-driven predictions can be trusted and understood within the intricate dynamics of nanoscale interactions [203]:

a) Fidelity: Fidelity quantifies how accurately an explanation with a surrogate model reflects the behavior of the original model. A surrogate model serves to explain the predictions of the primary model, and fidelity measures the alignment between the two. A high-fidelity explanation implies that the surrogate model accurately captures the core decision-making processes of the original model. This is particularly crucial in MC, where understanding the molecular mechanisms that drive predictions can shed light on the interactions at the nanoscale.

b) Sparsity: Sparsity reflects the simplicity of an explanation by quantifying the number of features involved. Higher sparsity corresponds to fewer features being included, leading to more concise and interpretable explanations. In the realm of MC, where interactions at the nanoscale are inherently complex and multifaceted, sparse explanations are particularly valuable to identify the most relevant physical processes that influence predictions.

c) Stability: Stability measures the consistency of an explanation when the input is subject to small perturbations. In MC, where environmental factors and molecular interactions are dynamic, stable explanations ensure that the insights provided by the NN model are reliable and robust across different scenarios.

d) Causality: Causality assesses whether the features identified in the explanation are genuinely responsible for the model's output. In the context of MC, this involves identifying whether specific signaling molecules or interactions directly influence the predicted cellular responses. This metric is particularly valuable for validating experimental designs and ensuring the reliability of insights in complex molecular networks.

e) Comprehensibility: Comprehensibility reflects how easily a human can interpret the explanation. Although difficult to quantify directly, it can be associated with the simplicity of the explanation. For example, a shorter rule-based explanation with fewer conditions is generally easier to understand than a complex one.

2) Explainability Techniques: Benefiting from these mentioned metrics, there are several key techniques for providing explanations in NN models, as summarized below:

a) Feature Attribution Methods: These methods assign an importance score to each feature. The most common techniques include local interpretable model-agnostic explanation (LIME), shapley additive explanation (SHAP), layer-wise relevance

propagation (LRP), and deep learning important features (DeepLIFT).

LIME works by perturbing the input data and observing the changes in prediction to create a local, interpretable model. A weighted linear model is then trained with the perturbed data, aiming to approximate the original model; see details in [204].

SHAP values are derived from cooperative game theory, providing a unique solution that allocates payoffs to players in the fairest manner (features). The SHAP value for a given feature is evaluated as in [205, Eq. (4)]. This evaluation considers all possible combinations of input features, computing the marginal contribution of each feature to the model's prediction. In essence, SHAP estimates how much a particular feature, when added to different subsets of features, changes the prediction, and then averages these effects to yield a global contribution score. This mechanism allows for a consistent and locally accurate explanation of feature influence. We develop an illustrative example for this method in Section V-D.

LRP is a method designed to attribute an NN's prediction back to its input features [206]. This is achieved by backpropagating the relevance from the output layer through the network to the input features, where the relevance is determined by distributing the contributions proportionally to the weighted connections between neurons. In the context of MC, LRP helps identify which nanoscale signals or features play the most critical roles in the prediction, enabling better interpretability of complex NN models.

DeepLIFT extends LRP by comparing the actual input to a reference input or baseline, quantifying the difference in model outputs; see further details in [207]. By explicitly accounting for deviations from a baseline, DeepLIFT is particularly useful in scenarios where the relative importance of nanoscale molecular changes can provide crucial insights into the system's dynamics. For instance, in MC systems where slight variations in vesicle concentration, timing, or signal shape may signify different biological states or disease progressions, DeepLIFT can highlight which input perturbations most significantly influence the model's prediction. This enables researchers to trace back model decisions to specific molecular behaviors—such as an unexpected peak in molecule release rate or a delay in arrival time—thereby improving interpretability and fostering trust in AI-driven diagnostics or monitoring systems. Unlike gradient-based methods that may suffer from vanishing signals, DeepLIFT preserves contribution scores even in saturated or nonlinear regions, which are common in biochemical signaling cascades.

b) Saliency Maps: Saliency maps, commonly used in computer vision, highlight regions of an input image that are most important for a model's prediction [208]. However, this can be helpful in MC as they visually highlight the most influential features in the NN model's decision-making process. This technique generates visual representations that show which areas of input data contribute most to the model's output, allowing researchers to quickly identify key molecular factors or patterns driving the communication process.

c) Counterfactual Explanations: These provide insights by answering "what-if" questions [209]. For a given input, a counterfactual explanation identifies the smallest input change such that the model output changes to a target value. Where molecular interactions dictate system responses, counterfactual explanations can help researchers understand the causal relationships between molecular signals and communication outcomes. For instance, counterfactual explanations can reveal how these changes would affect the model's prediction of system behavior by modifying the concentration of a particular molecule or altering a specific signaling pathway. This technique enhances the interpretability of AI models by providing actionable insights into the conditions under which MC might behave differently, helping researchers better understand and control nanoscale interactions.

d) Manual Permutation Importance: The manual permutation importance (MPI) method is a model-agnostic technique used to quantify the contribution of individual features to a machine learning model's predictive accuracy. It measures feature importance by randomly shuffling the values of a specific feature and observing the resulting increase in model error, effectively disrupting its relationship with the target variable. This approach was first introduced in the context of Random Forests [210], where it was utilized to evaluate the significance of variables by assessing the impact of their permutation on prediction accuracy. For MC, where factors like vesicle diffusion, receptor binding, and molecular degradation introduce stochastic variability, explainability methods like MPI ensure that AI-driven models remain not only accurate but also biologically meaningful. By identifying which features most significantly affect model performance, MPI aids in understanding the underlying biological processes and enhances the interpretability of complex MC models. We develop an illustrative example for this method in Section V-D.

C. Reported Research

In recent advancements within MC, traditional methods often struggle with evaluating complex end-to-end channel models and minimizing the BER. Although these NN models are a promising alternative to conventional detection methods, a critical limitation is their inherent lack of transparency. As summarized below, research studies have begun to address these limitations, with a primary focus on feature attribution methods for improving interpretability. These examples motivate further research.

XAI methods have been applied to elucidate the inner workings of NN-based symbol detectors for a 2×2 MIMO MC channel in [211]. To interpret the NN's behavior, the authors employed XAI techniques such as LIME, partial dependence plot (PDP), and the individual conditional expectation (ICE) method. Their application revealed that the trained NN effectively operates as a threshold detector in the low ISI regime. This association of the NN with a threshold detector or a slope detector is the interpretation of the model, which also allows for providing proof of the correctness of the NN-based detector.

In a similar direction, researchers generated synthetic data based on MC channel models and real testbed measurements to train an NN for binary symbol detection in [212]. The key objective was to demystify the "black box" nature of NNs and provide assurance of their correctness in symbol detection tasks.

To achieve this, they employed XAI techniques, such as LIME and ICE plots. The analysis revealed that the trained NN's decision-making process closely mirrored that of standard peak and slope detectors in low and high ISI regimes, respectively. The NN behaved similarly to traditional methods used in MC for symbol detection based on signal features such as amplitude peaks and signal slopes.

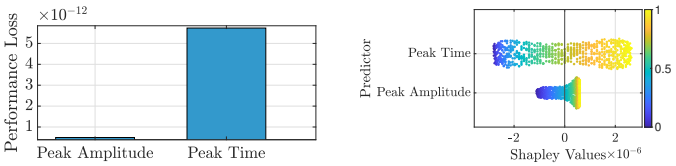
Another notable development in MC involves the integration of the IoBNT, which aims to create a network connecting both artificial and biological units to the internet. In this context, a common scenario features an external transmitter with substantial computational resources and an internal receiver with limited computational capabilities. To address the computational asymmetry between the transmitter and the receiver, the authors in [122] introduced an innovative AAEC architecture for end-to-end learning in MC systems. The researchers explored the explainability of the NN-based transmitter with a surrogate model. They demonstrated that the NN-based transmitter could be approximated by a zero-forcing precoder for low and moderate ISI. This allows us to anticipate the NN's performance with the well-known formulation of linear precoders. Additionally, the research addresses concerns about the transparency and trustworthiness of NN-based systems, which often function as a "black box". By investigating the explainability of the AAEC through XAI methods, the study contributes to building trust in AI-driven communication systems—a vital factor for sensitive applications envisioned in the IoBNT.

By integrating SHAP into MC detector design, the authors in [213] identified crucial feature points in the received molecular signal. SHAP-driven analysis highlights the most informative segments of molecular signal waveforms, providing a systematic explanation of how different NN models arrive at detection decisions. Previous work in MC has demonstrated the potential of combining model-based methods with ML to compensate for unknown channel parameters; however, the inherent opacity of deep neural networks remains a persistent barrier to practical deployment. By applying SHAP, this approach maintains the adaptability and high accuracy of data-driven detectors while enhancing explainability, ultimately facilitating safer and more reliable MC applications in areas such as targeted drug delivery and in vivo biochemical monitoring.

D. Illustrative Code Example and Results

As a code example, we take the model already developed in Section III-A4, which deploys a 2 node-NN as a distance estimator between immune and cancer cells. The model first extracts two features from the number of vesicles in immune cells; these two features are the amplitude and the time coordinate (peak time) of the minimum of the slope of this sequence; see Fig. 6 a) and b). The NN inputs are these two features, and the output is the predicted distance, as illustrated in Fig. 6 c). Following the work in [214], we implemented the MPI method to evaluate the significance of each feature for the NN's output, and also included the SHAP value for further illustration.

The results depicted in Fig. 14 underline the importance of the peak position. Feature 2 (peak time) demonstrates a significantly higher performance loss increase compared to Feature 1 (peak amplitude) in both methods, suggesting a stronger correlation between the peak time and the distance. Thereby, the peak time likely encodes key characteristics of vesicle-mediated communication. In addition to permutation-based analysis, SHAP values provide a more nuanced view of feature contributions across individual predictions. As illustrated in Fig. 14 b), the x-axis denotes the SHAP values, reflecting how much each feature influences the predicted distance, while the color gradient indicates the actual feature value (ranging from low in blue to high in yellow). Notably, the peak time spans a wider SHAP value distribution, showing it has both a stronger and more diverse influence on predictions compared to peak amplitude, whose SHAP values are centered tightly around zero. The consistently increasing SHAP values for higher peak time values imply a monotonic positive relationship with distance, confirming its dominant and biologically plausible role in vesicle timing as an informative signal of cell-to-cell proximity.[31] This consistency highlights its resilience and importance in predictions.



(a) Permutation importance. (b) Shapley summary.

Fig. 14: Feature importance analysis using manual permutation importance.

The results in Fig. 14 are in direct correspondence with the expected properties of the CIR in free diffusion channels. There exists a direct correlation between the travel time (here encoded within peak time) and the square of the distance, expressed as $t_{\text{peak}} \propto \frac{d^2}{D}$, as referred to in [215, below Eq. (2.8)]. The NN architecture effectively filters the input associated with the peak and assesses the direct relationship between peak time and distance.

E. Concluding Remarks and Outlook

The transparent operation of technical products is crucial in sensitive applications such as healthcare, where understanding the inner workings of deployed NNs may determine their adoption within the IoBNT framework. Currently, this aspect of NNs in IoBNT is explored less than their functional applications, see Fig. 12. Research in this area is still in its early stages and primarily focuses on the PHY layer. Preliminary research interprets the operation of NNs in terms of classical methods, including threshold detectors, slope detectors, and zero-forcing precoders. This literature reports explainable methods in specific ISI regimes, although an interpretation of the switching of the same NN model between low and high ISI regimes is still missing; this is a topic for further research.

More work is needed in this field due to the relevance of explainability methods for healthcare applications and the small number of contributions. Explainability in the context

[31] We provide this code at https://github.com/tkn-tub/NN_molecular_communications/tree/main/Section_V_XAI.

of MC networks is still nascent, and more research is needed to understand the inner workings of the models and to ensure their reliability in real-world applications; see all examples of NN models reported in Section III, where most of them miss a proper interpretation of their inner functioning.

Lastly, a promising frontier in MC research involves integrating physical laws directly into the NN training pipeline, commonly referred to as physics-informed neural networks (PINNs). Unlike purely data-driven approaches, PINNs embed the governing partial differential equations (PDEs) of molecular transport (e.g., Fick's diffusion law, reaction-diffusion kinetics, or fluid dynamics in vesicle-mediated signaling) as additional loss terms in their training objective. In MC, these constraints capture the evolution of molecules in nanoscale environments, improving data efficiency and enforcing consistent behavior despite limited observations. Not only do these models exhibit better generalization to previously unseen conditions, but they also offer greater interpretability of the solution space, as latent layers must satisfy fundamental conservation laws and boundary conditions intrinsic to molecular transport.

VI. The Backbone of Neural Networks: Training Data

Data is the backbone of NN training, and this section provides an overview of existing datasets for MC. We include data sets that have not yet been exploited in the context of NNs. Furthermore, the datasets are discussed in terms of their generation and accessibility, and some remarks are made regarding the documentation of the datasets. Finally, current limitations and future perspectives on synthetic data generation are outlined.

Many datasets, particularly those derived from simulations and published in papers, i.e., data used for plotting, are not publicly available. This lack of accessibility influences the reproducibility and validation of the corresponding findings, limiting the research community's ability to build upon existing research. The appendix, Table V, provides a comprehensive overview of existing publicly available datasets. By focusing on publicly available and accessible datasets, we aim to provide a comprehensive summary that researchers from different domains can utilize.

A. General Considerations

Data can be generated based on observations of physical or virtual processes, such as wet-lab experiments or computer simulations. Data generation can be time-consuming and resource-intensive. Avoiding repeating the process, datasets shared among different research groups can (i) facilitate *reproducible research* and (ii) provide an *abstraction layer* for the actual process of interest, e.g., MC between synthetic cells, that allows other researchers to develop models and algorithms without the need to (re-)run and observe the process.

Both these goals hinge on the *availability* and the *usability* of the datasets. Also, a thorough *documentation* of the shared data is required. These aspects, which relate primarily to data and metadata *handling*, will be covered later in this section. However, for determining *which data* to collect in the first place, the properties of the process from which data is generated and the primary purpose of data collection play a key role.

The purpose of collecting data can be to validate a specific theoretical model. In this case, it is essential to control the experimental conditions as best as possible and obtain a representative set of data samples under these conditions. The main focus of data collection, in this case, is to confirm or reject a specific, often quantitative, hypothesis on the process, e.g., the average received SNR, that originates from the theoretical model.[32] On the other hand, as is the case, e.g., for NN training, data can also be used to build a model. In this case, data should be collected to avoid training set imbalances and overfitting the model on a specific narrow subset of experimental conditions.

Training data can be distinguished into two types: (i) Synthetic and (ii) experimental data. In this survey, synthetic data refers to artificially generated data created through algorithms or simulations mimicking real-world scenarios. Experimental data is associated with collecting experimental samples, alongside the challenges of building controlled environments where various parameters can be precisely changed.

B. Synthetic Data Generation

1) Existing Datasets: Existing datasets refer mainly to two environments: Pipe-like and free-diffusion channels in fluid and air-based media. For pipe-like channels, the authors in [156] introduce a cylindrical water-filled duct designed to study steady-state flow conditions across various channel lengths. The molecules are released uniformly distributed over the channel's cross-section, propagate through the channel in a flow-dominated regime, and are counted by a cylindrical observing receiver. The authors benchmark their simulation results against particle-based simulation (PBS) and an analytical channel modeling approach. This dataset is accessible in [216] and also includes a template configuration for the OpenFOAM [217] environment, making it easier to get started with the simulation tool.

The above dataset has also been extended in [218] to cover not only the flow-dominated regime but also the dispersion and the mixed regimes [219]. Furthermore, targeting healthcare applications for advanced plaque modeling [220], the work in [221] develops a dataset accessible in [222], using the computational fluid dynamics (CFD) solver OpenFOAM. A previously published dataset in [216] provides a framework for MC specific post-processing, i.e., utilizing Python to analyze the OpenFOAM simulation output, for example, in terms of the CIR.

The dataset in [223] includes results from CFD simulations performed in the proprietary ANSYS software to study the transmission of airborne pathogens in turbulent airflow environments. The simulations examine the spread of pathogen-laden droplets and aerosols, "mimicking scenarios such as coughing-induced flows" [223].[33] An overview and summary of all simulation results, along with a detailed explanation, can be found in [225].

[32]The theoretical model can also be an NN, in which case the hypothesis results from the inference step.

[33]The corresponding code for modeling MC scenarios is available as a GitHub repository in [224].

While the datasets discussed so far mostly contain data from fluid dynamics simulations, some other datasets reproduce time sequences as a result of processing algorithms. This is the case within the dataset in [226], where the estimated distance in a macroscale MC system, consisting of an electric sprayer containing alcohol in a reservoir without an additional fan and an alcohol sensor, is reproduced as Matlab files and raw data. The estimated distance is produced by ML-based methods, see [63], and the Fluid Dynamics-based Distance Estimation (FDDE) algorithm, see [227].

The dataset in [228] contains, in addition to the dataset itself, the code for creating the dataset using Matlab, which is published on a university server, making the results from [229], [230] accessible to other researchers. The authors of [229], [230] address the development of multiple-access schemes and detection strategies for diffusion-based MC, focusing on mitigating ISI and multiple-access interference (MAI). Furthermore, [229] and [230] introduce the type-spread molecule shift keying (MoSK) modulation technique, which uses additional molecule types to reduce the ISI effect. Type-spread MoSK is expanded for a multiple-access system to support multiple nanomachines. In addition, a molecular code-division multiple-access scheme is proposed in [229], [230], relying on two molecule types.

Published datasets can also describe the parameters used in a publication (here: csv file) [231]. For example, in [232], an improved channel model for MC with a partially absorbing receiver is proposed. The partially absorbing receiver has four parameters determined by particle swarm optimization (PSO). The optimum of the determined parameters, as mentioned in [232], can be found in the corresponding dataset in [231].

MCFormer, a Transformer-based MC detector which adapts natural language processing methods to MC, is documented in [114].[34] Moreover, an optimized PBS algorithm that leverages matrix operations to efficiently generate training data with reduced complexity compared to conventional approaches is presented. The results in [114] demonstrate that MCFormer delivers near-optimal accuracy in noise-free conditions, outperforming DNN.

2) Dataset Generation Tools: In addition to the published datasets, simulators and software tools are also available to generate synthetic datasets based on users' requirements and system model parameters. Generally, in some MC simulators, the medium flow is simulated explicitly, while other tools utilize analytical models or approximations for the medium flow or neglect the medium flow entirely [234]. The latter set of tools focuses mainly on Monte Carlo simulation and is referred to as *flow-agnostic simulators*. In contrast, the physical fluid flow characteristics are considered in the former one, referred to as *flow-aware simulators*. The Table VI in the appendix provides an overview of the various simulators used to generate the datasets. Most of the tools in appendix, Table VI, are publicly available, except for the commercial simulators COMSOL Multiphysics, ANSYS Fluent, and Matlab, for which a license is required to utilize the full range of available functions. Further requirements also arise for the MC graphical processing unit (GPU) simulator (also known as "parallel simulation framework for nanonetworking" [235]), as it requires an Nvidia GPU supporting the Nvidia compute unified device architecture (CUDA) [235].

The flow-agnostic simulators encompass tools designed for various applications, such as bacterial MC (nanoNS3 [236], BNSim [237]), reaction-transport modeling (URDME [238]), and Brownian motion-based simulations (Smoldyn [239], [240], N3Sim [241], [242], AcCoRD [243]). Diverse computational frameworks support these simulators, including Matlab, COMSOL, and NS3. In contrast, fluid dynamics simulators used in the context of MC, such as OpenFOAM, COMSOL Multiphysics, and ANSYS Fluent, are primarily based on the concept of CFD and numerical solutions of the Navier-Stokes equation in general. Availability varies significantly, with several simulators offering freely accessible source codes (e.g., OpenFOAM, Munich microfluidic toolkit, and NS3). In contrast, others, like ANSYS Fluent and COMSOL Multiphysics, are commercially licensed. This highlights the trade-off between cost and accessibility across the simulators, with open-source tools typically targeting niche research applications and commercially available software catering to broader engineering applications.

In addition to the simulators, other existing analytical simulation codes, data generation codes, and models have been published on GitHub as repositories for code development. The projects include files for artificial neural network (ANN) models, analytical simulation codes, data generation codes, and LSTM codes published, including Jupyter Notebook, Python, and Matlab files.[35] Three other GitHub projects have been created as part of a lecture series in which the results in [88] were reproduced.[36] Furthermore, a trained ANN model is published on Matlab Central [248] to evaluate the number of received molecules for a spherical reflecting transmitter and a spherical absorbing receiver using an ANN approach [62].

Matlab code is also developed to simulate the random walk and the signal reconstruction process at the receiver in free-diffusion channels [249], [250]. Introducing the mobile human ad hoc network (MoHANET) concept in which MC principles and analysis are applied to pathogen-laden droplets, code developments also encompass airborne pathogen transmission, integrating insights from epidemiology, biology, medicine, and fluid dynamics [251], [252]. These code files represent proof-of-concept results validated using empirical COVID-19 data from [253]. Finally, the published code also accounts for the modeling of a stochastic biofilm formation model based on bacterial QS [254], [255], [256], [257].

The code in [258] includes an NN model to realize a signal sequence detector for a mobile MC system based on the Informer model in [99]. Considering a diffusion-based environment, the mobile MC system comprises a point transmitter and a spherical passive receiver [99]. The signal

[34]MCFormer's Python code and dataset are provided in the GitHub repository in [233].

[35]This is the case in [244], which models a free-diffusion MC channel and implements an NN-based detector.

