Review

Antonio Ciarlo*, David Bronte Ciriza, Martin Selin, Onofrio M. Maragò, Antonio Sasso, Giuseppe Pesce, Giovanni Volpe* and Mattias Goksör

Deep learning for optical tweezers

https://doi.org/10.1515/nanoph-2024-0013
Received January 9, 2024; accepted April 23, 2024; published online May 23, 2024

Abstract: Optical tweezers exploit light–matter interactions to trap particles ranging from single atoms to micrometer-sized eukaryotic cells. For this reason, optical tweezers are a ubiquitous tool in physics, biology, and nanotechnology. Recently, the use of deep learning has started to enhance optical tweezers by improving their design, calibration, and real-time control as well as the tracking and analysis of the trapped objects, often outperforming classical methods thanks to the higher computational speed and versatility of deep learning. In this perspective, we show how cutting-edge deep learning approaches can remarkably improve optical tweezers, and explore the exciting, new future possibilities enabled by this dynamic synergy. Furthermore, we offer guidelines on integrating deep learning with optical trapping and optical manipulation in a reliable and trustworthy way.

Keywords: optical tweezers; deep learning; optical manipulation

1 Introduction

Optical trapping and optical manipulation exploit light–matter interactions to trap and manipulate various types of micro- and nano-particles. These techniques date back to Arthur Ashkin, who demonstrated in the 1970s that it is possible to levitate microparticles in a fluid using a focused laser beam [1]–[4]. Later, A. Ashkin and coworkers demonstrated that it is also possible to trap particles in 3D using a strongly focused laser beam [4] – a technique now known as optical tweezers [5], [6].

Optical tweezers are now an ubiquitous tool in science, allowing for flexible, non-invasive manipulation of nano- and micro-particles as well as for the measurement of forces acting on them. Both trapping and force measurement using optical tweezers have proved fundamental in fields ranging from statistical mechanics [7]–[11], nanothermodynamics [12], soft matter [13], [14], and biology [15]–[21] to microfabrication [22], [23] and atomic physics [24]–[26]. Different kinds of optical tweezers have been developed to tackle the specific challenges of each application, such as trapping of nanoparticles using plasmons [27], [28] and Raman tweezers [14].

Deep learning is a collection of computer algorithms that can improve and adapt their solutions by learning the rules connecting input and output directly from data [29], solving problems ranging from particle tracking and characterization [30] to protein folding [31] and face recognition [32]. The first steps towards the deep learning revolution were taken in the 1940s with the mathematical modeling of biological neurons by neuroscientist Warren McCulloch and logician Walter Pitts [33]. The recent growth of deep learning has been driven largely by the recent increase in the computational power of processors and the size of datasets, but also by the spread of user-friendly all-purposes deep learning frameworks, such as PyTorch [34], [35] and Keras/TensorFlow [36], [37], which enable quick and easy deployment of deep learning solutions for a wide range of tasks.

Several aspects of optical tweezers that are difficult to study theoretically, either due to the computational cost or because of the high modelling complexity, can now be addressed using deep learning. Deep learning can improve the calculation of optical forces by increasing its speed [38] and even accuracy [39], helping to realistically simulate more complex systems. From an experimental standpoint, deep learning can enhance the calibration of optical tweezers [40] and improve the tracking of trapped particles [41]. Furthermore, recent progress in deep learning is also

***Corresponding authors: Antonio Ciarlo and Giovanni Volpe**, Department of Physics, University of Gothenburg, Gothenburg, Sweden, E-mail: antonio.ciarlo@physics.gu.se (A. Ciarlo), giovanni.volpe@physics.gu.se (G. Volpe). https://orcid.org/0000-0002-4029-7668 (A. Ciarlo). https://orcid.org/0000-0001-5057-1846 (G. Volpe)
David Bronte Ciriza and Onofrio M. Maragò, CNR-IPCF, Istituto per i Processi Chimico-Fisici, Messina, Italy, E-mail: brontecir@gmail.com (D.B. Ciriza), onofrio.marago@cnr.it (O.M. Maragò)
Martin Selin and Mattias Goksör, Department of Physics, University of Gothenburg, Gothenburg, Sweden, E-mail: martin.selin@physics.gu.se (M. Selin), mattias.goksor@physics.gu.se (M. Goksör)
Antonio Sasso and Giuseppe Pesce, Dipartimento di Fisica "Ettore Pancini", Università degli Studi di Napoli Federico II, Naples, Italy, E-mail: antonio.sasso@unina.it (A. Sasso), giuseppe.pesce2@unina.it (G. Pesce)

benefiting the real-time control of optical tweezers [42] and the design optimization [43].