[36]These code projects are (i) Sangani [245], (ii) Patel [246] (also improved the code in terms of the CIR of a passive receiver or enzymatic degradation), and (iii) Shastri in [247].

sequence detector computes the autocorrelation coefficient of the input sequence to determine the optimal sequence length. Numerically obtained results in [99] demonstrate that the performance of the Informer-based model is better than that of the Transformer-based model in terms of detection ability. However, the dataset is not publicly available.

In [259], a "model for generating synthetic data for a biological MC" for training an NN for the discrimination of transmitted bits is presented. Testbed measurement data from [260] is used as a benchmark for the synthetically generated data. However, the dataset is not publicly available but will be made available upon request.

C. Experimental Data Generation

Experimental data generation from testbeds can generally be distinguished into air-based [116], [261] and liquid-based [262], [263], [264], [265], [266], [267], [268], [269], [270], [271], [272] data. As listed in [47], there are more testbeds than those mentioned here, but, to the best of the authors' knowledge, no data were made publicly available for other testbeds.

The datasets published on IEEE Dataport [262] and Zenodo [263] include fluorescence signal measurements (emission wavelength and intensities) of MoSK transmissions in a liquid-based testbed, which was proposed in the related publication in [273]. In the testbed setup, graphene quantum dots (GQDs), soluble in water and fluorescent, serves as a signaling molecule. The transmitter infuses the GQDs using injection valves, and a fluorescence-based receiver detects and decodes the fluorescence signals from blue-GQDs and cyan-GQDs, which serve as the molecules to shift. The testbed's performance is assessed based on synchronization, detection thresholds, and symbol recognition through a principal component analysis (PCA), which requires a broad dataset.

The dataset in [116] provides experimental measurements from a macroscale, single-input, single-output, air-based MC testbed. It also includes the Matlab processing code of the dataset-related publication in [212]. The dataset has been reported in various research studies in the literature, including XAI in MC channels [212], RL-based synchronization mechanisms [80], [82], and adaptive detectors [81]. The dataset in [116] was extended as a result of the research in [274]. The new associated repository, accessible in [261], records the experimental measurement data for ethanol molecules in the air. It also provides data sheets of the testbed components and the Python code for controlling the sprayer, reading the sensor output, and implementing communication protocols.

The dataset in [264] publishes a biocompatible MC testbed that utilizes magnetic nanoparticles as information carriers, so-called superparamagnetic iron oxide nanoparticless (SPIONs). The SPIONs are injected into a constant background flow using an injection pump, cf. [275, Fig. 1]. The receiver detects the SPIONs by a change in the inductance of the nearby fluid caused by the presence of the SPIONs. This testbed is utilized to investigate channel parameters such as (among others) background flow, channel length, and channel diameter. It should be emphasized that [275] discusses the dataset in detail and explains its structure, which is a limitation in other published datasets.

The received sequence of interfacial shift keying (ISK) transmissions in a liquid-based testbed is accessible on IEEE DataPort [265] and Zenodo [266]. In ISK, the modulation of the signal exploits the effect of viscosity fingering, i.e., two miscible fluids form a (temporary) interface, given that they differ in either their viscosity or density [276]. The testbed for experimental evaluation consists of a transmitter containing an infusion pump, a six-way injection valve, and a 10-way selection valve, which allows the injection of up to ten solutions [276, Fig 1]. Fluorescent carbon nanoparticles are used as information carriers. On the receiver side, a fluorescence detector measures the system's fluorescence output, demodulating and decoding the transmitted signal.

Images obtained from the particle image velocimetry (PIV) and planar laser-induced fluorescence (PLIF) tools, which are referred to for tracking and detecting fluorescent tracers in liquid, are accessible in [267], [268]. The dataset results from the research in [277], which develops two methodologies for particle tracking and detection in liquids. Both methods are based on a laser sheet illuminating a planar section of the medium [277, Figs. 2 and 3], where fluorescent tracers, captured by a camera, serve as information carriers. The repository also includes Matlab code for further image processing to track and detect the fluorescent tracers on the obtained camera snapshots.

The open-access dataset in [269] contains experimental measurements from a biological MC testbed [278]. The testbed utilizes *Escherichia coli* bacteria, which "express the light-driven proton pump *gloeorhodopsin* from *Gloeobacter violaceus*." Stimulating the bacteria by external light, the bacteria act as a transmitter and release protons into a liquid channel. The protons serve as signaling molecules, changing the pH value in the system, which is later detected. The repository also provides a detailed description of the dataset and comprises two zip files containing the data and a Matlab code example for processing it.

In [158], a liquid-based microfluidic MC testbed is presented. The source files for [158] are accessible in [270], [279]. For transmitting the information, CSK is used; in particular, information is encoded in the concentration of sodium hydroxide in that testbed. In addition, the testbed design accounts for the chemical reactions in the channel and the microfluidic geometry such that the transmitter can shape the signal to be transmitted while the receiver can threshold, amplify, and detect the sent chemical signal after the propagation through the channel [158]. Therefore, the chemical reactions are based on well-known pH-based reactants, such as hydrochloric acid and sodium chloride [158]. A phosphate-buffered saline solution is used for dilutions, and a spectrometer on the receiver side detects the transmitted information. The raw data (mainly as csv files) in [270] refers to the plotted results in [158] as well as in the appendix of the main paper. The GitHub repository in [279] contains the software for the complete automation of the testbed, including timed chemical injection using syringe pumps, measurement of the flow rate, and control of the flow rate using a proportional–integral—derivative controller, and measurement of the UV-visible spectrum.

The freely accessible dataset in [272] contains experimental measurements from long-term experiments using the closed-

loop MC testbed described in [280]. The experiment was run for 125 hours, obtaining more than 250 kbit of transmitted data via MC. The testbed utilizes a green fluorescent protein variant "Dreiklang" (GFPD), as the information carrier. Using light of a specific wavelength, GFPD can be switched reversibly between two different states [281]. The testbed differs from other liquid-based testbeds because it is a closed-loop structure, not an end-to-end structure, e.g., by pumping a liquid from one reservoir to another. Therefore, GFPD is only injected once. The testbed contains an optical transmitter for writing information, an optical eraser for erasing information, and a receiver for reading the fluorescence state of the GFPD [280]. In [272], the raw measurement dataset (as a zip folder containing csv and json files and as a SQLite database) is published, related to the corresponding publication plots in [282]. In addition, the Python code for synchronization and detection is published in a GitHub repository [271], which provides instructions on how to read the SQLite database.

D. Discussion

The datasets mentioned above exhibit different levels of accessibility, completeness, and documentation. As a reader's guide, we develop a traffic-light system in Table I in which green represents the highest level of dataset completeness and red represents the least along the following dimensions:

- Reproducibility (only considered for synthetic datasets): Green dots indicate that the source code and its documentation are fully accessible. Yellow dots indicate that the dataset documentation is missing or that access to the source code is unavailable. The red dots indicate that all of the above are missing.
- Representativeness (only for experimental datasets): Evaluated by comparing the number of replicates (N) of single experiments to the maximum number of repetitions over all experimental datasets. For the considered set of experimental datasets, a maximum number of $N = 50$ replicates applies so that for the ranges between $N = 1$ and $N = 17$ replicates a red dot, between $N = 18$ and $N = 33$ replicates a yellow dot, and between $N = 34$ and $N = 50$ replicates a green dot was assigned. The dataset in [271], [272] is an exception here, as it is the first long-term experimental MC system of its kind. The dataset size clearly stands out from the other datasets and is consequently rated green.
- Usability: Encompasses the cases "the dataset is available", "the dataset processing code is available", and "the code for plotting is available". If all three criteria are met, green follows; for two criteria (regardless of which), yellow follows; otherwise, red is applied.
- Availability: Here, we consider only two cases, either yellow (data record restricted availability, for example, behind an account wall) or green (data record freely available). Datasets that are not available are not listed, so no red is assigned here.
- Documentation: Considers the completeness of the parameters' metadata and the dataset's documentation. *Green:* Both aspects are fulfilled; *Yellow:* Only one of them is fulfilled; *Red:* None of them are fulfilled.

TABLE I: 4D dataset clustering for synthetic and experimental data.

Reference	Reproducibility	Representativeness	Usability	Availability	Documentation
[222]	●	-	●	●	●
[262], [263]	-	●	●	●	●
[216]	●	-	●	●	●
[261]	-	●	●	●	●
[116]	-	●	●	●	●
[264]	-	●	●	●	●
[265], [266]	-	●	●	●	●
[267], [268]	-	●	●	●	●
[269]	-	●	●	●	●
[228]	●	-	●	●	●
[270]	-	●	●	●	●
[223]	●	-	●	●	●
[231]	●	-	●	●	●
[283]	●	-	●	●	●
[284]		●	●	●	●
[226]	●	-	●	●	●
[233]	●	-	●	●	●
[272], [282]	-	●	●	●	●

Available datasets for MC research are diverse and accessible through multiple platforms, cf. appendix, Table V; however, several limitations hinder a broad use, cf. Table I. One limitation is the lack of standardization in data formats and annotations, which complicates integration and comparative analysis across datasets. Additionally, some datasets lack detailed metadata or related publications, reducing the transparency and reproducibility of the research outcomes, cf. Table I. The sizes of certain datasets are disproportionately small, limiting their applicability for machine learning or extensive simulation studies. Another issue is platform dependence. While IEEE DataPort, GitHub, and Zenodo are widely used, access to some university-hosted datasets may be restricted or not well-documented.

For the data generation tools, one limitation is compatibility, as many depend on specific software frameworks (e.g., Matlab or NS3), which can introduce compatibility issues when replicating results across different versions. Additionally, the assumptions inherent in a simulator's modeling approach constrain the representativeness of the datasets it generates. For example, tools like Smoldyn and MesoRD are optimized for reaction-diffusion systems but may fail to capture complex environmental heterogeneities. Similarly, simulators like nanoNS3 or N3Sim, although effective for nanoscale communication, may need to generalize more effectively to macroscopic systems. Availability is also an issue, as some tools have restricted access, e.g., requiring licenses for Matlab or COMSOL, outdated links, or lack of active maintenance, thereby reducing their reliability for long-term use. Finally, documentation quality varies widely. Some simulators offer less guidance to novice users than others, making it difficult for new users to adopt and effectively utilize them.

E. Concluding Remarks and Outlook

To summarize, in recent years, data generation, documentation, and publication have received attention, addressing the need for robust training datasets and reliable research outcomes. Our survey of existing datasets also shows that eight datasets are derived from experimental testbeds. The use case for sharing experimental data is stronger than that for synthetic data, primarily because wet lab experiments are more challenging to reproduce than simulations. Additionally, dataset publication

still requires more extensive and comprehensive documentation practices. Recommendations, e.g., as published in [285], are in the direction of standardizing dataset publication but still require a broader practice within the research community.

We also identify further research directions towards the connections between testbeds and the training and deployment of NNs as follows.

1) Training in Body-like Testbeds: The pharmaceutical industry is today looking closer to experimental results developed in organs-on-chip; see [286]. As a proven tool for drug testing [287], this technology mimics the operation of organs and the dynamics of the fluids. As these devices are expected to become a daily occurrence in laboratories, we expect them to be used as testbeds to generate data and integrate NN-models within them.

2) Transferring from Simulators to Real-World Testbeds: Following appendix, Table VI, a broad range of simulation tools exists for MC. On the one hand, these simulation tools are often open-source and easy to install, so researchers use them to implement their NN/ML approaches and benchmark them against analytical approaches, such as in [134]. On the other hand, the components of experimental testbeds are often limited in their computational capabilities, as seen in the testbed presented in [81], making the implementation of computationally intensive NN and ML approaches challenging. However, as the MC community moves towards practical applications, NN and ML tools applied to MC simulation tools should be transferred to experimental testbeds, enabling benchmarking, at least in pre-defined real-world scenarios, due to the targeted parameter control of experimental testbed setups.

VII. Conclusion

This survey examines the variety of NN architectures for the functional development of IoBNT networks, provisioning with the most recent methods towards their integration in MC links. A thorough evaluation of reported research concludes that NN architectures become essential for handling the complex nature of MC connections, where closed-form expressions for model-based solutions are frequently impractical to develop. As such, NN are the *de facto* model for devising practical solutions within the various layers of IoBNT networks. A closer look at the reported literature reveals that most of the reported developments primarily focus on the PHY layer for decoding tasks, where the BiRNN achieves the highest performance in ISI channels and with less complexity compared to RNNs, feedforward NNs, and CNNs. Conversely, fewer works target solutions in the upper layers, and only a few methods explore the much-needed localization and detection applications in IoBNT networks. We expect more developments of NN-based methods towards application-oriented solutions of IoBNT networks. This survey also outlines existing datasets for training and testing NN modules, as well as tools for their generation, both synthetically and experimentally. The literature is maturing in supplying the necessary datasets for NN training; however, we anticipate further progress in the usability and reproducibility dimension of datasets, along with increased data generation for in-body environments.

Furthermore, studies in the interconnected areas of NN and IoBNT underline long-term research topics: Biocompatible deployments of NNs at the nanoscale are still in their early phases, nanoscale architectures beyond feedforward NNs also demand consideration to realize the IoBNT's full potential, and physics-informed architectures, grounded in domain knowledge, can be utilized to develop self-explainable solutions. We conclude with a thought-provoking remark: AI can be a common ground between biology and engineering to facilitate interdisciplinary approaches. AI methods can serve as a common framework that converges the two disciplines of biology and engineering.

Appendix

This material provides a brief background on various topics related to NN architectures, training, and databases, supporting the primary document. The NN architecture description primarily supports the evaluation of its complexity, as discussed in Section III of the primary document. To that end, we present a detailed evaluation in this material, presented in a table format, which conveys performance in Table II and complexity in Table III. The entries in this table are also anchored in the MC geometry, architecture, and application scenario. We also provide more details on the guidelines for rendering and maintaining databases for training NN modules. We also include Tables V and VI, which list the datasets and simulators used for the synthetic generation of data. Finally, we also describe cell-to-cell MC links from a biological perspective, which supports the illustrative code examples in Section III.A.4 and V.D. The cited references in this paper are printed in the primary document.

To support the reading of the primary document, we print below a list, in alphabetical order, of the abbreviations included within the paper.

AAEC	asymmetric autoencoder
AEC	autoencoder
AI	artificial intelligence
ANN	artificial neural network
ARX	auto-regressive exogenous
ASK	amplitude shift keying
BCE	binary cross entropy
BER	bit error rate
BiRNN	bidirectional recurrent neural network
BVS	BloodVoyagerS
CFD	computational fluid dynamics
CIR	channel impulse response
CNN	convolutional neural network
CRN	chemical reaction networks
CSK	concentration shift keying
CUDA	compute unified device architecture
DeepLIFT	deep learning important features
DNA	deoxyribonucleic acid
DNN	deep neural network
DRL	deep reinforcement learning
DSP	digital signal processing
FBMLE	filter-based maximum likelihood estimation

FC	fully connected
FIR	finite impulse response
FPR	false positive rate
GA	genetic algorithm
GNN	graph neural network
GPU	graphical processing unit
GQD	graphene quantum dot
GRN	gene regulatory network
GRU	gated recurrent unit
HCS	human circulatory system
ICE	individual conditional expectation
IIR	infinite impulse response
ILI	inter-link interference
IoBNT	Internet of Bio-Nano Things
ISI	inter-symbol interference
ISK	interfacial shift keying
ITR	interference-to-total received molecule ratio
LIME	local interpretable model-agnostic explanation
LRP	layer-wise relevance propagation
LSTM	long short-term memory
MAC	medium access control
MAI	multiple-access interference
MAP	maximum a posteriori
MC	molecular communication
MIMO	multiple-input multiple-output
MIP	manual permutation importance
MISO	multiple-input single-output
ML	machine learning
MLE	maximum likelihood estimation
MLP	multilayer perceptron
MM	media modulation
MNIST	modified national institue of standards and technology
MoHANET	mobile human ad hoc network
MoSK	molecule shift keying
MPI	manual permutation importance
MSE	mean square error
NN	neural network
OOK	on-off keying
PBS	particle-based simulation
PCA	principal component analysis
PDE	partial differential equation
PDP	partial dependence plot
PHY	physical layer
PINN	physics-informed neural network
PIV	particle image velocimetry
PLIF	planar laser-induced fluorescence
PPO	proximal policy optimization
PSO	particle swarm optimization
QS	quorum sensing
ReLU	rectified linear unit
RF	radio frequency
RL	reinforcement learning
RMSE	root mean squared error
RMSprop	root mean square propagation
RNA	ribonucleic acid
RNN	recurrent neural network
SGD	stochastic gradient descent

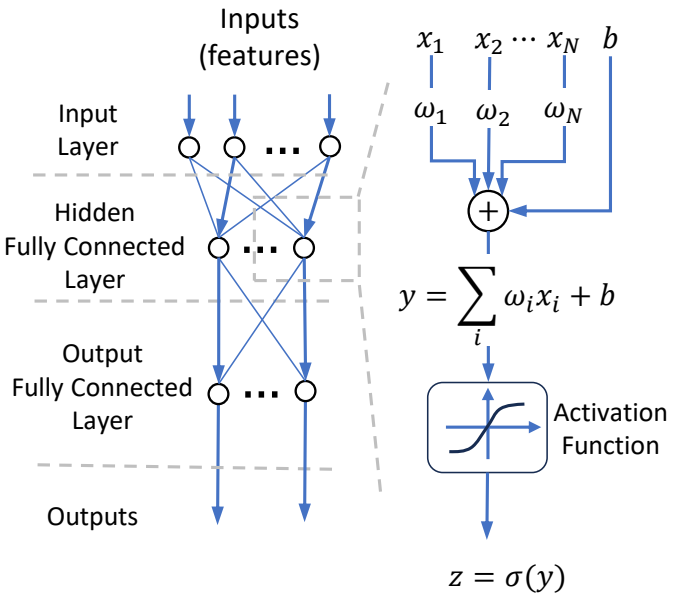


Fig. 15: Architecture of a feedforward NN.

SHAP	shapley additive explanation
SNR	signal-to-noise ratio
SPION	superparamagnetic iron oxide nanoparticles
STO	symbol time offset
TCN	temporal convolutional neural network
TPR	true positive rate
XAI	explainable artificial intelligence

Deep NNs in MC schemes have been employed in various architectures, including feedforward NN, BiRNN with LSTM cells, Autoencoders, and Transformers. This section provides a brief overview of these architectures, offering insights into their applications in MC channels.

A. Feedforward NNs as a Universal Approximator

This architecture is the most information-agnostic one used in communication. Due to its universal approximation, it has been primarily reported for channel estimation rather than decoding; see Sec. III.A in the primary document. The feedforward deep NN architecture consists of fully connected layers in cascade, as represented in Fig. 15. This architecture follows the pioneering work in [288], which describes the calculus running in the nervous system. The fully connected layer produces an arbitrary number of outputs, equal to the number of deployed nodes. These outputs are straightforwardly connected to the inputs of the following fully connected layer. In its simplest form, a deep NN consists of two fully connected layers, referred to as the hidden and output layers.

On each layer, each node operates independently of neighbor nodes and transforms the inputs through a linear combination; see the weighted sum yielding y in Fig. 15, and a non-linear calculation through a so-called activation function of y. Each neuron encompasses as many coefficients as inputs into the layer, providing the flexibility to approximate a given output sequence from the inputs. Examples of activation functions are the sigmoid, ReLU, and the $\tanh$ function (see [289, Sec. 3.4]). These two operations (linear combination and non-linear

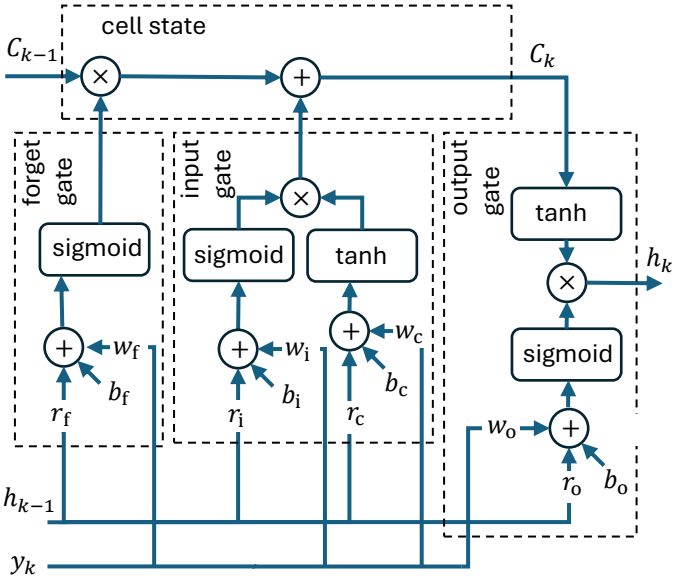


Fig. 16: Architecture of the LSTM unit.

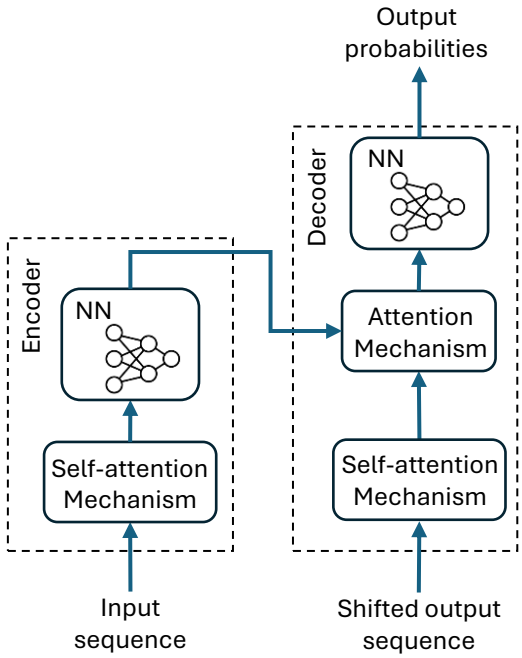


Fig. 17: Transformers architecture (reduced schematic) [110].

activation function) entail deep NNs as a universal approximator. The weighted sum accounts for the linear transformation of the input set, encompassing operations such as translation, rotation, and contraction. The non-linear activation function accounts for operations like transforming axes and boundaries into arbitrary shapes.