This perspective presents an overview of optical tweezers and deep learning, highlighting their recent collaborative developments. We speculate on possible future innovations resulting from this synergy, particularly by proposing the application of the most advanced deep learning approaches. To conclude, we suggest strategies for those aiming to utilize deep learning in combination with optical trapping and optical manipulation in a reliable and safe way.

2 Optical tweezers

Optical tweezers are an ubiquitous tool in science and they are contributing to the progress of fields like biology, physics, and nanotechnology [5], [6]. As can be seen in Figure 1, the field of optical trapping and optical manipulation is rapidly expanding. Instead, after the first two pioneering experiments in 1970 on optical levitation [1] and in 1986 on the 3D optical trapping of dielectric particles [4], optical tweezers have been widely used in literature, starting from their first use in biology [16], [18]–[20] and culminating in the Nobel Prize to Arthur Ashkin in 2018. Based on light–matter interactions, optical forces can trap particles in the proximity of a focused laser beam. Furthermore, the trapping forces are typically so small that, by employing a trapped particle as a probe, it is possible to measure forces well below those reachable with an atomic force microscope (AFM) and micro-fabricated cantilevers [44]. Despite of the recent progress in optical trapping, there are still many open challenges [6], including the calculation of optical forces, the efficient calibration of an optical trap, the position detection of a trapped particle, and the development of new optical trapping systems.

The calculation of optical forces has typically relied on approximations that depend on the trapping regime defined by size of the particle [5], [45]. The trapping regimes are the geometrical-optics regime, the Rayleigh regime, and the intermediate regime. The *geometrical-optics regime* is valid when the size of the particle is much larger than the wavelength λ_0 of the trapping light. In this case, the wave nature of the light can be neglected and optical forces can

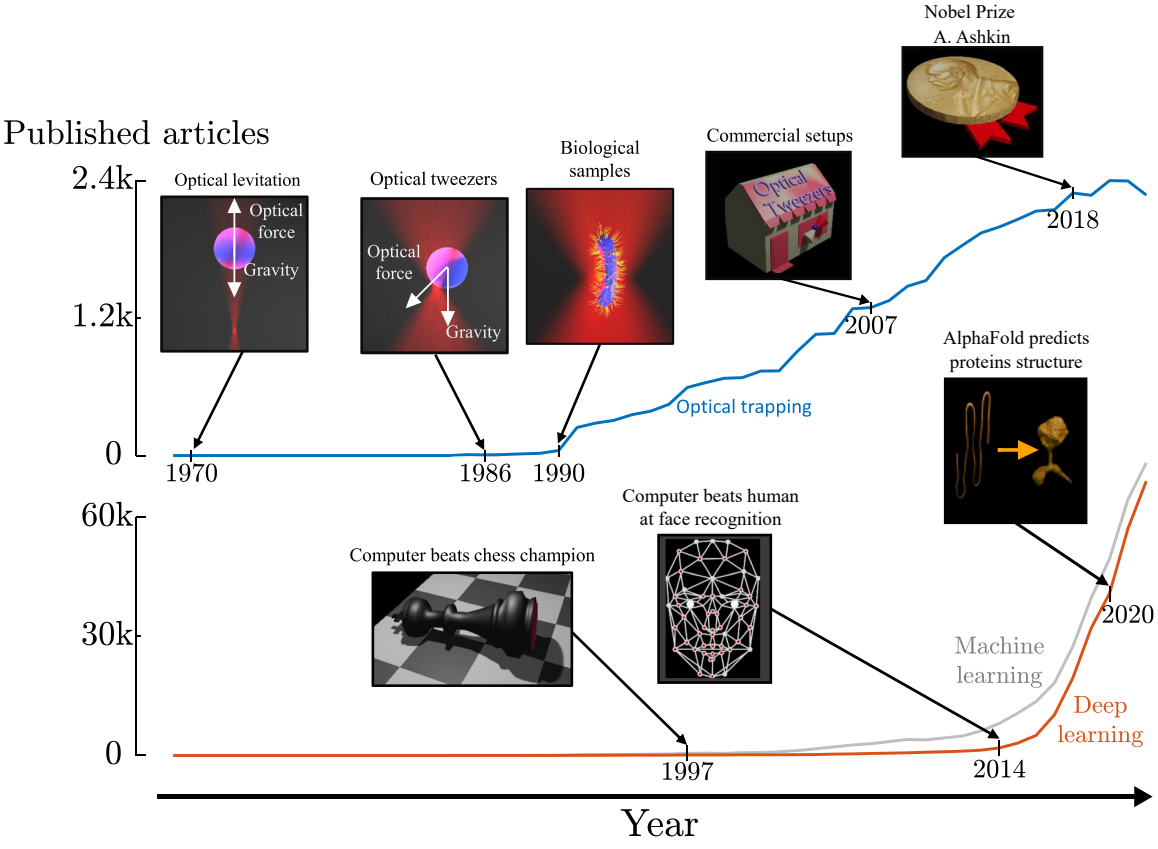

Figure 1: The rise of optical trapping and deep learning in scientific publications. Number of articles published per year that use "optical trapping" (blue line), "machine learning" (gray line), or "deep learning" (orange line) in their title, abstract, or keywords. Milestones in the development of these fields are highlighted with illustrations. Data obtained from Web of Science™ on November 2023.

be calculated using ray optics [46], [47]. Instead, the *Rayleigh regime* occurs when the linear dimensions of the trapped object are much smaller than λ_0. Thus, the trapped object behaves like a dipole and the optical forces are mostly proportional to the gradient of the light intensity [48]. Finally, the *intermediate regime* lays in between, where the linear dimensions of the trapped object are comparable with λ_0. In this case, the optical forces need to be calculated from the electromagnetic fields obtained as an exact solution of the scattering problem, which can be a very complex and computationally intensive process [49]–[51]. Common to all the regimes is that the trapping forces for small displacements from the trapping position can be approximated as a harmonic force

$$F(r) = -k \cdot r, \tag{1}$$

where k is the stiffness of the trap, r is the displacement from the equilibrium position, and $F(r)$ is the optical force.

Calibrating an optical tweezers consists of determining the relation between the position of a particle and the force it experiences. For small displacements from the equilibrium position, it is sufficient to determine the trap stiffness. The traditional approaches to calibration rely on explicit mathematical recipes such as the potential method [52], the autocorrelation method [53], the power spectrum analysis [54], the mean square displacement method, the equipartition method, or the maximum-likelihood-estimator analysis (FORMA) [55]. While these approaches perform well when the field is static, conservative, and a high amount of data are available, they present some limitations when the force field does not satisfy these assumptions.

In optical trapping experiments, the location of the particle is often the most critical parameter. Even though the previously mentioned calibration techniques differ in their approaches, they all rely on this knowledge. There are two main possibilities for tracking the position of the particle. For a single particle in an optical trap, one can use the trapping laser as a probe to determine its position, for instance using a quadrant photodiode (QPD) or a position sensitive detector (PSD). However, when there are multiple particles or multiple traps, interpreting the QPD signal becomes more complex and cameras are typically necessary. These cameras provide a larger view of the experimental system under investigation, containing much more information than the QPD/PSD signals but with the drawback of a lower acquisition rate.

Nowadays, in order to expand the applicability of optical trapping, new techniques to control optical tweezers are being developed. External real-time feedback allows to correct the trapping force by adjusting either the intensity of the light or the position of the trap [56], [57]. Introducing external feedback increases the effective trap stiffness but comes with the drawbacks of a limited bandwidth and of a higher sensitivity to errors in the detection of the position of the particle. To overcome these problems, automatic feedback control mechanisms have been postulated for plasmonic tweezers [28] and realized for intracavity optical trapping [58].

3 Deep learning

Deep learning is a branch of computer science that, by using artificial neural networks, allows computers to learn from data and improve their performance without explicit programming. It is a subset of machine learning, as shown Figure 2, that utilizes artificial neural networks with multiple layers. The term “deep” refers to the use of these multiple layers. Nowadays, its use is growing exponentially after the chess computer DeepBlue defeated the world chess champion Garry Kasparov in 1997 and GaussianFace [59] surpassed humans in face recognition, as shown in Figure 1. Today, deep learning has become a useful tool in science, helping in the prediction of complex systems such as AlphaFold [31], which predicts the 3D structure of proteins starting from the sequences of their amino acids. Typically, deep learning approaches extract hierarchical features from data to realize complex tasks such as image recognition, natural language processing, and speech synthesis with remarkable accuracy and efficiency. They achieve this by automatically learning hierarchical features from raw data, reducing the need for manual feature engineering, whereas traditional machine learning models, like linear regression, principal component analysis, or decision trees, often require explicit feature extraction. This has led to a near-exponential growth in the use of machine learning and, in particular, deep learning [29], as shown in Figure 1.