B. Accounting for Data Relevancy within LSTM Networks

This architecture is the most reported one in MC channels for decoding. The LSTM architecture follows a more complex structure. Cells are interconnected bidirectionally to decode symbols, leveraging on previous and future time slots. This architecture implements three different gates to control the relevancy of the input data in the calculation of the cell's output.[37] The input data to the unit cell consists not only of the raw data (y_k) and hidden state (h_{k-1}), but also the previous cell state (C_{k-1}), which accounts for the data relevancy in the hidden state. This is a form of a focus mechanism to highlight the most relevant content of the cell state.

The cell state is realized with a multiplier and an adder. The multiplier filters the previous cell state (C_k) with the calculated amount to forget, as determined by the forget gate. The adder accounts for including the new information from the inputs h_{k-1} and y_k, as determined by the input gate. The amount to forget is implemented with the coefficient w_f and the sigmoid function (output ranging between 0 and 1). In contrast, the amount to add is implemented with the hyperbolic tangent function ($\tanh$), where the output ranges between -1 and 1, and using the coefficient w_c. The input gate also uses another forget gate, but through a different coefficient (w_c).

[37]Details on LSTM cells are given in the Olah's blog, accessible in http://colah.github.io/posts/2015-08-Understanding-LSTMs and also within the diagram in the link https://ch.mathworks.com/help/deeplearning/ref/nnet.cnn.layer.lstmlayer.html#mw_9f7c5f93-4bf2-4ddb-b922-b1c122668b9a_sep_mw_8cad7fee-7610-4134-9354-b7ed3a45204d_head

C. Attention in the Loop with the Transformer Architecture

The Transformer architecture performs exceptionally well in language translation and generation. As the major distinction, this architecture overcomes the sequential processing of LSTM networks,[38] and forwards parallel capabilities by getting rid of recurrent structures as previous standard developments as in [115], [290], [291], [292].[39] Besides, the Transformer architecture integrates attention models within two feedforwards NNs, those for encoding and decoding[110]; see a reduced schematic representation in Fig. 17.[40] The attention model evaluates a context vector to support the decoder component in predicting the most likely next symbol, a concept under development in [115], [294], [295], [296], [297], which has been shown to significantly enhance the alignment capabilities of neural machine translators.[41]

The encoder and decoder in the Transformer architecture, as shown in Fig. 17, implement a self-attention mechanism using the scaled dot product operation. This architecture's second significant distinction is realized as a more efficient mechanism than the implementation with additive attention; see [298, Sec. 4.5]. The self-attention mechanism evaluates the product of matrices of queries, keys, and values (see [110, Eq. (1)]). These matrices are solely based on the input sequence and a pre-multiplicative learned matrix, adjusted during training.[42] Additionally, the decoder employs an attention mechanism,

[38]LSTM completes the processing sequentially due to the cascade connection of cells like the ones in Fig. 16.

[39]See a perspective introduction to main core development for the Transformer architecture in the interview [293].

[40]We remark that a similar direction to remove recurrent structures is given in [294] but using the CNN architecture instead.

[41]Although we refer to symbols in the context of MC channel, the attention model in [115] and the Transformer architecture in [110] are developed as language translators and symbols are instead referring to words.

[42]The self-attention mechanism differs from the recursive architecture, where these matrices are calculated based on the LSTM hidden states.

where the query is evaluated based on the input sequence, while the keys and values are evaluated based on the output sequence. While the self-attention block extracts information from the input sequence, the attention block produces its output using what is relevant within the encoder's output. Although this architecture discards RNNs, inherently, the attention model is a form of recurrence, as its output is used to evaluate the internal coefficients.

D. Reinforcement Learning: Smartly Actuating on the Environment

While the traditional ML approaches, i.e., supervised and unsupervised learning, detect a type in the data set presented to them, Reinforcement Learning optimizes its decisions via trial-and-error interaction with an environment without the need for an a priori data set [299]. This approach can be used to actuate the communication scheme parameters, such as the threshold for detecting incoming symbols. RL in general consists of an RL agent and an attached environment. The environment is typically defined as a (partially observable) Markov decision process consisting of

- a set of states S,
- a set of actions A,
- a distribution of initial states $p(s_0)$,
- a reward function $r : S \times A \to R$,
- transition probabilities $p(s_t + 1|s_t, a_t)$,
- termination probabilities $T(s_t, a_t)$, and
- a discount factor gamma element $[0, 1]$ [87].

The agent optimizes a policy function that maps the state to a distribution over actions. Given an environment state s_t, the agent acts a_t according to its policy function and is given a reward based on the result of the applied action. During the training, the policy function is optimized to maximize the expected future rewards [87].

The RL agent learns the impact of its actions, their causes, and effects, within the environment in steps, imitating the natural human learning process [300]. The learning process itself is supported by two different functions, the policy and the value functions. The policy determines how the agent behaves at each new step, while the value function estimates the goodness of a state in terms of the expected reward. The networks used to approximate the two functions can also employ different types of DNNs, such as ANNs, CNN, or RNNs. With the inclusion of these types of network layers, RL then becomes DRL [299]. The type of network used depends on the task the RL agent needs to solve.

E. Training a Neural Network

NNs usually represents a type of supervised learning model, where the correct output for each input observation is known during training. In other words, similar to all other supervised learning models, NNs are also trained based on labelled datasets, and try to learn how to generate an estimate of the true outputs during the training process. The process of training a NN involves adjusting the trainable parameters of all the layers to make accurate predictions based on the input data. This process can be done in three main steps [301]:

- Defining the objective: The network is trained using a dataset where each input has a known correct output. The goal is to make the network's predictions as close as possible to these true outputs. A loss function is used to measure the discrepancy between the predictions and the actual values. This function calculates the difference between the network's predictions and the true outputs, giving a single number that represents the model's performance. The ultimate goal is to minimize the chosen loss function, with mean squared error and cross-entropy being among the most widely used loss functions in training NNs.
- Defining the optimization method: An optimization algorithm, such as gradient descent, is used to find the network's parameters that minimize the defined loss function. This method systematically adjusts the network's parameters to minimize the loss function. gradient descent relies on determining how each parameter influences the loss, a process known as calculating the gradient.
- Initializing the network's parameters: The network's parameters must be initialized before the training begins. Proper initialization is crucial for ensuring stable and efficient training. Common strategies include random initialization, where weights are assigned small random values.
- Updating the network's parameters: gradient descent relies on the gradient of the loss function, a vector of partial derivatives that quantifies how the loss changes with respect to each network parameter. NNs often consists of multiple layers, making it challenging to trace how a parameter in an early layer affects the final loss. This is where backpropagation comes in. Backpropagation is a robust algorithm that calculates these influences layer by layer, starting from the output and moving backwards. It ensures that all network parameters are updated appropriately to improve the model's prediction accuracy.

F. Training an Autoencoder

Autoencoders follow a similar training process to standard NNs but are distinct in their objective and structure. Autoencoders are unlike traditional supervised learning models, which map inputs to predefined target outputs. Autoencoders aim to learn an efficient representation of the input data by encoding it into a lower-dimensional latent space and reconstructing it back to its original form.

Unlike supervised learning, which aims to predict external labels, an Autoencoders is trained to minimize the difference between its input and output, where the loss function measures reconstruction error. Similar to supervised learning, commonly used loss functions for training Autoencoders are MSE or binary cross entropy (BCE), with BCE being the most relevant in communication system design.

Analogous to supervised learning, optimization algorithms like gradient descent are used to find Autoencoders 's parameters; however, unlike supervised learning, the gradients must propagate back to the encoder, requiring all the layers

between the encoder and decoder, including the channel, to be differentiable. If such a differentiable channel model is available, it can be incorporated as non-trainable layers for end-to-end training [302].

Real-world channels are often non-differentiable or even unknown, making direct training challenging and leading to performance losses when relying on inaccurate models. Another workaround involves training Autoencoders on an inaccurate but differentiable channel model and then fine-tuning the decoder using real measurements, yet this limits encoder optimization. Alternative approaches include learning a differentiable channel model using an NN which mimics the channel [303] or employing iterative training methods, where the decoder is updated with true gradients while the transmitter is trained using approximated gradients [304].

G. NN's Hyperparameters

Hyperparameters refer to those components of the NN whose numerical values can not be derived from the dataset, as they define the architecture per se. Consequently, they are not part of the training process [141]. These are components related to

- the architecture, such as the number of nodes and hidden layers, drop-out rate,
- the selection of the loss function type, such as binary cross-entropy, multiclass cross-entropy, root mean square propagation (RMSprop),
- the selection of the activation functions, such as sigmoid, rectified linear unit (ReLU), 'softmax', 'tanh', 'softsign',
- the training parameters such as the batch size [101], learning rate [305], momentum [306], number of epochs), or
- the selection of the optimizer, such as stochastic gradient descent (SGD), Adam, RMSprop.

These are parameters set by experience and by validating the operation of the NN.

A rule of thumb to manually set these parameters is to tune them during the training and validation phases; see [307, Sec. 5.3]. The dataset can be divided into 80 % for training, 10 % for validation, and 10 % for testing. The NN is trained with the given set of hyperparameters and adjusted till verifying performance within the validation test. If good performance is also obtained within the testing set, then this is a good candidate for the NN hyperparameters.

The hyperparameters can also be automatically tuned, and various tuning methods exist in the literature. Examples of implemented algorithms include the grid search (a brute-force approach) and random search (a less computationally costly approach). More elaborate algorithms, such as the popular SGD or Bayesian optimization, exploit the impact of hyperparameter variability on NN performance; see further details in [141].

H. Deployment of NNs in MC Environments

The above-mentioned NNs architectures enable communication links in MC environments such as free diffusion, drifted channels, and within experimental testbeds. Table II summarizes the deployed NNs architectures per MC environment and application. In free-diffusion environments, the reported literature refers to receiver size and communication range in the cell scale, i.e., operating in the $\mu\mathrm{m}$ range (mostly less than $10\,\mu\mathrm{m}$). Besides, the literature reports particles with relatively fast mobility in free diffusion channels if we consider the value of the diffusion coefficient, D-column in Table II.[43] To have a point of comparison, the diffusion coefficient of vesicle molecules, as exchanged by cells in the human tissues, is in the order of $10^{-4}\,\mathrm{nm}^2/\mathrm{ns}$ (see [75]), while most of the particles reflected in the literature are larger than $10^{-1}\,\mathrm{nm}^2/\mathrm{ns}$.[44] The communication range for deployed NNs increases within experimental testbeds and the human vessels in the meter range.

The deployed NNs apply in various applications such as channel estimation, synchronization, data communication, and detection. For this application, the second right half of Table II summarizes communication-related parameters, including the number of released molecules, symbol duration, SNR ranges, and performance metrics. In free-diffusion channels, assumptions about the number of released particles (over thousands) are quite large compared to the cell-to-cell natural process, where the number of vesicles exchanged is typically below a hundred, as seen in the work in [75]. For data communication, the achieved BER is quite favorable, most of the reported values are in the 10^{-1} to 10^{-4} range when the SNR ranges 0 to $10\,\mathrm{dB}$.

We also include a second table in Table III that summarizes the complexity of the proposed architectures. We list the number of nodes per fully connected layer, RNN layers (which develop LSTM cells), and within the CNN layer. These calculations are accomplished as follows:

- Feedforward NNs: We compute the total of parameters by adding the number of coefficients and bias per layer. Per layer, the total of coefficients is $n_{\mathrm{in}} \times n_{\mathrm{out}}$, and the total of biases is n_{out}, where n_{in} is the number of inputs, n_{out} is the number of outputs for the given layer. The number of nodes in the output layer is one, except for the entry in [90], where the nodes in the output layer are two.
- RNN and BiRNN architectures: We compute the number of parameters according to the gates within the LSTM cells, as all the reported methods deploy those. The LSTM cell comprises four gates: input, forget, cell, and output; see Fig. 16. Each gate implements a separate set of weights for the input, hidden states, and biases, with eight weights and four biases, totaling 12 parameters per cell. In total, the amount of parameters for n_{cells} cells will be $n_{\mathrm{cells}} \times (12)$. In the case of the BiRNN, calculations are twice the number as for the RNN, as each layer implements both a backward and forward direction.
- CNN architecture: The filter size gives the number of parameters. A filter of size K will implement K weights and a single bias. Additionally, the total number of filters is determined by the comparative sizes of the input and output. For instance, if the input is a 1D vector as 128×1

[43]The diffusion coefficient reflects the variability of the position of the particle with time. See its definition in [215, page 10].

[44]Diffusion coefficients in the units of $\mathrm{nm}^2/\mathrm{ns}$ refer to molecules of the size of Potassium atoms while diffusing in water, see [308, Sec. V]

and the output is a 2D matrix as 128×64, the number of filters is 64. As another example, if the input and output are 2D matrices given by 64×64 and 64×128, then two filters would be needed as the dimension of the output is twice in the rows than the dimension in the input.

- Transformer architecture: The complexity for this architecture is evaluated based on the code published by the authors in [99].[45] In this entry, the number of nodes per hidden layer, as 200, refers to the six fully connected layers implemented by the transformer architecture, while the other amounts, 5 and 1, refer to the remaining fully connected layers implemented within the decoder. This architecture also deploys three CNN placed at the input of the Transformer for feature extraction.

Communication among cells is a crucial mechanism for their survival and creation. In the process, an animal cell produces a molecule that is later detected by a receptor protein at a target cell, triggering a chain of intracellular signals. Within the cell's interior, a signaling cascade occurs, targeting the activation of metabolic enzymes, gene expression, or changes to the cytoskeleton. Communication among cells occurs on a long range, such as endocrine cells releasing hormone molecules into the bloodstream, and on a short range through diffusion in the extracellular medium, known as paracrine signaling; see [309, Chap. 15].

Examples of molecules include proteins, peptides, amino acids, and hormones, which, upon detection at the target cell, produce receptors that relay signals into the cellular interior. The metabolism of these proteins directs processes such as shape, movement, or gene expression, for instance. A cell typically possesses a few dozen receptors to distinguish specific molecules from thousands of possibilities. These receptors are located in the cell membrane for large extracellular molecules that are too large or too hydrophilic to cross the membrane, or in the interior of the cell for molecules (like steroid and thyroid hormones) that can traverse the cell membrane. Examples of receptors inside the cell are those that activate gene expression in response to cortisol, a process that takes many minutes or hours to produce a response.

Examples of receptors in the cell membrane (the vast majority) are of three kinds: ion-channels-linked, G-protein-linked, and enzyme-linked. Their main distinction is related to the intracellular signal they produce. The ion-channel-linked receptors produce an electrical effect through a flow of ions. G-protein-linked receptors produce a protein that diffuses into the plasma membrane, whereas enzyme-linked receptors produce a cascade of signals that travel to the cell's interior. These three receptor types may detect different incoming extracellular molecules; they are not specialized in a single type and depend on cell specialization. For instance, a G-linked receptor detects acetylcholine signaling molecules in a heart muscle, while an ion-channel-linked receptor detects the same molecule in skeletal muscle cells; both receptors trigger different responses.

In this section, we list recommended practices for preparing datasets, as introduced in Section VI of the primary document. The following "rules for the care and feeding of scientific data" are published by Goodman *et al.* [310] and modified for this work.

1) **Sharing is Caring:** The first rule is the most important - publish the dataset and convince other researchers to publish their datasets. Furthermore, request a publicly available dataset, following the subsequent rules when reviewing papers.
2) **Making the Data Set Publicly Available:** Share and provide a dataset as early as possible. Store the dataset in a well-known, easily accessible, and long-lasting data archive, ideally tagging it with a digital object identifier (DOI). Following Goodman *et al.* [310], a proper data archive contains *(i)* a DOI, *(ii)* a documentation of the dataset, as well as metadata, and *(ii)* a "good curation practice".
3) **Thinking About Re-Use:** While preparing the dataset and corresponding documentation, remember the intended re-use. Prepare the dataset and documentation accordingly, using standard formats, or ensure easy access to the dataset. Also, track the versions of the dataset.
4) **Publishing the Workflow:** Include at least a description of the data flow showing how data (intermediate and final) is generated. Consider encapsulating your workflow for more complex scenarios, such as using an online service.
5) **Linking the Publications:** Link the dataset in your publications as a standard reference (including DOI) and link the publication in your dataset documentation.
6) **Publishing the Code:** Publish the code, if available, e.g., for simulators and plots.
7) **Getting Credit:** State how to acknowledge or publish the published dataset using a bib item or a license, respectively, in the documentation. Provide all necessary information for the intended acknowledgment.
8) **Targeting Data Storage:** Use the intended research community's standard repository to make the dataset publicly available.
9) **Rewarding Data Share:** Acknowledge researchers who share their data and/or code. Cite the datasets and show best practices. Engage the community by giving feedback on the quality of datasets.

An overview of existing datasets on MC, ordered by platform, can be found in Table V. Besides the reference for the dataset itself and the platform on which it is published, the properties year, size, dataset type, and the related research are listed. As datasets are also created using simulators, Table VI lists existing simulators alphabetically, along with their corresponding application areas. Furthermore, references to the published source code are provided for the reader.

[45]We completed the calculation for complexity analyzing the file `git_Transfomer` as accessible in https://github.com/Zhichao-Zhang-Zjut/Informer-based/tree/main

TABLE II: Summary of NN architectures, applications, and parameters related to the MC channel and communication.

			End-to-end channel-related parameters			Communication-related parameters					
MC Geometry	NNs arch.	Application	Comm. range	Receiver radius	D [nm^2/ns]	Released molecules	Symbol duration	SNR [dB]	Performance metrics		Ref.
Point transmitter-Free diffusion-Spherical absorbing receiver	Feedforward NNs	Channel est.	2 to 10 µm	4 µm	33×10^{-4}, 66×10^{-4}	200			Error	3.3 %	Sec. III.A.4
			2 to 11 µm	3 to 7 µm	5×10^{-1}, 10^{-1}	3×10^{3}			RMSE	≤ 0.3	[66]
			2 to 11 µm	4 to 10 µm						≤ 1 molecule	[62]
		Data comm.	500 nm	45 nm	4.2×10^{-1}		270 ms	−5 to 80	BER	4×10^{-1} to 2×10^{-2}	[100]
							450 ms	−5 to 35		5×10^{-1} to 2×10^{-5}	
			5 µm	50 nm	1.01	4×10^{4} to 4.5×10^{4}	100 ms	30 to 56		2×10^{-1} to 10^{-4}	[90], [96], [97]
			14 µm	1 µm	0.06	10^{3} and 2×10^{3}	3 s	2 to 14		3×10^{-1} to 2×10^{-5}	[91]
		Localization	10 µm	4 µm	0.79	10^{4}	2 and 5 s		Error	< 50 %	[131]
		Fusion data	4 to 9 µm	4 µm	5×10^{-1}, 7.9×10^{-1}		70 ms	2 to 5	P_d	>0.94	[135]
									P_{fa}	$\leq 7.5\times10^{-2}$	
	RNN		4 to 9 µm	4 µm	5×10^{-1}, 7.9×10^{-1}		70 ms	2 to 5	P_d	>0.94	[135]
									P_{fa}	$\leq 7.5\times10^{-2}$	
	Autoencoder	Data comm.	1 to 10 µm	10 µm	0.79	10^{3}			BER	5×10^{-2} to 10^{-6}	[118]
	BiRNN		1 to 50 µm		1	10^{4} to 5×10^{4}	10 ms	0 to 60		10^{-1} to 10^{-5}	[89]
	Transformer		7 to 12 µm	5 µm	5	6×10^{3} to 18×10^{3}	35 and 100 ms	0 to 40		2×10^{-1} to 4×10^{-3}	[94], [99]
Drifted channels			10 µm	1.5 µm	0.79	4×10^{3}	200 ms	10 to 40		10^{-1} to 4×10^{-3}	[114]
	Autoencoder		38 mm		1.24×10^{5}		100 to 600 ms	0 to 10		2×10^{-2} to 2×10^{-6}	[119]
	RNN	Synch.	40 cm		0.1	10^{4}	20 s	30	P_d	0.6	Sec. III.B.4
									STO	3.4 s	
Experimental testbeds: Vessel-like channels	Transformer	Data comm.	< 1 m				1.2 s		BER	8×10^{-2} to 9×10^{-2}	[102]
	BiRNN									7.4×10^{-2}	[105]
			< 1 m				250 to 500 ms			$< 10^{-4}$	[103], [104]
	CNN		5 cm				0.25, 0.5, 1 s		Accuracy	≥ 33 %	[106]
	Feedforward NN						0.5, 1, 2, 3 s		BER	0.19 to 0.4	[112]
Experimental testbeds: Open air		Channel est.	1 to 2 m				250, 500, 750 ms		RMSE	< 0.2	[63]
	BiRNN	Data comm.	1 m		0.84	4.9×10^{23}	1 to 6 s	27.5	BER	10^{-2} to 10^{-6}	Sec. III.C.4
Experimental testbeds: *in-vivo* bacteria colony	CNN		100 µm	6.7 µm	0.75	10^{5} bacteria	4 s	10 to 35		3×10^{-1} to 10^{-4}	[109]
							1 min			$< 10^{-2}$	[108]
Human vessels	Feedforward NN	Channel est.							RMSE	$< 5\times10^{-2}$	[64]
		Detection	≈ 2.5 m			10^{3}			Accuracy	< 85 %	[12]

Notes:
• In the Ref. column of the table, the entries referring to section are pointing to the section in the primary document.