Deep learning is typically based on deep (i.e., multilayer) artificial neural networks with many trainable parameters that transform input data into output data [29]. These parameters are automatically adjusted during the *training process*, in which the system learns the rules that connect the input data to the desired outputs by operating on known input/output pairs, called training data, using algorithms such as stochastic steepest descent and error backpropagation [60]. Thus, specific problems can be addressed reliably without explicitly knowing the rules connecting input and output, especially when the data to be analyzed closely resemble the training data.

The fundamental building block of neural networks is the artificial neuron [33]. The artificial neuron processes

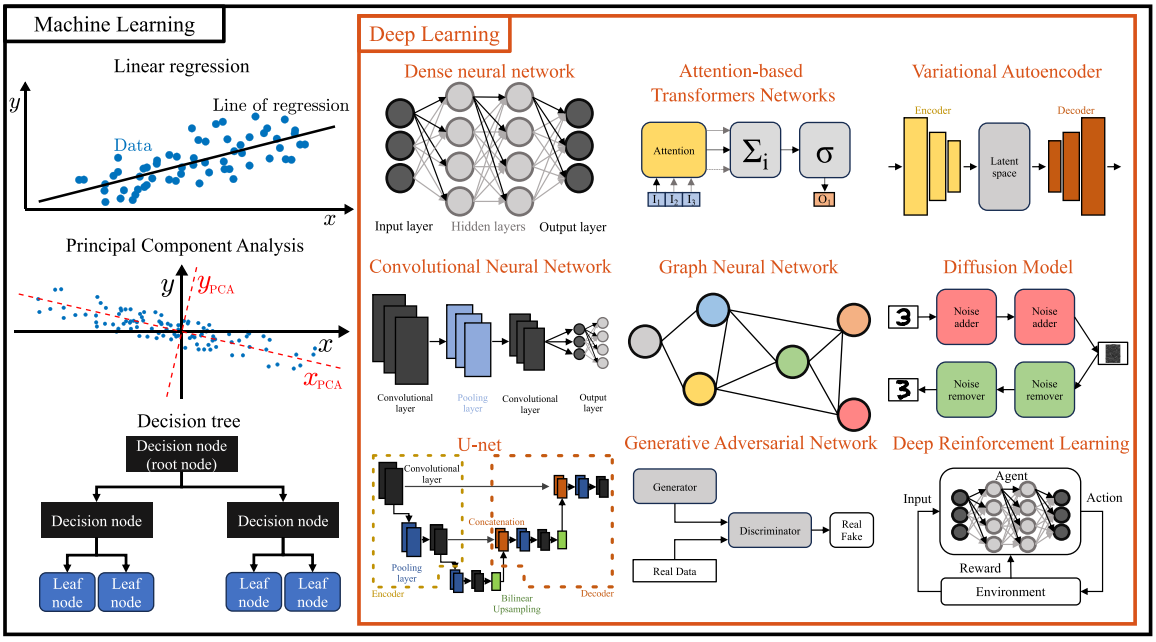

Figure 2: Machine learning and deep learning. Deep learning (orange rectangle) is a subset of machine learning (black rectangle). Machine learning approaches include linear regression, principal component analysis, and decision trees. Deep learning approaches include dense neural networks, convolutional neural networks, U-nets, attention-based transformer networks, graph neural networks, generative adversarial networks, variational autoencoders, diffusion model, and deep reinforcement learning.

its inputs by performing a weighted sum and returning a transformation (typically a nonlinear activation function) of the resulting sum. During the training process, the trainable parameters, often referred to as weights, are tuned to optimize the output of the neuron. Artificial neurons can be connected in layers, with each neuron receiving input from neurons of the previous layer and passing its output to the next layer, forming the most standard artificial neural network.

Deep learning can be implemented through different network structures, i.e., different architectures, and choosing the right one depends on the task at hand. The efficiency and effectiveness of the solution are strongly influenced by the architecture because different problems have distinct data characteristics and complexities. Generally speaking, more complex data require more complex models in terms of the number of parameters needed for fitting and analysis. Moreover, deep learning can be used to generate synthetic data of high quality. To illustrate only the architectures used in the literature for optical tweezers applications and the architectures we propose for potential and new applications, it is convenient to group the different architectures into three groups based on their purposes: data analysis (Dense neural networks, convolutional neural networks, U-nets, recurrent neural networks, transformers networks, graph neural networks), data generation (Generative adversarial networks, variational autoencoders, diffusion models), and decision making (Deep reinforcement learning).