- Missing entries in the tables are not specified in the corresponding reference.
- The column D Molecule refers to the diffusion coefficient of molecules.
- The column Released molecules refers to the number of molecules released at the emitter.
- Few contributions report the channel length. The work in [100] reports a channel length of 6 units. The work in [102] reports 3, 5, and 13. The work in [112] reports 1. Our solution in Sec. III.C.4 reports a channel length of 7.
- All reported Data comm. schemes uses the OOK modulation except for [106] which report multilevel modulation as well.
- The acronyms RMSE and FF NNs in the table refers to root mean squared error and feedforward neural networtk, respectively.
- The entry table for reference in [62] also develop a MC geometry comprised of a volumetric transmitter. Besides, we evaluated the RMSE metric by inspecting the printed Fig. 3 on the same paper.
- The flow velocity corresponding to the entry in the Table "Drifted channel" is $30\,\mu\mathrm{m/s}$ as follows from reference [114], $5.5\,\mathrm{cm/s}$ as in [119], and $10\,\mathrm{cm/s}$ as in the entry for Sec.III.B.4. Additionally, the entry table Open air also specifies a flow velocity of $3.5\,\mathrm{m/s}$ for Sec. III.C.4 in the primary document.
- The RNN and BiRNN entries in the table implement LSTM cells.
- The entry table for the references [90], [96], [97] refer to a mobile scenario where the diffusion coefficient of the emitter is $4.74 \times 10^{-5}\,\mathrm{nm^2/ns}$ and for the receiver is $2.31 \times 10^{-3}\,\mathrm{nm^2/ns}$. Similarly, the entry table for the reference in [94], [98], [99] define a diffusion coefficient of the receiver as $10^{-5}\,\mathrm{nm^2/ns}$.
- The length of the channel in the entry table corresponding to references [103], [104], [105] is evaluated according to the pictures printed in the papers.
- The number of released molecules refers to the nanosensor instead of the entry table with reference [12].

TABLE III: Complexity of reported NNs architectures, see Section H for details on the calculation of the number of parameters.

MC Geometry		NNs arch.	Application	Input length	Fully connected layers			RNN layers		CNN layers			Number of parameters	Ref.
					Number of FC layers	Number of hidden layers	Nodes per hidden layer	Number of layers	Number of LSTM cells per layers	Number of layers	Number of filters	Filter size		
Point transmitter-Free diffusion-Spherical absorbing receiver		Feedforward NNs	Channel est.	2	1	1	2						6	Sec. III.A.4
			Localization	10^3	1	5	400						1 041 603	[131]
			Detection	16	1	1	16						272	[135]
			Data comm.	6	5	10	4						837	[88]
				11	1	2	20						722	[96], [97]
				11	1	12	10						1242	[90]
				120	1	2	70 and 10						9111	[91]
		Transformer		12	8	1	200, 5, and 1			3	3	10, 3, and 1	15 688	[99]
		RNN	Detection	32	1	1	32	1	32				417	[136]
Experimental testbeds	Open air	BiRNN	Data comm.	8				6	8				576	Sec. III.C.4
	Vessel-like channels			5	2	1 and 1	25	2	5				540	[105]
				40				32	40				2880	[104]
		RNN	Synch.	1	2	1 and 1	256 and 256	1	128				35 328	Sec. III.B.4
		CNN	Data comm.	128	3	1	4096, 4096, and 6			3	64, 2, and 2	7, 5, and 3	33 587 738	[106]
in-vivo bacteria colony				$15 \times 60 \times 2$	2	1	32 and 2			1	16	15	462 144	[108]
				5						2	2	3	30	[109]
Human vessels		Feedforward NN	Detection	2	1	1	6						18	[12]
Drifted channels		Autoencoder	Data comm.	1	Enc: 3 Dec: 5	Enc: 1 Dec: 1							244	[119]

Notes:
- In the column "Ref." of the table, the entries referring to "Sec." point to the section in the main document.
- Missing entries in the table are not specified in the corresponding reference.
- The number of parameters in the table entries for [96], [97] are already given within these references.
- The RNN and BiRNN entries in the table implement LSTM cells.
- We assume output length as 1 for the entry in the table referring to [135], 2 for [88], [100], and 3 for [90].
- The column table labeled 'Number of parameters' refers to the total number of weights and biases.
- The column table labeled 'Number of layers' refers to the number of hidden layers in the feedforward NN architecture.

TABLE IV: Summary of NNs architectures, applications, and hyperparameters related to their training.

MC Geometry		NNs arch.	Application	Optimizer algorithm	Learning rate	Number of samples training	Number of epochs	Batch size	Ref.
Point transmitter-Free diffusion-Spherical absorbing receiver		Feedforward NNs	Channel est.	BFGS		481	15		Sec. III.A.4
			Localization	Adam	10^{-2}		100	256	[131]
			Detection		10^{-3}	10^5 samples	100	10	[135]
			Data comm.	LM	10^{-2}	10^3 bit	200	50	[100]
						5×10^4 bit		1×10^3	[88]
					10^{-3}	5×10^4 bit	50, 55, and 125		[96], [97]
				BFGS			184, 188, and 215		
						10^4 bit			[90]
					10^{-3}		500		[91]
		RNN	Detection	Gradient descent	10^{-3}	8×10^3 samples	100	10	[136]
Experimental testbeds	Open air	BiRNN	Data comm.	Adam	10^{-3}	10^6 samples	10	10	Sec. III.C.4
	Vessel-like channels					8×10^4 bit	2×10^4		[105]
					10^{-3}	120 bit	200	10	[104]
		RNN	Synch	BFGS	2×10^{-4}	1.2×10^6	35×10^3		Sec. III.B.4
		CNN	Data comm.	Adam	10^{-3}		20	64	[106]
in-vivo bacteria colony		CNN		SGDM					[108]
Human vessels		Feedforward NN	Detection				4×10^4		[12]
Drifted channels		Autoencoder	Data comm.		9×10^{-3}		2×10^3	40	[119]

Notes:
• In the Ref. column of the table, the entries referring to "Sec." are pointing to the corresponding section in the main document.
• The abbreviations LM, BFGS, and SGDM introduced in the column "Optimizer algorithm" refer to Lavenberg-Marquardt, Broyden-Fletcher-Goldfarb-Shanno (see [311]), and stochastic gradient descent with momentum.

TABLE V: Existing datasets on MC, ordered by platform.

	Name of the dataset	Reference	Year	Size	Data type	Related work	Platform
(1)	Dataset for Advanced Plaque Modeling for Atherosclerosis Detection using Molecular Communication	[222]	2024	∼1.5 GB	Synthetic	[312]	IEEE DataPort
(2)	Dataset of "Experimental Implementation of Molecule Shift Keying for Enhanced Molecular Communication"	[262], [263]	2023	∼300 MB	Experimental	[273]	IEEE DataPort [262], Zenodo [263]
(3)	Dataset for the Simulation of Microfluidic Molecular Communication using OpenFOAM	[216]	2023	∼40 GB	Synthetic	[156]	IEEE DataPort
(4)	Dataset for Analog Network Coding in Molecular Communications: A Practical Implementation	[261]	2023	∼0.8 MB	Experimental	[274]	IEEE DataPort
(5)	Dataset for Macroscale Molecular Communication Testbed	[116]	2023	∼0.2 MB	Experimental	[80], [81], [82], [212]	IEEE DataPort
(6)	Channel Parameter Studies with a Biocompatible Testbed for Molecular Communication	[264]	2023	∼2 MB	Experimental	[275]	IEEE DataPort
(7)	The Data Related to Interfacial Shift Keying Allows a High Information Rate in Molecular Communication	[265], [266]	2022	∼0.6 MB	Experimental	[276], [313]	IEEE DataPort [265], Zenodo [266]
(8)	Molecular Signal Tracking and Detection Methods in Fluid Dynamic Channels (+ Method and Data)	[267], [268]	2019, 2020	∼26.8 GB	Experimental	[277]	IEEE DataPort
(9)	A Molecular Communication Testbed Based on Proton Pumping Bacteria	[269]	2019	∼0.6 MB	Experimental	[278]	IEEE DataPort
(10)	Dataset in Support of the Southampton Doctoral Thesis "Type-Spread and Multiple-Access Molecular Communications"	[228]	2023	∼25 MB	Synthetic	[230]	University server
(11)	Real-Time Signal Processing via Chemical Reactions for a Microfluidic Molecular Communication System	[270]	2023	∼2.4 GB	Experimental	[158], [279]	Zenodo
(12)	Closed-Loop Long-Term Experimental Molecular Communication System	2025	[271], [272]	∼667 MB	Experimental	[84], [280], [281], [282]	Zenodo [272], GitHub [271]
(13)	CFD Simulation Dataset for Airborne Pathogen Transmission in Turbulent Channels	[223]	2024	∼506.3 MB	Synthetic	[224], [225]	Zenodo
(14)	Received Signal Modeling and BER Analysis for Molecular SISO Communications	[231]	2022	∼3.1 kB	Synthetic	[232]	Zenodo
(15)	Channel Estimation and Performance Analysis of SISO Molecular Communications	[283], [314]	2021	∼3.3 kB	Synthetic	Not specified	Zenodo
(16)	"molecular_communication"	[284]	2020	∼1.5 kB	Not specified	Not specified	Kaggle
(17)	"Distance-Estimation-in-Molecular-Communication"	[226]	2020	∼620 kB	Synthetic	[63], [227]	GitHub
(18)	"MCFormer: A Transformer-Based Detector for Molecular Communication with Accelerated Particle-Based Solution"	[233]	2023	∼41.8 MB	Synthetic	[114]	GitHub

TABLE VI: Overview of data-generating simulators, alphabetically sorted. URDME and MMFT denote the Unstructured Reaction-Diffusion Master Equation and the Munich MicroFluidic Toolkit, respectively.

Category	Simulator and dependence	Application area	Source code
Flow-agnostic	AcCoRD [243]	Microscopic and mesoscopic diffusion-based MC	GitHub [315]
	BiNS [316] & BiNS2 [317]	Diffusion-based MC in blood vessels	-
	BioNetGen [318]	Biochemical systems	Website/manual (https://bionetgen.org/)
	blood-voyager-s [319], BVS-Vis [320], MEHLISSA [321], BVS-Net [325] (ns-3 extensions)	Moving objects in human body	GitHub (blood-voyager-s [322], BVS-Vis [323], MEHLISSA [324], BVS-Net [326])
	BNSim [237]	Bacterial networks	Website (https://radum.ece.utexas.edu/bnsim-bacteria-network-simulator/)*
	MC GPU Simulator (Nvidia GPU supporting CUDA)	Diffusion-based MC	GitHub [235]
	MesoRD [327] (visualising results in Matlab)	Reaction-diffusion simulations	SourceForge [328]
	MolComSim	Simple active and passive MC	GitHub [329]
	MUCIN [330] (Matlab extension)	Diffusion-based MC with drift	Matlab Central File Exchange [331]
	Multicellular MC Simulator [332], [333]	(Large scale) multicellular MC scenarios	GitHub [334]
	nanoNS3 [236] (ns-3 extension)	Bacterial MC networks	Download (http://gnan.ece.gatech.edu/ns-allinone-3.24.zip)
	N3Sim [241], [242]	Diffusion-based MC	Website (http://www.n3cat.upc.edu/n3sim)*
	N^4Sim [335]	Nervous systems and synaptic MC	GitHub [336]
	Smartcell [337]	Cellular processes	Available upon email request; initial download link not available
	Smoldyn [239]	Diffusion, membrane interactions, and molecule reactions	Website https://www.smoldyn.org/
	URDME [238] (including Matlab and COMSOL interface)	Reaction-transport simulation and modeling	GitHub [338]
Flow-aware	ANSYS Fluent	Wide range of fluidic scenarios (CFD)	Commercially available
	COMSOL Multiphysics	Wide range of fluidic scenarios (CFD)	Commercially available
	Droplet-Based Microfluidic Simulator [339], [340] (MMFT extension)	Droplet-based fluidic MC	Github [341]
	Matlab	PDEs, CFD, and particle tracking for MC and fluidic scenarios	Commercially available
	OpenFOAM (Pogona [342], [343])	Wide range of fluidic scenarios (CFD)	freely available (Pogona - GitHub [344])

* Download not available anymore

REFERENCES

[1] E. Schrödinger, *What is Life? The Physical Aspect of the Living Cell*. Cambridge University Press, 1944, p. 196.

[2] I. F. Akyildiz, M. Pierobon, S. Balasubramaniam, and Y. Koucheryavy, "The Internet of Bio-Nano Things," *IEEE Communications Magazine*, vol. 53, no. 3, pp. 32–40, Mar. 2015. DOI: 10.1109/MCOM.2015.7060516.

[3] F. Dressler and S. Fischer, "Connecting In-Body Nano Communication with Body Area Networks: Challenges and Opportunities of the Internet of Nano Things," *Elsevier Nano Communication Networks*, vol. 6, pp. 29–38, Jun. 2015. DOI: 10.1016/j.nancom.2015.01.006.

[4] I. F. Akyildiz, M. Ghovanloo, U. Guler, T. Ozkaya-Ahmadov, A. F. Sarioglu, and B. D. Unluturk, "PANACEA: An Internet of Bio-NanoThings Application for Early Detection and Mitigation of Infectious Diseases," *IEEE Access*, vol. 8, pp. 140 512–140 523, Jan. 2020. DOI: 10.1109/access.2020.3012139.

[5] J. Philibert, "One and a Half Century of Diffusion: Fick, Einstein, Before and Beyond," *Diffusion Fundamentals*, vol. 4, pp. 1–19, Nov. 2006. DOI: 10.62721/diffusion-fundamentals.4.39.

[6] H. Xiao, K. Dokaj, and O. B. Akan, "What Really is "Molecule" in Molecular Communications? The Quest for Physics of Particle-Based Information Carriers," *IEEE Transactions on Molecular, Biological and Multi-Scale Communications*, vol. 10, no. 1, pp. 43–74, Mar. 2024. DOI: 10.1109/TMBMC.2023.3338950.

[7] A. Beck and M. Kurz, "A perspective on machine learning methods in turbulence modeling," *GAMM-Mitteilungen*, vol. 44, no. 1, pp. 1–27, Mar. 2021. DOI: 10.1002/gamm.202100002.

[8] H. Seckler, J. Szwabiński, and R. Metzler, "Machine-Learning Solutions for the Analysis of Single-Particle Diffusion Trajectories," *The Journal of Physical Chemistry Letters*, vol. 14, no. 35, pp. 7910–7923, Aug. 2023. DOI: 10.1021/acs.jpclett.3c01351.

[9] F. Cichos, K. Gustavsson, B. Mehlig, and G. Volpe, "Machine learning for active matter," *Nature Machine Intelligence*, vol. 2, no. 2, pp. 94–103, Feb. 2020. DOI: 10.1038/s42256-020-0146-9.

[10] N. Farsad and A. Goldsmith, "Neural Network Detection of Data Sequences in Communication Systems," *IEEE Transactions on Signal Processing*, vol. 66, no. 21, pp. 5663–5678, Nov. 2018. DOI: 10.1109/tsp.2018.2868322.

[11] B. H. Koo, C. Lee, H. B. Yilmaz, N. Farsad, A. W. Eckford, and C. B. Chae, "Molecular MIMO: From Theory to Prototype," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 3, pp. 600–614, Mar. 2016. DOI: 10.1109/JSAC.2016.2525538.

[12] J. Torres Gómez et al., "DNA-Based Nanonetwork for Abnormality Detection and Localization in the Human Body," *IEEE Transactions on Nanotechnology*, vol. 23, pp. 794–808, Nov. 2024. DOI: 10.1109/TNANO.2024.3495541.

[13] Y. C. Eldar, A. Goldsmith, D. Gündüz, and H. V. Poor, *Machine Learning and Wireless Communications*. Cambridge University Press, 2022. DOI: 10.1017/9781108966559.

[14] R. Khanzadeh, S. Angerbauer, F. Enzenhofer, A. Springer, and W. Haselmayr, "Towards End-to-End Learning for Salinity-based Molecular Communication," in *7th Workshop on Molecular Communications (WMC 2023)*, Erlangen, Germany, Apr. 2023, pp. 1–2.

[15] J. Torres Gómez et al., *Dataset for Cell-to-Cell Communications*, 2024. DOI: 10.21227/vv2r-qn28.

[16] M. Kuscu and B. D. Unluturk, "Internet of Bio-Nano Things: A review of applications, enabling technologies and key challenges," *ITU Journal on Future and Evolving Technologies*, vol. 2, no. 3, pp. 1–24, Dec. 2021. DOI: 10.52953/chbb9821.

[17] N. Etemadi, M. Farahnak-Ghazani, H. Arjmandi, M. Mirmohseni, and M. Nasiri-Kenari, "Abnormality Detection and Localization Schemes Using Molecular Communication Systems: A Survey," *IEEE Access*, vol. 11, pp. 1761–1792, 2023. DOI: 10.1109/access.2022.3228618.

[18] I. F. Akyildiz, A. Kak, and S. Nie, "6G and Beyond: The Future of Wireless Communications Systems," *IEEE Access*, vol. 8, pp. 133 995–134 030, 2020. DOI: 10.1109/access.2020.3010896.

[19] E. Lagasse and M. Levin, "Future medicine: from molecular pathways to the collective intelligence of the body," *Trends in Molecular Medicine*, vol. 29, no. 9, pp. 687–710, Sep. 2023. DOI: 10.1016/j.molmed.2023.06.007.

[20] I. F. Akyildiz, M. Pierobon, and S. Balasubramaniam, "Moving forward with molecular communication: from theory to human health applications [point of view]," *Proceedings of the IEEE*, vol. 107, no. 5, pp. 858–865, May 2019. DOI: 10.1109/jproc.2019.2913890.

[21] D. Aktas et al., "Odor-Based Molecular Communications: State-of-the-Art, Vision, Challenges, and Frontier Directions," *IEEE Communications Surveys & Tutorials*, pp. 1–34, 2024. DOI: 10.1109/comst.2024.3487472.

[22] K. Yang et al., "A Comprehensive Survey on Hybrid Communication in Context of Molecular Communication and Terahertz Communication for Body-Centric Nanonetworks," *IEEE Transactions on Molecular, Biological and Multi-Scale Communications*, vol. 6, no. 2, pp. 107–133, Nov. 2020. DOI: 10.1109/tmbmc.2020.3017146.

[23] S. Zafar et al., "A Systematic Review of Bio-Cyber Interface Technologies and Security Issues for Internet of Bio-Nano Things," *IEEE Access*, vol. 9, pp. 93 529–93 566, 2021. DOI: 10.1109/access.2021.3093442.

[24] Y. Koucheryavy, A. Yastrebova, D. P. Martins, and S. Balasubramaniam, "A Review on Bio-Cyber Interfaces for Intrabody Molecular Communications Systems," arXiv, eess.SY, Apr. 2021. DOI: 10.48550/arXiv.2104.14944.

[25] P. Kulakowski, K. Turbic, and L. M. Correia, "From Nano-Communications to Body Area Networks: A Perspective on Truly Personal Communications," *IEEE Access*, vol. 8, pp. 159 839–159 853, Jan. 2020. DOI: 10.1109/access.2020.3015825.

[26] Y. Lu, R. Ni, and Q. Zhu, "Wireless Communication in Nanonetworks: Current Status, Prospect and Challenges," *IEEE Transactions on Molecular, Biological and Multi-Scale Communications*, vol. 6, no. 2, pp. 71–80, Nov. 2020. DOI: 10.1109/tmbmc.2020.3004304.

[27] J. L. Marzo, J. M. Jornet, and M. Pierobon, "Nanonetworks in Biomedical Applications," *Current Drug Targets*, vol. 20, no. 8, pp. 800–807, May 2019. DOI: 10.2174/1389450120666190115152613.

[28] B. Dong, Y. Ma, Z. Ren, and C. Lee, "Recent progress in nanoplasmonics-based integrated optical micro/nano-systems," *Journal of Physics D: Applied Physics*, vol. 53, no. 21, Mar. 2020. DOI: 10.1088/1361-6463/ab77db.

[29] N. Saeed, M. H. Loukil, H. Sarieddeen, T. Y. Al-Naffouri, and M. S. Alouini, "Body-Centric Terahertz Networks: Prospects and Challenges," *IEEE Transactions on Molecular, Biological and Multi-Scale Communications*, vol. 8, no. 3, pp. 138–157, Sep. 2022. DOI: 10.1109/tmbmc.2021.3135198.

[30] S. Abadal et al., "Graphene-Based Antenna Design for Communications in the Terahertz Band," in *Nanoscale Networking and Communications Handbook*, J. R. Vacca, Ed., 1st ed., Boca Raton, FL: CRC Press, 2019, pp. 25–45.

[31] S. Abadal, C. Han, V. Petrov, L. Galluccio, I. F. Akyildiz, and J. M. Jornet, "Electromagnetic Nanonetworks Beyond 6G: From Wearable and Implantable Networks to On-Chip and Quantum Communication," *IEEE Journal on Selected Areas in Communications*, vol. 42, no. 8, pp. 2122–2142, Aug. 2024. DOI: 10.1109/jsac.2024.3399253.

[32] F. Lemic et al., "Survey on Terahertz Nanocommunication and Networking: A Top-Down Perspective," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 6, pp. 1506–1543, Jun. 2021. DOI: 10.1109/jsac.2021.3071837.

[33] X.-X. Yin, A. Baghai-Wadji, and Y. Zhang, "A Biomedical Perspective in Terahertz Nano-Communications—A Review," *IEEE Sensors Journal*, vol. 22, no. 10, pp. 9215–9227, May 2022. DOI: 10.1109/jsen.2022.3161013.

[34] S. Canovas-Carrasco, R. Asorey-Cacheda, A.-J. Garcia-Sanchez, J. Garcia-Haro, K. Wojcik, and P. Kulakowski, "Understanding the Applicability of Terahertz Flow-Guided Nano-Networks for Medical Applications," *IEEE Access*, vol. 8, pp. 214 224–214 239, 2020. DOI: 10.1109/access.2020.3041187.

[35] R. C. Moioli et al., "Neurosciences and Wireless Networks: The Potential of Brain-Type Communications and Their Applications," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 3, pp. 1599–1621, 2021. DOI: 10.1109/comst.2021.3090778.

[36] J. M. Jornet and A. Sangwan, "Nanonetworking in the Terahertz Band and Beyond," *IEEE Nanotechnology Magazine*, vol. 17, no. 3, pp. 21–31, Jun. 2023. DOI: 10.1109/mnano.2023.3262105.

[37] S. Qiu et al., "Review of Physical Layer Security in Molecular Internet of Nano-Things," *IEEE Transactions on NanoBioscience*, vol. 23, no. 1, pp. 91–100, Jan. 2024. DOI: 10.1109/tnb.2023.3285973.

[38] V. Jamali, A. Ahmadzadeh, W. Wicke, A. Noel, and R. Schober, "Channel Modeling for Diffusive Molecular Communication - A Tutorial Review," *Proceedings of the IEEE*, vol. 107, no. 7, pp. 1256–1301, Jul. 2019. DOI: 10.1109/jproc.2019.2919455.

[39] T. Nakano, Y. Okaie, S. Kobayashi, T. Hara, Y. Hiraoka, and T. Haraguchi, "Methods and Applications of Mobile Molecular Communication," *Proceedings of the IEEE*, vol. 107, no. 7, pp. 1442–1456, Jul. 2019. DOI: 10.1109/jproc.2019.2917625.

[40] E. Caferzade, "Modeling and Simulating Chemotaxis Bacteria Networks with Drift in MATLAB," Master's Thesis, School of Electrical Engineering and Computer Science, Berlin, Germany, Sep. 2023.

[41] J. S. Burggraf, "Modeling and Simulating Chemotaxis Bacteria Networks in MATLAB," Bachelor Thesis, School of Electrical Engineering and Computer Science, Berlin, Germany, Jul. 2022.

[42] X. Huang, Y. Fang, and N. Yang, "A survey on estimation schemes in molecular communications," *Elsevier Digital Signal Processing*, vol. 124, pp. 1–13, May 2022. DOI: 10.1016/j.dsp.2021.103163.

[43] M. S. Kuran, H. B. Yilmaz, I. Demirkol, N. Farsad, and A. Goldsmith, "A Survey on Modulation Techniques in Molecular Communication via Diffusion," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 1, pp. 7–28, Jan. 2021. DOI: 10.1109/comst.2020.3048099.

[44] M. C. Gursoy, M. Nasiri-Kenari, and S. Mitra, "Towards high data-rate diffusive molecular communications: A review on performance enhancement strategies," *Elsevier Digital Signal Processing*, vol. 124, pp. 1–17, May 2022. DOI: 10.1016/j.dsp.2021.103161.

[45] P. Hofmann, J. A. Cabrera, R. Bassoli, M. Reisslein, and F. H. P. Fitzek, "Coding in Diffusion-Based Molecular Nanonetworks: A Comprehensive Survey," *IEEE Access*, vol. 11, pp. 16 411–16 465, 2023. DOI: 10.1109/access.2023.3243797.

[46] B.-H. Koo, C. Lee, A. E. Pusane, T. Tugcu, and C.-B. Chae, "MIMO Operations in Molecular Communications: Theory, Prototypes, and Open Challenges," *IEEE Communications Magazine*, vol. 59, no. 9, pp. 98–104, Sep. 2021. DOI: 10.1109/mcom.110.2000984.

[47] S. Lotter et al., "Experimental Research in Synthetic Molecular Communications – Part I," *IEEE Nanotechnology Magazine*, vol. 17, no. 3, pp. 42–53, Jun. 2023. DOI: 10.1109/mnano.2023.3262100.

[48] S. Lotter et al., "Experimental Research in Synthetic Molecular Communications – Part II," *IEEE Nanotechnology Magazine*, vol. 17, no. 3, pp. 54–65, Jun. 2023. DOI: 10.1109/mnano.2023.3262377.

[49] M. Kuscu, E. Dinc, B. A. Bilgin, H. Ramezani, and O. B. Akan, "Transmitter and Receiver Architectures for Molecular Communications: A Survey on Physical Design With Modulation, Coding, and Detection Techniques," *Proceedings of the IEEE*, vol. 107, no. 7, pp. 1302–1341, Jul. 2019. DOI: 10.1109/jproc.2019.2916081.

[50] M. Veletic and I. Balasingham, "Synaptic Communication Engineering for Future Cognitive Brain–Machine Interfaces," *Proceedings of the IEEE*, vol. 107, no. 7, pp. 1425–1441, Jul. 2019. DOI: 10.1109/jproc.2019.2915199.

[51] J. M. Jornet et al., "Optogenomic Interfaces: Bridging Biological Networks With the Electronic Digital World," *Proceedings of the IEEE*, vol. 107, no. 7, pp. 1387–1401, Jul. 2019. DOI: 10.1109/jproc.2019.2916055.

[52] E. Kim et al., "Redox Is a Global Biodevice Information Processing Modality," *Proceedings of the IEEE*, vol. 107, no. 7, pp. 1402–1424, Jul. 2019. DOI: 10.1109/jproc.2019.2908582.

[53] Y. Huang, F. Ji, Z. Wei, M. Wen, and W. Guo, "Signal Detection for Molecular Communication: Model-Based vs. Data-Driven Methods," *IEEE Communications Magazine*, vol. 59, no. 5, pp. 47–53, May 2021. DOI: 10.1109/mcom.001.2000957.

[54] A.-A. A. Boulogeorgos, S. E. Trevlakis, S. A. Tegos, V. K. Papanikolaou, and G. K. Karagiannidis, "Machine Learning in Nano-Scale Biomedical Engineering," *IEEE Transactions on Molecular, Biological and Multi-Scale Communications*, vol. 7, no. 1, pp. 10–39, Mar. 2021. DOI: 10.1109/tmbmc.2020.3035383.

[55] R. T. Nagipogu, D. Fu, and J. H. Reif, "A survey on molecular-scale learning systems with relevance to DNA computing," *Nanoscale*, vol. 15, no. 17, pp. 7676–7694, 2023. DOI: 10.1039/d2nr06202j.

[56] A. Halužan Vasle and M. Moškon, "Synthetic biological neural networks: From current implementations to future perspectives," *Biosystems*, vol. 237, pp. 1–11, Mar. 2024. DOI: 10.1016/j.biosystems.2024.105164.

[57] M. Cao, X. Xiong, Y. Zhu, M. Xiao, L. Li, and H. Pei, "DNA computational device-based smart biosensors," *Trends in Analytical Chemistry (TrAC)*, vol. 159, pp. 1–10, Feb. 2023. DOI: 10.1016/j.trac.2022.116911.

[58] A. Rizwan et al., "A Review on the Role of Nano-Communication in Future Healthcare Systems: A Big Data Analytics Perspective," *IEEE Access*, vol. 6, pp. 41 903–41 920, May 2018. DOI: 10.1109/ACCESS.2018.2859340.

[59] P. L. Gentili, M. P. Zurlo, and P. Stano, "Neuromorphic engineering in wetware: the state of the art and its perspectives," *Frontiers in Neuroscience*, vol. 18, pp. 1–6, Sep. 2024. DOI: 10.3389/fnins.2024.1443121.

[60] D. Bi, A. Almpanis, A. Noel, Y. Deng, and R. Schober, "A Survey of Molecular Communication in Cell Biology: Establishing a New Hierarchy for Interdisciplinary Applications," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 3, pp. 1494–1545, 2021. DOI: 10.1109/comst.2021.3066117.

[61] M. Egan et al., "Toward Interdisciplinary Synergies in Molecular Communications: Perspectives from Synthetic Biology, Nanotechnology, Communications Engineering and Philosophy of Science," *Life*, vol. 13, no. 1, pp. 1–14, Jan. 2023. DOI: 10.3390/life13010208.

[62] H. B. Yilmaz, C. Lee, Y. J. Cho, and C.-B. Chae, "A machine learning approach to model the received signal in molecular communications," in *IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom 2017)*, Istanbul, Turkey: IEEE, Jun. 2017, pp. 1–5. DOI: 10.1109/blackseacom.2017.8277667.

[63] F. Güleç and B. Atakan, "Distance estimation methods for a practical macroscale molecular communication system," *Elsevier Nano Communication Networks*, vol. 24, pp. 1–15, May 2020. DOI: 10.1016/j.nancom.2020.100300.

[64] S. Mohamed, J. Dong, S. M. A. El-Atty, and M. A. Eissa, "Bio-Cyber Interface Parameter Estimation with Neural Network for the Internet of Bio-Nano Things," *Wireless Personal Communications: An International Journal*, vol. 123, no. 2, pp. 1245–1263, Sep. 2021. DOI: 10.1007/s11277-021-09177-6.

[65] N. Farsad, H. B. Yilmaz, A. W. Eckford, C.-B. Chae, and W. Guo, "A Comprehensive Survey of Recent Advancements in Molecular Communication," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 3, pp. 1887–1919, 2016. DOI: 10.1109/comst.2016.2527741.

[66] C. Lee, H. B. Yilmaz, C.-B. Chae, N. Farsad, and A. Goldsmith, "Machine learning based channel modeling for molecular MIMO communications," in *18th IEEE International Workshop on Signal Processing Advances in Wireless Communications (SPAWC 2017)*, Sapporo, Japan: IEEE, Jul. 2017, pp. 1–5. DOI: 10.1109/spawc.2017.8227765.

[67] H. U. Ozdemir, H. I. Orhan, M. Turan, B. Buyuktas, and H. B. Yilmaz, "Estimating Capture Probabilities for Complex Topologies in 2D Molecular Communication via Diffusion Channel using Artificial Neural Networks," in *9th IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom 2021)*, Bucharest, Romania: IEEE, May 2021, pp. 1–6. DOI: 10.1109/blackseacom52164.2021.9527790.

[68] H. U. Ozdemir, H. I. Orhan, M. Turan, B. Büyüktaş, and H. B. Yilmaz, "Estimating channel coefficients for complex topologies in 3D diffusion channel using artificial neural networks," *Elsevier Nano Communication Networks*, vol. 42, pp. 1–12, Dec. 2024. DOI: 10.1016/j.nancom.2024.100549.

[69] Z. Cheng, M. Chen, H. Liu, M. Xia, and W. Gong, "Channel modeling for diffusion-based molecular MIMO communications using deep learning," *Elsevier Nano Communication Networks*, vol. 42, pp. 1–9, Dec. 2024. DOI: 10.1016/j.nancom.2024.100543.

[70] M. Damrath, B. Yilmaz, C.-B. Chae, and P. A. Hoeher, "Array Gain Analysis in Molecular MIMO Communications," *IEEE Access*, vol. 6, pp. 61 091–61 102, 2018. DOI: 10.1109/access.2018.2875925.

[71] Z. Cheng, H. Liu, Z. Xu, J. Li, and K. Chi, "Deep Learning-Based Estimation of Emission Time and Arrival Time in Diffusive Multi-Receiver Molecular Communication," *IEEE Transactions on Molecular, Biological and Multi-Scale Communications*, vol. 11, no. 2, pp. 257–268, Jun. 2025. DOI: 10.1109/tmbmc.2025.3546503.

[72] E. Ozbey, Y. K. Cicekdag, and H. B. Yilmaz, "Artificial neural network based misorientation correction in molecular 4x4 MIMO systems," *Elsevier Nano Communication Networks*, vol. 42, pp. 1–9, Dec. 2024. DOI: 10.1016/j.nancom.2024.100544.

[73] S. Dhillon and A. Kostrzewski, *Clinical Pharmacokinetics*. London, United Kingdom: Pharmaceutical Press, 2006, p. 280.

[74] M. Hagan and M. Menhaj, "Training feedforward networks with the Marquardt algorithm," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 5, no. 6, pp. 989–993, 1994. DOI: 10.1109/72.329697.

[75] M. Zoofaghari, F. Pappalardo, M. Damrath, and I. Balasingham, "Modeling Extracellular Vesicles-Mediated Interactions of Cells in the Tumor Microenvironment," *IEEE Transactions on NanoBioscience*, vol. 23, no. 1, pp. 71–80, Jan. 2024. DOI: 10.1109/tnb.2023.3284090.

[76] V. Jamali, A. Ahmadzadeh, and R. Schober, "Symbol Synchronization for Diffusion-Based Molecular Communications," *IEEE Transactions on NanoBioscience*, vol. 16, no. 8, pp. 873–887, Dec. 2017. DOI: 10.1109/tnb.2017.2782761.

[77] M. Mukherjee, H. B. Yilmaz, B. B. Bhowmik, J. Lloret, and Y. Lv, "Synchronization for Diffusion-Based Molecular Communication Systems via Faster Molecules," in *IEEE International Conference*

on Communications (ICC 2019), Shanghai, China: IEEE, May 2019, pp. 1–5. DOI: 10.1109/ICC.2019.8761827.

[78] H. ShahMohammadian, G. G. Messier, and S. Magierowski, "Blind Synchronization in Diffusion-Based Molecular Communication Channels," *IEEE Communications Letters*, vol. 17, no. 11, pp. 2156–2159, Nov. 2013. DOI: 10.1109/lcomm.2013.100713.131727.

[79] N. C. Luong et al., "Applications of Deep Reinforcement Learning in Communications and Networking: A Survey," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 4, pp. 3133–3174, 2019. DOI: 10.1109/comst.2019.2916583.

[80] L. Y. Debus, P. Hofmann, J. Torres Gómez, F. H. P. Fitzek, and F. Dressler, "Reinforcement Learning-based Receiver for Molecular Communication with Mobility," in *IEEE Global Communications Conference (GLOBECOM 2023)*, Kuala Lumpur, Malaysia: IEEE, Dec. 2023, pp. 558–564. DOI: 10.1109/GLOBECOM54140.2023.10436754.

[81] P. Hofmann, J. Torres Gómez, F. Dressler, and F. H. P. Fitzek, "Testbed-based Receiver Optimization for SISO Molecular Communication Channels," in *5th IEEE International Balkan Conference Communications and Networking (BalkanCom 2022)*, Sarajevo, Bosnia and Herzegovina: IEEE, Aug. 2022, pp. 120–125. DOI: 10.1109/BalkanCom55633.2022.9900720.

[82] L. Y. Debus, P. Hofmann, J. Torres Gómez, and F. Dressler, "Synchronized Relaying in Molecular Communication: An AI-based Approach using a Mobile Testbed Setup," *IEEE Transactions on Molecular, Biological and Multi-Scale Communications*, vol. 10, no. 3, pp. 470–475, Sep. 2024. DOI: 10.1109/TMBMC.2024.3420792.

[83] D. Casaleiro, N. Souto, and J. C. Silva, "Synchronisation and Detection in Molecular Communication using a Deep-Learning-based Approach," *IEEE Access*, pp. 192 539–192 553, 2024. DOI: 10.1109/access.2024.3519310.

[84] L. Brand et al., "Media Modulation Based Molecular Communication," *IEEE Transactions on Communications*, vol. 70, no. 11, pp. 7207–7223, Nov. 2022. DOI: 10.1109/TCOMM.2022.3205949.

[85] T. Brakemann et al., "A Reversibly Photoswitchable GFP-like Protein with Fluorescence Excitation Decoupled from Switching," *Nature Biotechnology*, vol. 29, no. 10, pp. 942–947, Oct. 2011. DOI: 10.1038/nbt.1952.

[86] J. Schulman, R. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal Policy Optimization Algorithms," arXiv, cs.LG, Jul. 2017, pp. 1–12. DOI: 10.48550/ARXIV.1707.06347.

[87] M. Andrychowicz et al., "What Matters In On-Policy Reinforcement Learning? A Large-Scale Empirical Study," arXiv, cs.LG, Jun. 2020, pp. 1–48. DOI: 10.48550/ARXIV.2006.05990.

[88] X. Qian, M. D. Renzo, and A. W. Eckford, "Molecular Communications: Model-Based and Data-Driven Receiver Design and Optimization," *IEEE Access*, vol. 7, pp. 53 555–53 565, Jan. 2019. DOI: 10.1109/access.2019.2912600.

[89] G. H. Alshammri, W. K. M. Ahmed, and V. B. Lawrence, "Adaptive Batch Training Rule-Based Detection Scheme for On-OFF-Keying Diffusion-Based Molecular Communications," in *13th IEEE Nanotechnology Materials and Devices Conference (NMDC 2018)*, Portland, OR: IEEE, Oct. 2018, pp. 1–4. DOI: 10.1109/nmdc.2018.8605873.

[90] U. K. Agrawal, A. K. Shrivastava, D. Das, and R. Mahapatra, "Neural Network Detector in Mobile Molecular Communication for Fast Varying Channels," in *International Conference on Connected Systems and Intelligence (CSI 2022)*, Trivandrum, India: IEEE, Aug. 2022, pp. 1–5. DOI: 10.1109/csi54720.2022.9924143.

[91] S. Sharma, D. Dixit, and K. Deka, "Deep Learning based Symbol Detection for Molecular Communications," in *IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS 2020)*, New Delhi, India: IEEE, Dec. 2020, pp. 1–6. DOI: 10.1109/ants50601.2020.9342782.

[92] S.-J. Kim, P. Singh, and S.-Y. Jung, "A machine learning-based concentration-encoded molecular communication system," *Elsevier Nano Communication Networks*, vol. 35, pp. 1–12, Mar. 2023. DOI: 10.1016/j.nancom.2022.100433.

[93] O. T. Baydas, O. Cetinkaya, and O. B. Akan, "Estimation and Detection for Molecular MIMO Communications in the Internet of Bio-Nano Things," *IEEE Transactions on Molecular, Biological and Multi-Scale Communications*, vol. 9, no. 1, pp. 106–110, Mar. 2023. DOI: 10.1109/tmbmc.2023.3252943.

[94] Z. Cheng, Z. Zhang, and J. Sun, "Signal Detection of Cooperative Multi-Hop Mobile Molecular Communication via Diffusion," *IEEE Transactions on Molecular, Biological and Multi-Scale Communications*, vol. 10, no. 1, pp. 101–111, Mar. 2024. DOI: 10.1109/tmbmc.2024.3360341.

[95] O. Kara, G. Yaylali, A. E. Pusane, and T. Tugcu, "Molecular index modulation using convolutional neural networks," *Elsevier Nano Communication Networks*, vol. 34, pp. 1–8, Dec. 2022. DOI: 10.1016/j.nancom.2022.100420.

[96] A. K. Shrivastava, D. Das, and R. Mahapatra, "Performance Evaluation of Mobile Molecular Communication System Using Neural Network Detector," *IEEE Wireless Communications Letters*, vol. 10, no. 8, pp. 1776–1779, Aug. 2021. DOI: 10.1109/lwc.2021.3079522.

[97] A. K. Shrivastava, D. Das, R. Mahapatra, and N. Varshney, "Scaled Conjugate Gradient Algorithm for Neural Network Detector in Mobile Molecular Communication," in *IEEE Global Communications Conference (GLOBECOM 2021)*, Madrid, Spain: IEEE, Dec. 2021, pp. 1–6. DOI: 10.1109/globecom46510.2021.9685034.

[98] Z. Cheng, Z. Zhang, J. Jiang, and J. Sun, "Signal Detection of Mobile Multi-user Molecular Communication System Using Transformer-Based Model," in *8th International Conference on Computer and Communication Systems (ICCCS 2023)*, Guangzhou, China: IEEE, Apr. 2023, pp. 85–90. DOI: 10.1109/icccs57501.2023.10151419.

[99] Z. Cheng, Z. Zhang, X. Jin, W. Gong, and K. Chi, "An Informer-Based Signal Sequence Detector for Mobile Molecular Communication," *IEEE Communications Letters*, vol. 28, no. 6, pp. 1397–1401, Jun. 2024. DOI: 10.1109/lcomm.2024.3381600.

[100] X. Qian and M. Di Renzo, "Receiver Design in Molecular Communications: An Approach Based on Artificial Neural Networks," in *15th IEEE International Symposium on Wireless Communication Systems (ISWCS 2018)*, Lisbon, Portugal: IEEE, Aug. 2018, pp. 1–5. DOI: 10.1109/iswcs.2018.8491088.

[101] T. Agrawal, *Hyperparameter Optimization in Machine Learning: Make Your Machine Learning and Deep Learning Models More Efficient*. Apress, 2021, p. 177. DOI: 10.1007/978-1-4842-6579-6.

[102] L. Chen and L. Sun, "Self-Attention-Based Real-Time Signal Detector for Communication Systems With Unknown Channel Models," *IEEE Communications Letters*, vol. 25, no. 8, pp. 2639–2643, Aug. 2021. DOI: 10.1109/lcomm.2021.3082708.

[103] N. Farsad and A. Goldsmith, "Detection Algorithms for Communication Systems Using Deep Learning," arXiv, cs.LG, Jul. 2017, pp. 1–10. DOI: 10.48550/arXiv.1705.08044.

[104] N. Farsad and A. Goldsmith, "Sliding Bidirectional Recurrent Neural Networks for Sequence Detection in Communication Systems," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2018)*, Calgary, Canada: IEEE, Apr. 2018, pp. 2331–2335. DOI: 10.1109/icassp.2018.8462140.

[105] L. Sun and Y. Wang, "CTBRNN: A Novel Deep-Learning Based Signal Sequence Detector for Communications Systems," *IEEE Signal Processing Letters*, vol. 27, pp. 21–25, 2020. DOI: 10.1109/lsp.2019.2953673.