3.1 Data analysis

Dense neural networks (DNNs) are artificial networks in which all the nodes in each layer are connected to all the nodes in the adjacent layers. They have a structure characterized by a first layer referred to as the input layer and a last layer referred to as the output layer (dark circles in Figure 2), and one or more layers in between referred to as hidden layer (gray circles in Figure 2). They are used to deal with tabular data, sequential data, and data with small dimensions. When dealing with high-dimensional data, such as images, the number of connections between the layers increases drastically leading to problems such as overfitting, meaning that the neural network performs exceptionally well on the training data but fails to generalize to new, unseen data.

To deal with high-dimensional data, convolutional neural networks (CNNs) employ 2D layers of neurons partially connected one to the other [61]–[63]. The key layers are the convolutional layers, which use filters to scan the input and

perform convolutional operations, as shown in Figure 2. A filter uses the same weights for different subsets of the input image, thus reducing the number of required trainable parameters and the risk of overfitting. More importantly, each filter corresponds to a feature map that detects a feature in the input data. In this way, the convolutional layer can detect different features of the input for each of its filters. Typically, the image size decreases as it passes through the layers, reducing the computational load and providing access to the information present at different length scales. Often, a dense neural network is added to the final layer of the convolutional neural network to generate an output representing comprehensive information associated with the input, for example, the coordinates of the position of a particle [41]. By reducing the dimensionality of the input, CNNs identify more abstract and high-level features from the data, such as the general shape of a particle or cell, at the expense of low-level features. Therefore, CNNs excel in image detection, recognition, and segmentation [64], [65].

U-nets [66] are characterized by their "U-shaped" design consisting of a contracting path (encoder) connected to an expanding path (decoder) connected also by skip connections, as shown in Figure 2. These skip connections bridge earlier and later layers in the network, ensuring that both low-level and high-level features are effectively combined by enabling the direct transfer of feature maps. The contracting path reduces the dimension of the input thanks to several convolutional layers, capturing and summarizing local information to learn high-level features. Instead, the expanding path consists of transposed convolutions (or deconvolutions) to up-sample the feature map restoring the dimension of the input. Through the skip connections, the expanding path receives high-resolution feature maps preserving the low-level features in the final output. Between the contracting and expanding path, i.e., at the bottom of the U shape, there is a bottleneck layer having the most abstract and high-level representation of the input data. Even if U-nets solve the loss of low-level features, they still need, like any CNN, a large number of diverse training data to reach good performances and acceptable reliability. For example, U-Nets have achieved significant success in the analysis of brain tumors images from MRI scans [67], denoising astronomical images [68], and characterizing the microstructure of samples imaged with scanning electron microscopy [69].

Unlike the previous architectures, recurrent neural networks (RNNs) retain and utilize information from previous time steps [70]. For this reason, RNNs incorporate memory gates that adjust their internal state based on prior data [55]. A fundamental characteristic of RNNs is their capability to establish recurrent connections, generating a feedback loop within the network. This enables the information to circulate within the network, making it responsive to the order and timing of input data. However, conventional recurrent neural networks encounter constraints resulting in difficulties in capturing prolonged dependencies effectively, including the vanishing gradient problem [71]. To address this issue, advanced models such as long short-term memory (LSTM) [72] and gated recurrent unit (GRU) [73] networks have been developed. These structures contain more advanced memory gates that can select and retain information over extended sequences, making them especially effective in tasks such as speech recognition, where long-term contextual information is crucial. Overall, RNNs excel in applications where the sequence of data elements is important, such as natural language processing [74], protein analysis [75], [76], optical coherence tomography data segmentation [77], and adaptive optics control [78].

Attention-based transformers networks (ATNs) employ self-attention mechanisms to analyze sequential data, enabling them to identify how even distant elements in the sequence interact and influence each other [79], as shown in Figure 2. The first step is to add some position information to the sequential input data through positional encoding (typically creating a vector applying the cosine function for every odd index of the input data and a vector applying the sine function for every even index). Then, an encoder layer maps all the input sequences into a continuous representation. It is composed of 2 sub-modules: the multi-headed attention and the dense neural network. The multi-headed attention layer allows the model to focus on specific elements of the input data, assigning them different levels of importance during the learning process thanks to a scoring matrix (determining the amount of attention one element of the input should have on the others). The word "multi-headed" refers to the fact that this layer analyzes simultaneously the input with different attention sub-modules called "heads". The dense neural network, which follows multi-headed attention, enhances the representations of the input elements to learn higher-level information. After the encoder, its output is sent to a decoder that has two multi-headed attention layers followed by a dense neural network. The first multi-headed attention layer receives the output of the encoder after positional encoding and sends its output to the second multi-headed layer that combines it directly with the output of the encoder (without positional encoding) allowing the decoder to understand which encoder input is relevant to put a focus on. In the end, the dense neural

network classifies the input and chooses the highest probability prediction for the output. Transformers have proved themselves very useful in language modeling [80], text generation [81], and image captioning [82].