[106] M. Bartunik, O. Keszocze, B. Schiller, and J. Kirchner, "Using Deep Learning to Demodulate Transmissions in Molecular Communication," in *16th IEEE International Symposium on Medical Information and Communication Technology (ISMICT 2022)*, Lincoln, NE: IEEE, May 2022, pp. 1–6. DOI: 10.1109/ismict56646.2022.9828263.

[107] M. Bartunik, J. Kirchner, and O. Keszocze, "Artificial intelligence for molecular communication," *it - Information Technology (itIT)*, vol. 65, no. 4-5, pp. 155–163, Aug. 2023. DOI: 10.1515/itit-2023-0029.

[108] F. Vakilipoor, D. Scazzoli, F. Ratti, G. Scalia, and M. Magarini, "Hybrid deep learning-based feature-augmented detection for molecular communication systems," in *9th ACM International Conference on Nanoscale Computing and Communication (NANOCOM 2022)*, Barcelona, Spain: ACM, Oct. 2022, pp. 1–6. DOI: 10.1145/3558583.3558859.

[109] C. Bai, A. Zhu, X. Lu, Y. Zhu, and K. Wang, "Temporal Convolutional Network-Based Signal Detection for Magnetotactic Bacteria Communication System," *IEEE Transactions on NanoBioscience*, vol. 22, no. 4, pp. 943–955, Oct. 2023. DOI: 10.1109/tnb.2023.3262555.

[110] A. Vaswani et al., "Attention is all you need," in *31st International Conference on Neural Information Processing Systems (NIPS 2017)*, Long Beach, CA: Curran Associates Inc., Dec. 2017, pp. 6000–6010.

[111] N. Farsad and A. Goldsmith, "Detection Over Rapidly Changing Communication Channels Using Deep Learning," in *52nd Asilomar Conference on Signals, Systems, and Computers*, Pacific Grove, CA: IEEE, Oct. 2018, pp. 1–5. DOI: 10.1109/acssc.2018.8645187.

[112] B.-H. Koo, H. J. Kim, J.-Y. Kwon, and C.-B. Chae, "Deep Learning-based Human Implantable Nano Molecular Communications," in *IEEE International Conference on Communications (ICC 2020)*, Virtual Conference: IEEE, Jun. 2020, pp. 1–7. DOI: 10.1109/icc40277.2020.9148818.

[113] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," arXiv, cs.LG 1412.6980, Dec. 2014, pp. 1–15. DOI: 10.48550/arXiv.1412.6980.

[114] X. Lu, C. Bai, A. Zhu, Y. Zhu, and K. Wang, "MCFormer: A Transformer-Based Detector for Molecular Communication With Accelerated Particle-Based Solution," *IEEE Communications Letters*, vol. 27, no. 10, pp. 2837–2841, Oct. 2023. DOI: 10.1109/lcomm.2023.3303091.

[115] D. Bahdanau, K. Cho, and Y. Bengio, "Neural Machine Translation by Jointly Learning to Align and Translate," arXiv, cs.CL, May 2014, pp. 1–15. DOI: 10.48550/ARXIV.1409.0473.

[116] P. Hofmann, J. Torres Gómez, F. H. Frank H.P., and F. Dressler, *Dataset for Macroscale Molecular Communication Testbed*, 2023. DOI: 10.21227/ytkm-xp81.

[117] C. Xiang, Y. Zhang, Y. Huang, W. Tan, X. Chen, and M. Wen, "Hybrid Recurrent Neural Network for Signal-Dependent Noise Suppression in Molecular Communication," *IEEE Transactions on Molecular, Biological and Multi-Scale Communications*, vol. 11, no. 2, pp. 283–291, Jun. 2025. DOI: 10.1109/tmbmc.2025.3546208.

[118] S. Mohamed, D. Jiang, A. R. Junejo, and D. C. Zuo, "Model-Based: End-to-End Molecular Communication System Through Deep Reinforcement Learning Auto Encoder," *IEEE Access*, vol. 7, pp. 70 279–70 286, 2019. DOI: 10.1109/access.2019.2916701.

[119] R. Khanzadeh, S. Angerbauer, A. Springer, and W. Haselmayr, "End-to-End Learning of Communication Systems with Novel Data-Efficient IIR Channel Identification," in *57th Asilomar Conference on Signals, Systems, and Computers*, Pacific Grove, CA: IEEE, Oct. 2023, pp. 40–46. DOI: 10.1109/ieeeconf59524.2023.10476924.

[120] S. Angerbauer, R. Khanzadeh, F. Enzenhofer, A. Springer, and W. Haselmayr, "Towards Asymmetric Auto-Encoders for the IoBNT," in *10th ACM International Conference on Nanoscale Computing and Communication (NANOCOM 2023)*, Coventry, United Kingdom: ACM, Sep. 2023, pp. 166–167. DOI: 10.1145/3576781.3608733.

[121] S. Angerbauer et al., "Salinity-Based Molecular Communication in Microfluidic Channels," *IEEE Transactions on Molecular, Biological and Multi-Scale Communications*, vol. 9, no. 2, pp. 191–206, Jun. 2023. DOI: 10.1109/tmbmc.2023.3277391.

[122] R. Khanzadeh et al., "Explainable Asymmetric Auto-Encoder for End-to-End Learning of IoBNT Communications," in *IEEE International Conference on Machine Learning for Communication and Networking (ICMLCN 2024)*, Stockholm, Sweden: IEEE, May 2024, pp. 412–418. DOI: 10.1109/ICMLCN59089.2024.10624774.

[123] Z. Cheng, J. Sun, Z. Zhang, P. Hu, and K. Chi, "Channel Modeling and Optimal Released Molecules for Mobile Molecular MIMO Communications Among Bionanosensors," *IEEE Sensors Journal*, vol. 23, no. 19, pp. 22 139–22 152, Oct. 2023. DOI: 10.1109/jsen.2023.3304971.

[124] R. Khanzadeh, S. Angerbauer, J. Torres Gómez, A. Springer, F. Dressler, and W. Haselmayr, "QL-based Adaptive Transceivers for the IoBNT Communications," *IEEE Transactions on Molecular, Biological and Multi-Scale Communications*, vol. 10, no. 3, pp. 476–480, Sep. 2024. DOI: 10.1109/TMBMC.2024.3420749.

[125] Z. Cheng, J. Yan, J. Sun, S. Zhang, and K. Chi, "Resource Allocation Optimization in Mobile Multiuser Molecular Communication by Deep Neural Network," *IEEE Transactions on Molecular, Biological and Multi-Scale Communications*, vol. 10, no. 3, pp. 409–421, Sep. 2024. DOI: 10.1109/tmbmc.2024.3412669.

[126] J. Torres Gómez, A. Kuestner, J. Simonjan, B. D. Unluturk, and F. Dressler, "Nanosensor Location Estimation in the Human Circulatory System using Machine Learning," *IEEE Transactions on Nanotechnology*, vol. 21, pp. 663–673, Oct. 2022. DOI: 10.1109/TNANO.2022.3217653.

[127] J. Torres Gómez, A. Kuestner, K. Pitke, J. Simonjan, B. D. Unluturk, and F. Dressler, "A Machine Learning Approach for Abnormality Detection in Blood Vessels via Mobile Nanosensors," in *19th ACM Conference on Embedded Networked Sensor Systems (SenSys 2021), 2nd ACM International Workshop on Nanoscale Computing, Communication, and Applications (NanoCoCoA 2021)*, Coimbra, Portugal: ACM, Nov. 2021, pp. 596–602. DOI: 10.1145/3485730.3494037.

[128] P. Galván, F. Lemic, G. Calvo, S. Abadal, and X. Costa-Pérez, "Tailoring Graph Neural Network-based Flow-guided Localization to Individual Bloodstreams and Activities," in *11th ACM International Conference on Nanoscale Computing and Communication (NANOCOM 2024)*, Milan, Italy: ACM, Oct. 2024, pp. 109–115. DOI: 10.1145/3686015.3689356.

[129] X. Jin, Z. Cheng, M. Chen, H. Liu, W. Gong, and K. Chi, "Transformer-Based Receiver Localization in Vessel-Like and Flow-Induced Molecular Communication via Diffusion," *IEEE Communications Letters*, vol. 28, no. 10, pp. 2283–2287, Oct. 2024. DOI: 10.1109/lcomm.2024.3432146.

[130] G. C. Bartra et al., "Graph Neural Networks as an Enabler of Terahertz-Based Flow-Guided Nanoscale Localization Over Highly Erroneous Raw Data," *IEEE Journal on Selected Areas in Communications*, vol. 42, no. 8, pp. 1992–2008, Aug. 2024. DOI: 10.1109/jsac.2024.3399257.

[131] O. D. Kose, M. C. Gursoy, M. Saraclar, A. E. Pusane, and T. Tugcu, "Machine Learning-Based Silent Entity Localization Using Molecular Diffusion," *IEEE Communications Letters*, vol. 24, no. 4, pp. 807–810, Apr. 2020. DOI: 10.1109/lcomm.2020.2968319.

[132] J. Torres Gómez, R. Wendt, A. Kuestner, K. Pitke, L. Stratmann, and F. Dressler, "Markov Model for the Flow of Nanobots in the Human Circulatory System," in *8th ACM International Conference on Nanoscale Computing and Communication (NANOCOM 2021)*, Virtual Conference: ACM, Sep. 2021, pp. 1–7. DOI: 10.1145/3477206.3477477.

[133] A. B. López et al., "Toward Standardized Performance Evaluation of Flow-guided Nanoscale Localization," *IEEE Transactions on Molecular, Biological and Multi-Scale Communications*, vol. 11, no. 1, pp. 116–127, Mar. 2025. DOI: 10.1109/TMBMC.2024.3523428.

[134] G. Pascual, F. Lemic, C. Delgado, and X. Costa-Pérez, "Analytical Modelling of Raw Data for Flow-Guided In-body Nanoscale Localization," in *IEEE International Conference on Machine Learning for Communication and Networking (ICMLCN 2024)*, Stockholm, Sweden: IEEE, May 2024, pp. 428–433. DOI: 10.1109/icmlcn59089.2024.10625169.

[135] S. N. Solak and M. Oner, "Neural Network Based Decision Fusion for Abnormality Detection via Molecular Communications," in *IEEE Workshop on Signal Processing Systems (SiPS)*, Coimbra, Portugal: IEEE, Oct. 2020, pp. 1–5. DOI: 10.1109/sips50750.2020.9195212.

[136] S. N. Solak and M. Oner, "RNN based abnormality detection with nanoscale sensor networks using molecular communications," in *7th ACM International Conference on Nanoscale Computing and Communication (NANOCOM 2020)*, Virtual Conference: ACM, Sep. 2020, pp. 1–6. DOI: 10.1145/3411295.3411313.

[137] T. C. Mai, M. Egan, T. Q. Duong, and M. Di Renzo, "Event Detection in Molecular Communication Networks With Anomalous Diffusion," *IEEE Communications Letters*, vol. 21, no. 6, pp. 1249–1252, Jun. 2017. DOI: 10.1109/lcomm.2017.2669315.

[138] Q. Ma et al., "A Survey on Time-Series Pre-Trained Models," *IEEE Transactions on Knowledge and Data Engineering*, pp. 1–20, 2024. DOI: 10.1109/tkde.2024.3475809.

[139] J.-L. Wu, H. Xiao, and E. Paterson, "Physics-informed machine learning approach for augmenting turbulence models: A comprehensive framework," *Physical Review Fluids*, vol. 3, no. 7, pp. 1–28, Jul. 2018. DOI: 10.1103/physrevfluids.3.074602.

[140] X. Chen, L. Yang, J. Duan, and G. E. Karniadakis, "Solving Inverse Stochastic Problems from Discrete Particle Observations Using the Fokker–Planck Equation and Physics-Informed Neural Networks," *SIAM Journal on Scientific Computing*, vol. 43, no. 3, B811–B830, Jan. 2021. DOI: 10.1137/20m1360153.

[141] L. Yang and A. Shami, "On hyperparameter optimization of machine learning algorithms: Theory and practice," *Neurocomputing*, vol. 415, pp. 295–316, Nov. 2020. DOI: 10.1016/j.neucom.2020.07.061.

[142] R. Mosayebi, A. Ahmadzadeh, W. Wicke, V. Jamali, R. Schober, and M. Nasiri-Kenari, "Early Cancer Detection in Blood Vessels Using Mobile Nanosensors," *IEEE Transactions on NanoBioscience*, vol. 18, no. 4, pp. 103–116, Oct. 2019. DOI: 10.1109/tnb.2018.2885463.

[143] L. Felicetti, M. Femminella, G. Reali, T. Nakano, and A. V. Vasilakos, "TCP-Like Molecular Communications," *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 12, pp. 2354–2367, Dec. 2014. DOI: 10.1109/jsac.2014.2367653.

[144] L. Lai, H. El Gamal, H. Jiang, and H. V. Poor, "Cognitive Medium Access: Exploration, Exploitation, and Competition," *IEEE Transactions on Mobile Computing*, vol. 10, no. 2, pp. 239–253, Feb. 2011. DOI: 10.1109/tmc.2010.65.

[145] S. Haykin, "Cognitive radio: brain-empowered wireless communications," *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 2, pp. 201–220, Feb. 2005. DOI: 10.1109/JSAC.2004.839380.

[146] J. Torres Gómez, N. Spicher, J. L. González Rios, and F. Dressler, "Fine-tune Circuit Representation of Human Vessels through Reinforcement Learning: A Novel Digital Twin Approach for Hemodynamics," in *10th ACM International Conference on Nanoscale Computing and Communication (NANOCOM 2023)*, Coventry, United Kingdom: ACM, Sep. 2023, pp. 46–52. DOI: 10.1145/3576781.3608717.

[147] Á. Goñi-Moreno, "Biocomputation: Moving Beyond Turing with Living Cellular Computers," *Communications of the ACM*, vol. 67, no. 6, pp. 70–77, May 2024. DOI: 10.1145/3635470.

[148] N. Dehghani and M. Levin, "Bio-inspired AI: Integrating Biological Complexity into Artificial Intelligence," arXiv, q-bio.NC, Nov. 2024, pp. 1–14. DOI: 10.48550/ARXIV.2411.15243.

[149] A. Agiza, S. Marriott, J. K. Rosenstein, E. Kim, and S. Reda, "pH-Controlled enzymatic computing for digital circuits and neural networks," *Physical Chemistry Chemical Physics*, vol. 26, no. 31, pp. 20 898–20 907, 2024. DOI: 10.1039/d4cp02039a.

[150] B. Wang, M. Barahona, and M. Buck, "A modular cell-based biosensor using engineered genetic logic circuits to detect and integrate multiple environmental signals," *Biosensors and Bioelectronics*, vol. 40, no. 1, pp. 368–376, Feb. 2013. DOI: 10.1016/j.bios.2012.08.011.

[151] S. Brown and Z. Vranesic, *Fundamentals of Digital Logic with VHDL Design*, 3rd ed. Boston, MA: McGraw-Hill, 2009, p. 934.

[152] A. Hjelmfelt, E. D. Weinberger, and J. Ross, "Chemical implementation of neural networks and Turing machines," *Proceedings of the National Academy of Sciences (PNAS)*, vol. 88, no. 24, pp. 10 983–10 987, Dec. 1991. DOI: 10.1073/pnas.88.24.10983.

[153] L. Deng, "The MNIST Database of Handwritten Digit Images for Machine Learning Researc," *IEEE Signal Processing Magazine*, vol. 29, no. 6, pp. 141–142, Nov. 2012. DOI: 10.1109/msp.2012.2211477.

[154] D. Bi and Y. Deng, "Digital Signal Processing for Molecular Communication via Chemical-Reaction-Based Microfluidic Circuits," *IEEE Communications Magazine*, vol. 59, no. 5, pp. 26–32, May 2021. DOI: 10.1109/mcom.001.2000830.

[155] A. O. Bicen and I. F. Akyildiz, "System-Theoretic Analysis and Least-Squares Design of Microfluidic Channels for Flow-Induced Molecular Communication," *IEEE Transactions on Signal Processing*, vol. 61, no. 20, pp. 5000–5013, Oct. 2013. DOI: 10.1109/tsp.2013.2274959.

[156] P. Hofmann, P. Zhou, C. Lee, M. Reisslein, F. H. P. Fitzek, and C.-B. Chae, "OpenFOAM Simulation of Microfluidic Molecular Communications: Method and Experimental Validation," *IEEE Access*, vol. 12, pp. 109 494–109 512, 2024. DOI: 10.1109/access.2024.3438243.

[157] D. Bi, Y. Deng, M. Pierobon, and A. Nallanathan, "Chemical Reactions-Based Microfluidic Transmitter and Receiver Design for Molecular Communication," *IEEE Transactions on Communications*, vol. 68, no. 9, pp. 5590–5605, Sep. 2020. DOI: 10.1109/tcomm.2020.2993633.

[158] V. Walter, D. Bi, A. Salehi-Reyhani, and Y. Deng, "Real-time signal processing via chemical reactions for a microfluidic molecular communication system," *Nature Communications*, vol. 14, no. 1, pp. 1–14, 2023. DOI: 10.1038/s41467-023-42885-0.

[159] A. Amerizadeh et al., "Bacterial Receiver Prototype for Molecular Communication Using Rhamnose Operon in a Microfluidic Environment," *IEEE Transactions on NanoBioscience*, vol. 20, no. 4, pp. 426–435, Oct. 2021. DOI: 10.1109/tnb.2021.3090761.

[160] M. Kuscu, H. Ramezani, E. Dinc, S. Akhavan, and O. B. Akan, "Fabrication and microfluidic analysis of graphene-based molecular communication receiver for Internet of Nano Things (IoNT)," *Scientific Reports*, vol. 11, no. 1, pp. 1–20, Oct. 2021. DOI: 10.1038/s41598-021-98609-1.

[161] A. Abdali and M. Kuscu, "Frequency-Domain Model of Microfluidic Molecular Communication Channels With Graphene BioFET-Based Receivers," *IEEE Transactions on Communications*, vol. 72, no. 8, pp. 4564–4576, Aug. 2024. DOI: 10.1109/tcomm.2024.3376593.

[162] J. Perera et al., "Wet-Neuromorphic Computing: A New Paradigm for Biological Artificial Intelligence," *IEEE Intelligent Systems*, pp. 1–7, 2025. DOI: 10.1109/mis.2025.3555551.

[163] H. Cai et al., "Brain organoid reservoir computing for artificial intelligence," *Nature Electronics*, vol. 6, no. 12, pp. 1032–1039, Dec. 2023. DOI: 10.1038/s41928-023-01069-w.

[164] B. J. Kagan et al., "*In vitro* neurons learn and exhibit sentience when embodied in a simulated game-world," *Neuron*, vol. 110, no. 23, pp. 3952–3969, Dec. 2022. DOI: 10.1016/j.neuron.2022.09.001.

[165] O. Purcell and T. K. Lu, "Synthetic analog and digital circuits for cellular computation and memory," *Current Opinion in Biotechnology*, vol. 29, pp. 146–155, Oct. 2014. DOI: 10.1016/j.copbio.2014.04.009.

[166] S. Balasubramaniam, S. Somathilaka, S. Sun, A. Ratwatte, and M. Pierobon, "Realizing Molecular Machine Learning Through Communications for Biological AI," *IEEE Nanotechnology Magazine*, vol. 17, no. 3, pp. 10–20, Jun. 2023. DOI: 10.1109/mnano.2023.3262099.

[167] F.-L. A. Lau, R. Wendt, and S. Fischer, "Efficient in-message computation of prevalent mathematical operations in DNA-based nanonetworks," *Elsevier Nano Communication Networks*, vol. 28, pp. 1–17, Jun. 2021. DOI: 10.1016/j.nancom.2021.100348.

[168] Y. Brun, "Arithmetic computation in the tile assembly model: Addition and multiplication," *Theoretical Computer Science*, vol. 378, no. 1, pp. 17–31, Jun. 2007. DOI: 10.1016/j.tcs.2006.10.025.

[169] R. Daniel, J. R. Rubens, R. Sarpeshkar, and T. K. Lu, "Synthetic analog computation in living cells," *Nature*, vol. 497, no. 7451, pp. 619–623, May 2013. DOI: 10.1038/nature12148.

[170] S. S. Somathilaka, S. Balasubramaniam, D. P. Martins, and X. Li, "Revealing gene regulation-based neural network computing in bacteria," *Biophysical Reports*, vol. 3, no. 3, pp. 1–21, Sep. 2023. DOI: 10.1016/j.bpr.2023.100118.

[171] S. Somathilaka, A. Ratwatte, S. Balasubramaniam, M. C. Vuran, W. Srisa-an, and P. Liò, "Wet TinyML: Chemical Neural Network Using Gene Regulation and Cell Plasticity," arXiv, cs.NE, Mar. 2024, pp. 1–7. DOI: 10.48550/ARXIV.2403.08549.

[172] J. Kim, J. J. Hopfield, and E. Winfree, "Neural network computation by in vitro transcriptional circuits," in *18th International Conference on Neural Information Processing Systems (NIPS 2004)*, ser. NIPS'04, Vancouver, Canada: MIT Press, Dec. 2004, pp. 681–688.

[173] L. Qian, E. Winfree, and J. Bruck, "Neural network computation with DNA strand displacement cascades," *Nature*, vol. 475, no. 7356, pp. 368–372, Jul. 2011. DOI: 10.1038/nature10262.

[174] D. Soloveichik, G. Seelig, and E. Winfree, "DNA as a universal substrate for chemical kinetics," *Proceedings of the National Academy of Sciences (PNAS)*, vol. 107, no. 12, pp. 5393–5398, Mar. 2010. DOI: 10.1073/pnas.0909380107.

[175] L. Rizik, L. Danial, M. Habib, R. Weiss, and R. Daniel, "Synthetic neuromorphic computing in living cells," *Nature Communications*, vol. 13, no. 1, pp. 1–17, Sep. 2022. DOI: 10.1038/s41467-022-33288-8.

[176] A. G. Becerra, M. Gutiérrez, and R. Lahoz-Beltra, "Computing within bacteria: Programming of bacterial behavior by means of a plasmid encoding a perceptron neural network," *Biosystems*, vol. 213, pp. 1–16, Mar. 2022. DOI: 10.1016/j.biosystems.2022.104608.

[177] C. Chen, R. Wu, and B. Wang, "Development of a neuron model based on DNAzyme regulation," *RSC Advances*, vol. 11, no. 17, pp. 9985–9994, 2021. DOI: 10.1039/d0ra10515e.

[178] A. J. van der Linden et al., "DNA Input Classification by a Riboregulator-Based Cell-Free Perceptron," *ACS Synthetic Biology*, vol. 11, no. 4, pp. 1510–1520, Apr. 2022. DOI: 10.1021/acssynbio.1c00596.

[179] S. Okumura et al., "Nonlinear decision-making with enzymatic neural networks," *Nature*, vol. 610, no. 7932, pp. 496–501, Oct. 2022. DOI: 10.1038/s41586-022-05218-7.

[180] X. Xiong et al., "Molecular convolutional neural networks with DNA regulatory circuits," *Nature Machine Intelligence*, vol. 4, no. 7, pp. 625–635, Jul. 2022. DOI: 10.1038/s42256-022-00502-7.

[181] K. Sarkar, D. Bonnerjee, R. Srivastava, and S. Bagh, "A single layer artificial neural network type architecture with molecular engineered bacteria for reversible and irreversible computing," *Chemical Science*, vol. 12, no. 48, pp. 15 821–15 832, 2021. DOI: 10.1039/d1sc01505b.

[182] L. Smirnova et al., "Organoid intelligence (OI): the new frontier in biocomputing and intelligence-in-a-dish," *Frontiers in Neuroscience*, vol. 1, pp. 1–23, Feb. 2023. DOI: 10.3389/fsci.2023.1017235.

[183] X. Li, L. Rizik, V. Kravchik, M. Khoury, N. Korin, and R. Daniel, "Synthetic neural-like computing in microbial consortia for pattern recognition," *Nature Communications*, vol. 12, no. 1, pp. 1–12, May 2021. DOI: 10.1038/s41467-021-23336-0.

[184] A. Agiza et al., "Digital circuits and neural networks based on acid-base chemistry implemented by robotic fluid handling," *Nature Communications*, vol. 14, no. 1, pp. 1–9, Jan. 2023. DOI: 10.1038/s41467-023-36206-8.

[185] A. Pandi et al., "Metabolic perceptrons for neural computing in biological systems," *Nature Communications*, vol. 10, no. 1, pp. 1–13, Aug. 2019. DOI: 10.1038/s41467-019-11889-0.

[186] R. Sawlekar and G. Nikolakopoulos, "A Survey of DNA-based Computing Devices and their Applications," in *European Control Conference (ECC 2021)*, Delft, Netherlands: IEEE, Jun. 2021, pp. 769–774. DOI: 10.23919/ecc54610.2021.9654895.

[187] K. M. Cherry and L. Qian, "Scaling up molecular pattern recognition with DNA-based winner-take-all neural networks," *Nature*, vol. 559, no. 7714, pp. 370–376, Jul. 2018. DOI: 10.1038/s41586-018-0289-6.

[188] K. R. Rodriguez, N. Sarraf, and L. Qian, "A Loser-Take-All DNA Circuit," *ACS Synthetic Biology*, vol. 10, no. 11, pp. 2878–2885, Oct. 2021. DOI: 10.1021/acssynbio.1c00318.

[189] R. T. Nagipogu and J. H. Reif, "NeuralCRNs: A Natural Implementation of Learning in Chemical Reaction Networks," arXiv, cs.ET, Aug. 2024, pp. 1–42. DOI: 10.48550/ARXIV.2409.00034.

[190] S. Angerbauer, F. Enzenhofer, T. Pankratz, M. Hamidovic, A. Springer, and W. Haselmayr, "Novel Nano-Scale Computing Unit for the IoBNT: Concept and Practical Considerations," *IEEE Transactions on Molecular, Biological and Multi-Scale Communications*, pp. 549–565, 2024. DOI: 10.1109/tmbmc.2024.3397050.

[191] S. Angerbauer, N. Tuccitto, G. T. Sfrazzetto, R. Santonocito, and W. Haselmayr, "Investigation of Different Chemical Realizations for Molecular Matrix Multiplications," *IEEE Transactions on Molecular, Biological and Multi-Scale Communications*, vol. 10, no. 3, pp. 464–469, Sep. 2024. DOI: 10.1109/tmbmc.2024.3436905.

[192] S. Angerbauer, W. Haselmayr, F. Enzenhofer, T. Pankratz, R. Khanzadeh, and A. Springer, "Molecular Nano Neural Networks (M3N):In-Body Intelligence for the IoBNT," TechRxiv, preprint, Oct. 2023, pp. 1–7. DOI: 10.36227/techrxiv.24427435.v1.

[193] M. Uzun, K. B. Ikiz, and M. Kuscu, "Molecular Communication Channel as a Physical Reservoir Computer," arXiv, cs.ET, Apr. 2025, pp. 1–8. DOI: 10.48550/ARXIV.2504.17022.

[194] S. Stepney, "Physical reservoir computing: a tutorial," *Natural Computing*, vol. 23, no. 4, pp. 665–685, Nov. 2024. DOI: 10.1007/s11047-024-09997-y.

[195] Z. W. El-Hajj and E. B. Newman, "How much territory can a single E. coli cell control?" *Frontiers in Microbiology*, vol. 6, pp. 1–12, Apr. 2015. DOI: 10.3389/fmicb.2015.00309.

[196] M. Prakash and N. Gershenfeld, "Microfluidic Bubble Logic," *Science*, vol. 315, no. 5813, pp. 832–835, Feb. 2007. DOI: 10.1126/science.1136907.

[197] S. Szott et al., "Wi-Fi Meets ML: A Survey on Improving IEEE 802.11 Performance with Machine Learning," *IEEE Communications Surveys & Tutorials*, vol. 24, no. 3, pp. 1843–1893, Jul. 2022. DOI: 10.1109/COMST.2022.3179242.

[198] S. Angerbauer, F. Enzenhofer, H. Gattringer, A. Springer, and W. Haselmayr, "A Molecular Analog-to-Digital Converter," TechRxiv, cs, Jun. 2024, pp. 1–7. DOI: 10.36227/techrxiv.171777814.47534874/v1.

[199] B. A. Holt and G. A. Kwong, "Protease circuits for processing biological information," *Nature Communications*, vol. 11, no. 1, pp. 1–12, Oct. 2020. DOI: 10.1038/s41467-020-18840-8.

[200] T. Huynh, B. Sun, L. Li, K. P. Nichols, J. L. Koyner, and R. F. Ismagilov, "Chemical Analog-to-Digital Signal Conversion Based on Robust Threshold Chemistry and Its Evaluation in the Context of Microfluidics-Based Quantitative Assays," *Journal of the American Chemical Society*, vol. 135, no. 39, pp. 14 775–14 783, Sep. 2013. DOI: 10.1021/ja4062882.

[201] M. Moškon, Ž. Pušnik, L. Stanovnik, N. Zimic, and M. Mraz, "A computational design of a programmable biological processor," *Biosystems*, vol. 221, pp. 1–12, Nov. 2022. DOI: 10.1016/j.biosystems.2022.104778.

[202] A. B. Carlson, P. B. Crilly, and J. C. Rutledge, *Communication Systems: An Introduction to Signals and Noise in Electrical Communication*, 4th ed. New York City, NY: McGraw-Hill, 2002, p. 850.

[203] M. Nauta et al., "From Anecdotal Evidence to Quantitative Evaluation Methods: A Systematic Review on Evaluating Explainable AI," *ACM Computing Surveys*, vol. 55, no. 13s, pp. 1–42, Jul. 2023. DOI: 10.1145/3583558.

[204] M. T. Ribeiro, S. Singh, and C. Guestrin, ""Why Should I Trust You?": Explaining the Predictions of Any Classifier," in *22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Francisco, CA: ACM, Aug. 2016, pp. 1135–1144. DOI: 10.1145/2939672.2939778.

[205] S. M. Lundberg and S.-I. Lee, "A Unified Approach to Interpreting Model Predictions," in *31st International Conference on Neural Information Processing Systems (NIPS 2017)*, Long Beach, CA: Curran Associates Inc., Dec. 2017, pp. 4768–4777.

[206] A. Binder, G. Montavon, S. Lapuschkin, K.-R. Müller, and W. Samek, "Layer-Wise Relevance Propagation for Neural Networks with Local Renormalization Layers," in *Artificial Neural Networks and Machine Learning*, A. E.P. Villa, P. Masulli, and A. J. Pons Rivero, Eds., Barcelona, Spain: Springer International Publishing, 2016, pp. 63–71. DOI: 10.1007/978-3-319-44781-0_8.

[207] A. Shrikumar, P. Greenside, and A. Kundaje, "Learning important features through propagating activation differences," in *34th International Conference on Machine Learning (ICML 2017)*, Sydney, Australia: JMLR.org, Aug. 2017, pp. 3145–3153.

[208] T. N. Mundhenk, B. Y. Chen, and G. Friedland, "Efficient Saliency Maps for Explainable AI," arXiv, cs.CV, Mar. 2019, pp. 1–42. DOI: 10.48550/ARXIV.1911.11293.

[209] A. Wachter-Zeh, B. Mittelstadt, and C. Russell, "Counterfactual Explanations without Opening the Black Box: Automated Decisions and the GDPR," arXiv, cs.AI, Mar. 2017, pp. 1–52. DOI: 10.48550/ARXIV.1711.00399.

[210] L. Breiman, "Random Forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, Oct. 2001. DOI: 10.1023/A:1010933404324.

[211] X. Li, "Explainability of NN-based Detectors in MIMO Molecular Channels," Master's Thesis, School of Electrical Engineering and Computer Science, Berlin, Germany, Jul. 2023.

[212] J. Torres Gómez, P. Hofmann, F. H. P. Fitzek, and F. Dressler, "Explainability of Neural Networks for Symbol Detection in Molecular Communication Channels," *IEEE Transactions on Molecular, Biological and Multi-Scale Communications*, vol. 9, no. 3, pp. 323–328, Sep. 2023. DOI: 10.1109/TMBMC.2023.3297135.

[213] Y. Huang, M. Luo, X. Huang, M. Wen, and C.-B. Chae, "Demystifying Molecular Data-driven Detection with Explainable Artificial Intelligence," *IEEE Wireless Communications Letters*, 2025. DOI: 10.1109/lwc.2025.3554889.

[214] O. T. Basaran, J. Torres Gómez, and F. Dressler, "XAI-Enhanced Bilateral Molecular Communication: Revealing Cancer Microenvironment Dynamics via Extracellular Tumor Vesicles," in *IEEE International Conference on Machine Learning for Communication and Networking (ICMLCN 2025)*, Barcelona, Spain: IEEE, May 2025.

[215] H. C. Berg, *Random Walks in Biology*. Princeton University Press, 1993, p. 152.

[216] P. Hofmann, P. Zhou, C. Lee, M. Reisslein, F. H. Fitzek, and C.-B. Chae, *Dataset for the Simulation of Microfluidic Molecular Communication using OpenFOAM*, 2023. DOI: 10.21227/b71c-4286.

[217] D. Weller, J. J. Tabor, H. Jasak, and C. Fureby, "A tensorial approach to computational continuum mechanics using object-oriented techniques," *Computers in Physics*, vol. 12, no. 6, pp. 620–631, Nov. 1998. DOI: 10.1063/1.168744.

[218] P. Zhou, *DMPPIC*, 2024. [Online]. Available: https://github.com/zhoupengjie/DMPPIC.

[219] P. Zhou, R. Zheng, P. Hofmann, J. A. Cabrera, and F. H. P. Fitzek, "A Diffusive MPPIC Solver in OpenFOAM for Microfluidic Molecular Communication," in *11th ACM International Conference on Nanoscale Computing and Communication (NANOCOM 2024)*, Milan, Italy: ACM, Oct. 2024, pp. 126–127. DOI: 10.1145/3686015.3689420.

[220] A. Wietfeld et al., "Advanced Plaque Modeling for Atherosclerosis Detection Using Molecular Communication," arXiv, cs.ET, Nov. 2024, pp. 1–6. DOI: 10.48550/ARXIV.2411.13241.

[221] P. Hofmann et al., "A Molecular Communication Perspective on Detecting Arterial Plaque Formation," *IEEE Transactions on Molecular, Biological and Multi-Scale Communications*, vol. 10, no. 3, pp. 458–463, Sep. 2024. DOI: 10.1109/tmbmc.2024.3423005.

[222] P. Hofmann et al., *Dataset for Advanced Plaque Modeling for Atherosclerosis Detection using Molecular Communication*, 2024. DOI: 10.21227/mn8d-0h55.

[223] F. Gulec, *CFD Simulation Dataset for Airborne Pathogen Transmission in Turbulent Channels*, 2024. DOI: 10.5281/zenodo.13793238.

[224] F. Gulec, *CFD-Approach-for-the-Characterization-of-Airborne-Transmission-in-Turbulent-MC-Channels*, 2024. [Online]. Available: https : / / github . com / fatihguelec / CFD - Approach - for - the - Characterization - of - Airborne - Transmission - in - Turbulent - MC - Channels.

[225] F. Güleç, F. Dressler, and A. W. Eckford, "A Computational Approach for the Characterization of Airborne Pathogen Transmission in Turbulent Molecular Communication Channels," *IEEE Transactions on Molecular, Biological and Multi-Scale Communications*, vol. 9, no. 2, pp. 124–134, Jun. 2023. DOI: 10.1109/TMBMC.2023.3273193.

[226] F. Gulec, *Distance-Estimation-in-Molecular-Communication*, 2020. [Online]. Available: https : / / github . com / fatihguelec / Distance - Estimation-in-Molecular-Communication.

[227] F. Güleç and B. Atakan, "Fluid dynamics-based distance estimation algorithm for macroscale molecular communication," *Elsevier Nano Communication Networks*, vol. 28, pp. 1–9, Jun. 2021. DOI: 10.1016/j.nancom.2021.100351.

[228] W. Gao, *Dataset in Support of the Southampton Doctoral Thesis "Type-Spread and Multiple-Access Molecular Communications"*, 2023. DOI: http://dx.doi.org/10.5258/SOTON/D2545.

[229] W. Gao, T. Mak, and L.-L. Yang, "Molecular Type Spread Molecular Shift Keying for Multiple-Access Diffusive Molecular Communications," *IEEE Transactions on Molecular, Biological and Multi-*

Scale Communications, vol. 7, no. 1, pp. 51–63, Mar. 2021. DOI: 10.1109/tmbmc.2020.3041182.

[230] W. Gao, "Type-Spread and Multiple-Access Molecular Communications," PhD Thesis, School of Electronics and Computer Science, Southampton, United Kingdom, Feb. 2023.

[231] A. Das, B. Runwal, O. T. Baydas, O. Cetinkaya, and O. B. Akan, *Received Signal Modeling and BER Analysis for Molecular SISO Communications*, 2022. DOI: 10.5281/zenodo.7036057.

[232] A. Das, B. Runwal, O. T. Baydas, O. Cetinkaya, and O. B. Akan, "Received signal modeling and BER analysis for molecular SISO communications," in *9th ACM International Conference on Nanoscale Computing and Communication (NANOCOM 2022)*, Barcelona, Spain, Oct. 2022, pp. 1–6. DOI: 10.1145/3558583.3558854.

[233] X. Lu, *MCFormer*, 2023. [Online]. Available: https://github.com/Xiwen-Lu/MCFormer.

[234] M. Hamidović, S. Angerbauer, D. Bi, Y. Deng, T. Tugcu, and W. Haselmayr, "Microfluidic Systems for Molecular Communications: A Review From Theory to Practice," *IEEE Transactions on Molecular, Biological and Multi-Scale Communications*, vol. 10, no. 1, pp. 147–163, Mar. 2024. DOI: 10.1109/tmbmc.2024.3368768.

[235] K. Gökarslan and E. Çağırıcı, *MoleCom-Gpu*, 2017. [Online]. Available: https://github.com/MoleCom-Gpu/MoleCom-Gpu.

[236] Y. Jian et al., "nanoNS3: A network simulator for bacterial nanonetworks based on molecular communication," *Elsevier Nano Communication Networks*, vol. 12, pp. 1–11, Jun. 2017. DOI: 10.1016/j.nancom.2017.01.004.

[237] G. Wei, P. Bogdan, and R. Marculescu, "Efficient Modeling and Simulation of Bacteria-Based Nanonetworks with BNSim," *IEEE Journal on Selected Areas in Communications*, vol. 31, no. 12, pp. 868–878, Dec. 2013. DOI: 10.1109/JSAC.2013.SUP2.12130019.

[238] B. Drawert, S. Engblom, and A. Hellander, "URDME: a modular framework for stochastic simulation of reaction-transport processes in complex geometries," *BMC Systems Biology*, vol. 6, no. 1, pp. 1–17, Jun. 2012. DOI: 10.1186/1752-0509-6-76.

[239] S. S. Andrews, N. J. Addy, R. Brent, and A. P. Arkin, "Detailed Simulations of Cell Biology with Smoldyn 2.1," *PLoS Computational Biology*, vol. 6, no. 3, pp. 1–10, Mar. 2010. DOI: 10.1371/journal.pcbi.1000705.

[240] S. S. Andrews, "Smoldyn: particle-based simulation with rule-based modeling, improved molecular interaction and a library interface," *Bioinformatics*, vol. 33, no. 5, pp. 710–717, Dec. 2016. DOI: 10.1093/bioinformatics/btw700.

[241] I. Llatser et al., "Exploring the Physical Channel of Diffusion-Based Molecular Communication by Simulation," in *IEEE Global Telecommunications Conference (GLOBECOM 2011)*, Houston, TX: IEEE, Dec. 2011, pp. 1–5. DOI: 10.1109/glocom.2011.6134028.

[242] I. Llatser, D. Demiray, A. Cabellos-Aparicio, D. T. Altilar, and E. Alarcón, "N3Sim: Simulation framework for diffusion-based molecular communication nanonetworks," *Simulation Modelling Practice and Theory*, vol. 42, pp. 210–222, Mar. 2014. DOI: 10.1016/j.simpat.2013.11.004.

[243] A. Noel, K. C. Cheung, R. Schober, D. Makrakis, and A. Hafid, "Simulating with AcCoRD: Actor-based Communication via Reaction–Diffusion," *Elsevier Nano Communication Networks*, vol. 11, pp. 44–75, Mar. 2017. DOI: 10.1016/j.nancom.2017.02.002.

[244] X. Qian, M. Di Renzo, and A. Eckford, *Molecular-Communication*, 2020. [Online]. Available: https://github.com/MuskanM1/Molecular-Communication.

[245] P. Sangani, *Research-in-Molecular-Communication*, 2021. [Online]. Available: https://github.com/Priyank31/Research-in-Molecular-Communication.

[246] D. Patel, *Molecular-Communication*, 2020. [Online]. Available: https://github.com/devshree07/Molecular-Communications.

[247] Y. Shastri, *Molecular-Communication—Model-based-and-Data-Driven-Receiver-Design*, 2020. [Online]. Available: https://github.com/Yesha19/Molecular-Communication---Model-based-and-Data-Driven-Receiver-Design.

[248] H. Birkan, *ANN for Diffusion Channel with Reflecting Spherical to Absorbing Spherical*, MATLAB Central File Exchange, 2017. [Online]. Available: https://www.mathworks.com/matlabcentral/fileexchange/61382-ann-for-diffusion-channel-with-reflecting-spherical-to-absorbing-spherical.

[249] F. Gulec, *Signal-Reconstruction-in-Diffusion-based-Molecular-Communication*, 2018. [Online]. Available: https://github.com/fatihguelec/Signal-Reconstruction-in-Diffusion-based-Molecular-Communication.

[250] B. Atakan and F. Güleç, "Signal reconstruction in diffusion-based molecular communication," *Transactions on Emerging Telecommunications Technologies*, vol. 30, no. 12, pp. 1–14, Dec. 2019. DOI: 10.1002/ett.3699.

[251] F. Gulec, *Mobile-Human-Ad-Hoc-Networks*, 2022. [Online]. Available: https://github.com/fatihguelec/Mobile-Human-Ad-Hoc-Networks.

[252] F. Güleç, B. Atakan, and F. Dressler, "Mobile Human Ad Hoc Networks: A Communication Engineering Viewpoint on Interhuman Airborne Pathogen Transmission," *Elsevier Nano Communication Networks*, vol. 32-33, pp. 1–11, Jun. 2022. DOI: 10.1016/j.nancom.2022.100410.

[253] E. Dong, H. Du, and L. Gardner, "An interactive web-based dashboard to track COVID-19 in real time," *The Lancet Infectious Diseases*, vol. 20, no. 5, pp. 533–534, May 2020. DOI: 10.1016/s1473-3099(20)30120-1.

[254] F. Gulec and A. Eckford, "Stochastic Modeling of Biofilm Formation with Bacterial Quorum Sensing," in *IEEE International Conference on Communications (ICC 2023)*, Rome, Italy: IEEE, May 2023, pp. 4470–4475. DOI: 10.1109/icc45041.2023.10278566.

[255] F. Gulec and A. Eckford, "A Stochastic Biofilm Disruption Model Based on Quorum Sensing Mimickers," *IEEE Transactions on Molecular, Biological and Multi-Scale Communications*, vol. 9, no. 3, pp. 346–350, Sep. 2023. DOI: 10.1109/tmbmc.2023.3292321.

[256] F. Gulec, *Stochastic-Modeling-of-Biofilm-Formation-with-Bacterial-Quorum-Sensing*, 2022. [Online]. Available: https://github.com/fatihguelec/Stochastic-Modeling-of-Biofilm-Formation-with-Bacterial-Quorum-Sensing.

[257] F. Gulec, *Stochastic-Biofilm-Disruption-Model-Based-on-Quorum-Sensing-Mimickers*, 2022. [Online]. Available: https://github.com/fatihguelec/Stochastic-Biofilm-Disruption-Model-Based-on-Quorum-Sensing-Mimickers.

[258] Z. Zhang, *Informer-based*, 2024. [Online]. Available: https://github.com/Zhichao-Zhang-Zjut/Informer-based.

[259] D. Scazzoli, F. Vakilipoor, and M. Magarini, "Molecular communication data augmentation and deep learning based detection," *Elsevier Nano Communication Networks*, vol. 40, pp. 1–12, Jul. 2024. DOI: 10.1016/j.nancom.2024.100510.

[260] L. Grebenstein et al., "Biological Optical-to-Chemical Signal Conversion Interface: A Small-Scale Modulator for Molecular Communications," *IEEE Transactions on NanoBioscience*, vol. 18, no. 1, pp. 31–42, Jan. 2019. DOI: 10.1109/TNB.2018.2870910.

[261] P. Hofmann, J. A. Cabrera, R. Bassoli, and F. H. Fitzek, *Dataset for Analog Network Coding in Molecular Communications: A Practical Implementation*, 2023. DOI: 10.21227/57z0-9q64.

[262] N. Tuccitto, *Dataset of "Experimental Implementation of Molecule Shift Keying for Enhanced Molecular Communication"*, 2023. DOI: 10.21227/krbw-ge80.

[263] T. Nunzio, *Dataset of "Experimental Implementation of Molecule Shift Keying for Enhanced Molecular Communication"*, Dec. 2023. DOI: 10.5281/zenodo.10390855.

[264] M. Bartunik, *Channel Parameter Studies with a Biocompatible Testbed for Molecular Communication*, 2023. DOI: 10.21227/g15d-kz12. [Online]. Available: https://dx.doi.org/10.21227/g15d-kz12.

[265] F. Calì, G. Li-Destri, and N. Tuccitto, *The Data Related to Interfacial Shift Keying Allows a High Information Rate in Molecular Communication*, Oct. 2022. DOI: 10.21227/mj9p-pt58.

[266] F. Calì, G. Li-Destri, and N. Tuccitto, *The Data Related to Interfacial Shift Keying Allows a High Information Rate in Molecular Communication*, version 1.0.0, 2022. DOI: 10.5281/zenodo.7244181.

[267] M. Abbaszadeh, I. Atthanayake, P. J. Thomas, and W. Guo, *Molecular Signal Tracking and Detection Methods in Fluid Dynamic Channels*, 2019. DOI: 10.21227/ynet-ss80.

[268] M. Abbaszadeh, I. Atthanayake, P. J. Thomas, and W. Guo, *Molecular Signal Tracking and Detection Methods in Fluid Dynamic Channels (Method and Data)*, Jan. 2020. DOI: 10.21227/eae4-kg81.

[269] L. Grebenstein et al., *A Molecular Communication Testbed Based on Proton Pumping Bacteria*, 2019. DOI: 10.21227/3zj6-pm05.

[270] V. Walter, D. Bi, A. Salehi-Reyhani, and Y. Deng, *Real-Time Signal Processing via Chemical Reactions for a Microfluidic Molecular Communication System*, 2023. DOI: 10.5281/zenodo.8422465.

[271] M. Scherer et al., *Media Modulation Testbed: Code Package*, 2025. [Online]. Available: https://github.com/SyMoCADS/Media_Modulation_Testbed/.

[272] M. Scherer et al., *Closed-Loop Long-Term Experimental Molecular Communication System*, Feb. 2025. DOI: 10.5281/zenodo.13898880.

[273] F. Calì, S. Barreca, G. Li-Destri, A. Torrisi, A. Licciardello, and N. Tuccitto, "Experimental Implementation of Molecule Shift Keying

for Enhanced Molecular Communication," *IEEE Transactions on Molecular, Biological and Multi-Scale Communications*, vol. 10, no. 1, pp. 175–184, Mar. 2024. DOI: 10.1109/tmbmc.2024.3368759.

[274] P. Hofmann, J. A. Cabrera, R. Bassoli, and F. H. P. Fitzek, "Analog Network Coding in Molecular Communications: A Practical Implementation," in *IEEE Global Communications Conference (GLOBECOM 2023)*, Kuala Lumpur, Malaysia: IEEE, Dec. 2023, pp. 571–576. DOI: 10.1109/globecom54140.2023.10437513.

[275] M. Bartunik, J. Teller, G. Fischer, and J. Kirchner, "Channel Parameter Studies with a Biocompatible Testbed for Molecular Communication: Methods and Data," TechRxiv, report, Apr. 2023. DOI: 10.36227/techrxiv.22674685.v1.

[276] F. Calì, G. Li-Destri, and N. Tuccitto, "Interfacial Shift Keying Allows a High Information Rate in Molecular Communication: Methods and Data," *IEEE Transactions on Molecular, Biological and Multi-Scale Communications*, vol. 9, no. 3, pp. 300–307, Sep. 2023. DOI: 10.1109/tmbmc.2023.3290076.

[277] M. Abbaszadeh, I. Atthanayake, P. J. Thomas, and W. Guo, "Molecular Signal Tracking and Detection Methods in Fluid Dynamic Channels," *IEEE Transactions on Molecular, Biological and Multi-Scale Communications*, vol. 6, no. 2, pp. 151–159, Nov. 2020. DOI: 10.1109/tmbmc.2020.3009899.

[278] L. Grebenstein et al., "A Molecular Communication Testbed Based on Proton Pumping Bacteria: Methods and Data," *IEEE Transactions on Molecular, Biological and Multi-Scale Communications*, vol. 5, no. 1, pp. 56–62, Oct. 2019. DOI: 10.1109/tmbmc.2019.2957783.

[279] V. Walter, D. Bi, and Y. Deng, *MolCommUI*, 2023. [Online]. Available: https://github.com/kcl-yansha/MolCommUI.

[280] L. Brand et al., "Closed Loop Molecular Communication Testbed: Setup, Interference Analysis, and Experimental Results," in *59th IEEE International Conference on Communications (ICC 2024)*, Denver, CO: IEEE, Jun. 2024, pp. 4805–4811. DOI: 10.1109/icc51166.2024.10622231.

[281] L. Brand et al., "Switchable Signaling Molecules for Media Modulation: Fundamentals, Applications, and Research Directions," arXiv, cs.NI 2302.10356, Feb. 2023, pp. 1–7. DOI: 10.48550/arXiv.2302.10356.

[282] M. Scherer et al., "Closed-Loop Long-Term Experimental Molecular Communication System," arXiv, cs.ET, Feb. 2025, pp. 1–30. DOI: 10.48550/ARXIV.2502.00831.

[283] A. Das, B. Runwal, O. Cetinkaya, and O. B. Akan, *Channel Estimation and Performance Analysis of SISO Molecular Communications - Version v2*, 2021. DOI: 10.5281/zenodo.5701559.

[284] M. Thakkar, *molecular_communication*, 2020. [Online]. Available: https : / / www . kaggle . com / datasets / mithilesh16 / molecular - communication.

[285] K. Gebru et al., "Datasheets for datasets," *Communications of the ACM*, vol. 64, no. 12, pp. 86–92, Dec. 2021. DOI: 10.1145/3458723.

[286] H. Brownarchive, "Is this the End of Animal Testing?" *MIT Technology Review*, Jun. 2024.

[287] L. Ewart et al., "Performance assessment and economic analysis of a human Liver-Chip for predictive toxicology," *Nature Communications Medicine*, vol. 2, no. 1, pp. 1–16, Dec. 2022. DOI: 10.1038/s43856-022-00209-1.

[288] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The Bulletin of Mathematical Biophysics*, vol. 5, no. 4, pp. 115–133, Dec. 1943. DOI: 10.1007/bf02478259.

[289] S. Skansi, *Introduction to Deep Learning: From Logical Calculus to Artificial Intelligence*. Cham, Switzerland: Springer, 2018. DOI: 10.1007/978-3-319-73004-2.

[290] Y. Wu et al., "Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation," arXiv, cs.CL, Sep. 2016. DOI: 10.48550/ARXIV.1609.08144.

[291] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *27th International Conference on Neural Information Processing Systems (NIPS 2014)*, Montréal, Canada: MIT Press, Dec. 2014, pp. 3104–3112.

[292] K. Cho et al., "Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation," arXiv, cs.CL, Sep. 2014, pp. 1–15. DOI: 10.48550/ARXIV.1406.1078.

[293] S. Levy, "8 Google Employees Invented Modern AI. Here's the Inside Story," *Wired*, Mar. 2024.

[294] J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. Dauphin, "Convolutional Sequence to Sequence Learning," arXiv, cs.CL, Jul. 2017, pp. 1–15. DOI: 10.48550/ARXIV.1705.03122.

[295] S. Sukhbaatar, A. Szlam, J. Weston, and R. Fergus, "End-to-end memory networks," in *28th International Conference on Neural Information Processing Systems (NIPS 2015)*, Montréal, Canada: MIT Press, Dec. 2015, pp. 2440–2448.

[296] M.-T. Luong, H. Pham, and C. D. Manning, "Effective Approaches to Attention-based Neural Machine Translation," arXiv, cs.CL, Sep. 2015, pp. 1–11. DOI: 10.48550/ARXIV.1508.04025.

[297] J. Zhou, Y. Cao, X. Wang, P. Li, and W. Xu, "Deep Recurrent Models with Fast-Forward Connections for Neural Machine Translation," arXiv, cs.CL 1606.04199, Jul. 2016. DOI: 10.48550/arXiv.1606.04199.

[298] D. Britz, A. Goldie, M.-T. Luong, and Q. Le, "Massive Exploration of Neural Machine Translation Architectures," arXiv, cs.CL, Mar. 2017. DOI: 10.48550/ARXIV.1703.03906.

[299] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, "Deep Reinforcement Learning: A Brief Survey," *IEEE Signal Processing Magazine*, vol. 34, no. 6, pp. 26–38, Nov. 2017. DOI: 10.1109/MSP.2017.2743240.

[300] D. T. Hoang, N. Van Huynh, D. N. Nguyen, E. Hossain, and D. Niyato, *Deep Reinforcement Learning for Wireless Communications and Networking: Theory, Applications and Implementation*. Wiley-IEEE Press, 2023, p. 288.

[301] D. Jurafsky, J. H. Martin, A. Kehler, K. V. Linden, and N. Ward, *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition: An Introduction to ... Hall Series in Artificial Intelligence)*. Stanford, CA: Pearson, 2025.

[302] S. Dorner, S. Cammerer, J. Hoydis, and S. t. Brink, "Deep Learning Based Communication Over the Air," *IEEE Journal of Selected Topics in Signal Processing*, vol. 12, no. 1, pp. 132–143, Feb. 2018. DOI: 10.1109/jstsp.2017.2784180.

[303] H. Ye, G. Y. Li, B.-H. F. Juang, and K. Sivanesan, "Channel Agnostic End-to-End Learning Based Communication Systems with Conditional GAN," in *IEEE Global Communications Conference (GLOBECOM 2018)*, Abu Dhabi, United Arab Emirates, Dec. 2018. DOI: 10.1109/glocomw.2018.8644250.

[304] F. A. Aoudia and J. Hoydis, "Model-Free Training of End-to-End Communication Systems," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 11, pp. 2503–2516, Nov. 2019. DOI: 10.1109/jsac.2019.2933891.

[305] W. Jin, Z. J. Li, L. S. Wei, and H. Zhen, "The improvements of BP neural network learning algorithm," in *5th International Conference on Signal Processing Proceedings (ICSP 2000)*, Beijing, China: IEEE, Aug. 2000. DOI: 10.1109/icosp.2000.893417.

[306] D. E. Rumelhart and J. L. McClelland, "Learning Internal Representations by Error Propagation," in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition: Foundations*, D. E. Rumelhart and J. L. McClelland, Eds., MIT Press, 1987, pp. 318–362.

[307] T. O'Shea and J. Hoydis, "An Introduction to Deep Learning for the Physical Layer," *IEEE Transactions on Cognitive Communications and Networking*, vol. 3, no. 4, pp. 563–575, Dec. 2017. DOI: 10.1109/tccn.2017.2758370.

[308] H. Arjmandi, A. Ahmadzadeh, R. Schober, and M. N. Kenari, "Ion Channel Based Bio-Synthetic Modulator for Diffusive Molecular Communication," *IEEE Transactions on NanoBioscience*, vol. 15, no. 5, pp. 418–432, Jul. 2016. DOI: 10.1109/tnb.2016.2557350.

[309] B. Alberts et al., *Essential Cell Biolog*. New York City, NY: W.W. Norton & Company, 2013. DOI: 10.1201/9781315815015.

[310] A. Goodman et al., "Ten Simple Rules for the Care and Feeding of Scientific Data," *PLoS Computational Biology*, vol. 10, no. 4, pp. 1–5, Apr. 2014. DOI: 10.1371/journal.pcbi.1003542.

[311] S. Sun, Z. Cao, H. Zhu, and J. Zhao, "A Survey of Optimization Methods From a Machine Learning Perspective," *IEEE Transactions on Cybernetics*, vol. 50, no. 8, pp. 3668–3681, Aug. 2020. DOI: 10.1109/tcyb.2019.2950779.

[312] A. Wietfeld et al., "Advanced Plaque Modeling for Atherosclerosis Detection Using Molecular Communication," in *IEEE International Conference on Communications (ICC 2025)*, Montréal, Canada: IEEE, Jun. 2025, pp. 5756–5762.

[313] F. Calì et al., "Fluorescent nanoparticles for reliable communication among implantable medical devices," *Carbon*, vol. 190, pp. 262–275, Apr. 2022. DOI: 10.1016/j.carbon.2022.01.016.

[314] A. Das, B. Runwal, O. Cetinkaya, and O. B. Akan, *Channel Estimation and Performance Analysis of SISO Molecular Communications - Version v1*, 2021. DOI: 10.5281/zenodo.5701433.

[315] A. Noel, *AcCoRD*, 2020. [Online]. Available: https://github.com/adamjgnoel/AcCoRD.

[316] L. Felicetti, M. Femminella, and G. Reali, "A simulation tool for nanoscale biological networks," *Elsevier Nano Communication*

Networks, vol. 3, no. 1, pp. 2–18, Mar. 2012. DOI: 10.1016/j.nancom.2011.09.002.

[317] L. Felicetti, M. Femminella, and G. Reali, "Simulation of molecular signaling in blood vessels: Software design and application to atherogenesis," *Elsevier Nano Communication Networks*, vol. 4, no. 3, pp. 98–119, Sep. 2013. DOI: 10.1016/j.nancom.2013.06.002.

[318] L. A. Harris et al., "BioNetGen 2.2: advances in rule-based modeling," *Bioinformatics*, vol. 32, no. 21, pp. 3366–3368, Jul. 2016. DOI: 10.1093/bioinformatics/btw469.

[319] R. Geyer, M. Stelzner, F. Büther, and S. Ebers, "BloodVoyagerS: Simulation of the Work Environment of Medical Nanobots," in *5th ACM International Conference on Nanoscale Computing and Communication (NANOCOM 2018)*, Reykjavík, Iceland: ACM, Sep. 2018, 5:1–5:6. DOI: 10.1145/3233188.3233196.

[320] R. Geyer, C. Deter, and S. Fischer, "BVS-Vis: a web-based visualizer for BloodVoyagerS," in *7th ACM International Conference on Nanoscale Computing and Communication (NANOCOM 2020)*, Virtual Conference: ACM, Sep. 2020. DOI: 10.1145/3411295.3411300.

[321] R. Wendt and S. Fischer, "MEHLISSA: A Medical Holistic Simulation Architecture for Nanonetworks in Humans," in *7th ACM International Conference on Nanoscale Computing and Communication (NANOCOM 2020)*, Virtual Conference: ACM, Sep. 2020. DOI: 10.1145/3411295.3411305.

[322] R. Wendt, *blood-voyager-s*, 2020. [Online]. Available: https://github.com/RegineWendt/blood-voyager-s.

[323] R. Wendt, *BVS-Vis*, 2020. [Online]. Available: https://github.com/RegineWendt/BVS-Vis.

[324] R. Wendt, *MEHLISSA*, 2024. [Online]. Available: https://github.com/RegineWendt/MEHLISSA.

[325] L. Ebner et al., "BVS-Net: A Networking Tool for Studying THz-based Intra-body Communication Links," in *11th ACM International Conference on Nanoscale Computing and Communication (NANOCOM 2024), Work in Progress Papers (WiP)*, Milan, Italy: ACM, Oct. 2024, pp. 132–133. DOI: 10.1145/3686015.3691639.

[326] R. Wendt, L. Ebner, and J. T. Gomez, *BVS_Net*, 2019. [Online]. Available: https://github.com/tkn-tub/BVS_Net.

[327] J. Hattne, D. Fange, and J. Elf, "Stochastic reaction-diffusion simulation with MesoRD," *Bioinformatics*, vol. 21, no. 12, pp. 2923–2924, Apr. 2005. DOI: 10.1093/bioinformatics/bti431.

[328] J. Hattne, D. Fange, and J. Elf, *MesoRD*, 2005. [Online]. Available: https://sourceforge.net/projects/mesord/.

[329] B. Morgan, *MolComSim*, 2015. [Online]. Available: https://github.com/calypsomatic/MolComSim.

[330] H. B. Yilmaz and C.-B. Chae, "Simulation study of molecular communication systems with an absorbing receiver: Modulation and ISI mitigation techniques," *Simulation Modelling Practice and Theory*, vol. 49, pp. 136–150, Dec. 2014. DOI: 10.1016/j.simpat.2014.09.002.

[331] H. Birkan, *MolecUlar CommunicatIoN (MUCIN) Simulator*, 2024. [Online]. Available: https://www.mathworks.com/matlabcentral/fileexchange/46066-molecular-communication-mucin-simulator.

[332] T. Saiki and T. Nakano, "Design and Implementation of a Multicellular Molecular Communication Simulator," in *Joint 12th International Conference on Soft Computing and Intelligent Systems and 23rd International Symposium on Advanced Intelligent Systems (SCIS and ISIS 2022)*, Ise, Japan: IEEE, Nov. 2022, pp. 1–5. DOI: 10.1109/scisisis55246.2022.10002068.

[333] T. Saiki, S. Imanaka, S. Kobayashi, and T. Nakano, "A General-Purpose Simulation Platform for Multicellular Molecular Communication Systems," *IEEE Transactions on Molecular, Biological and Multi-Scale Communications*, vol. 11, no. 2, pp. 152–165, Jun. 2025. DOI: 10.1109/tmbmc.2025.3544141.

[334] T. Saiki and S. Imanaka, *Multicellular Molecular Communication System Simulator*, 2025. [Online]. Available: https://github.com/ImanakaShohei/MulticellularMolecularCommunicationSystemSimulator.

[335] N. A. Turgut, B. A. Bilgin, and O. B. Akan, "N^4 Sim: The First Nervous NaNoNetwork Simulator With Synaptic Molecular Communications," *IEEE Transactions on NanoBioscience*, vol. 21, no. 4, pp. 468–481, Oct. 2022. DOI: 10.1109/tnb.2021.3118851.

[336] N. A. Turgut, *N4Sim*, 2020. [Online]. Available: https://github.com/nafiturgut/N4Sim.

[337] M. Ander et al., "SmartCell, a framework to simulate cellular processes that combines stochastic approximation with diffusion and localisation: analysis of simple networks," *IEEE Proceedings – Systems Biology*, vol. 1, no. 1, pp. 129–138, Jun. 2004. DOI: 10.1049/sb:20045017.

[338] P. Bauer, S. Engblom, A. Senek, and D. Wilson, *URDME (Version 1.4)*, 2024. [Online]. Available: https://github.com/URDME/urdme.

[339] A. Grimmer, M. Hamidović, W. Haselmayr, and R. Wille, "Advanced Simulation of Droplet Microfluidics," *ACM Journal on Emerging Technologies in Computing Systems*, vol. 15, no. 3, pp. 1–16, Apr. 2019. DOI: 10.1145/3313867.

[340] G. Fink, F. Costamoling, and R. Wille, "MMFT Droplet Simulator: Efficient Simulation of Droplet-based Microfluidic Devices," *Software Impacts*, vol. 14, p. 100 440, Dec. 2022. DOI: 10.1016/j.simpa.2022.100440.

[341] A. Grimmer, M. Hamidović, W. Haselmayr, and R. Wille, *SIMPAC-2022-234*, 2022. [Online]. Available: https://github.com/SoftwareImpacts/SIMPAC-2022-234.

[342] J. P. Drees et al., "Efficient Simulation of Macroscopic Molecular Communication: The Pogona Simulator," in *7th ACM International Conference on Nanoscale Computing and Communication (NANOCOM 2020)*, Virtual Conference: ACM, Sep. 2020. DOI: 10.1145/3411295.3411297.

[343] L. Stratmann, J. P. Drees, F. Bronner, and F. Dressler, "Using Vector Fields for Efficient Simulation of Macroscopic Molecular Communication," *IEEE Transactions on Molecular, Biological and Multi-Scale Communications, Special Section - Advances in Molecular Communication*, vol. 7, no. 2, pp. 73–77, Jun. 2021. DOI: 10.1109/TMBMC.2021.3054930.

[344] J. P. Drees et al., *pogona*, 2020. [Online]. Available: https://github.com/tkn-tub/pogona.