Graph neural networks (GNNs) are designed to analyze data organized as graphs, capturing complicated relationships within them [83]–[85], as shown in Figure 2. A graph comprises a set of nodes (or vertices) linked by edges (or links). The nodes, in which information is stored within a vector known as a feature vector, correspond to the input data, while the edges represent the corresponding dependencies. The process begins by taking the input graph and passing it through a sequence of neural networks. This transformation transforms the structure of the input graph into a graph embedding (i.e., into vectors), preserving essential details about nodes, edges, and overall context. Next, the feature vectors associated with the nodes are passed to a neural network layer. These features are combined and aggregated within this layer, and the resulting information is then passed on to the next layer in the network. In this way, the GNN updates node representations iteratively to capture information from neighboring nodes, often by following a series of message-passing steps. During these steps, each node aggregates information from its neighbors, applies a learnable function, and updates its representation accordingly. The first obvious application of GNNs is the classification of nodes and the completion of graphs with missing links. More interesting applications in which GNNs excel are, for example, web recommendation systems [86], traffic prediction [87], and protein–protein interactions [88].

3.2 Data generation

Generative adversarial networks (GANs) create high-quality synthetic data by using a specific method called adversarial training [89]. This method uses two neural networks: the generator, which produces the synthetic data, and the discriminator, which verifies whether the data are real or fake, as shown in Figure 2. The adversarial training improves the synthetic data generation by training the generator and discriminator in alternating steps. First, the generator produces synthetic data from the input data and the discriminator tries to classify them. Following this, by using both real and synthetic data, the discriminator is trained to better classify data. Finally, the generator is updated to produce more realistic data by using the results of the training of the discriminator. This adversarial process continues iteratively until the generator produces synthetic data able to deceive the discriminator. Step-by-step, the generator can produce samples that are almost indistinguishable from real data, making GANs a powerful tool in data augmentation and data synthesis applications. A recent evolution of GANs, called time-series GANs (TGANs), allows the generation of time-series data by taking into account the temporal correlations of the time-series data [90]. However, training GANs can be challenging because they might suffer from mode collapse (producing limited diversity in generated samples). GANs are used not only for data generation, but also for image-to-image translation [91], for enhancing the resolution of images [92], and for anomaly detection [93].

Variational autoencoders (VAEs) are generative models that combine deep neural networks with probabilistic modeling to learn representations of data and generate new samples by mapping input data into a continuous latent space [94]. VAEs use deep neural networks to produce a meaningful latent space representation of the input data, where a latent space is a lower-dimensional space in which the input data are mapped into a distribution (typically, a multivariate Gaussian). To do this, VAEs use an encoder and a decoder, as shown in Figure 2. The encoder is a neural network (typically, a dense or convolutional neural network) that extrapolates from the input data the mean (μ) and the variance (σ) of the distribution in the latent space. Once these two parameters are known, the encoder uses them to sample a point (z) from the latent space by using the reparameterization trick following a standard Gaussian distribution, i.e., $z = \mu + \epsilon \cdot \sigma$ with σ Gaussian random noise term. Then, the decoder uses a neural network to reconstruct the original input data from the latent space representation obtained with the encoder. In this way, it takes points from the latent space and generates a new data sample that is similar to the input data one. VAEs have proved useful to reconstruct complex many-body physics [95], for regressions [96], and for music generation [97].

Diffusion models (DMs) are a deep learning architecture created to simulate the evolving changes in data over time or space, emulating the fundamental principles of diffusion processes and allowing a heterogeneous data production [98]. These models add noise or perturbations to the input data during different steps, converting them into an uncertain state, as shown in Figure 2. Subsequently, the model is trained to reverse this process using a neural network to predict and control the noise reduction, gradually restoring the data point to its original or desired state. This approach to noise reduction produces data samples that reflect the underlying trends and variability of the data distribution while ensuring coherence, realism, and high heterogeneity thanks to the randomness of the process. This means that DMs have an exclusive ability to capture

patterns and variations inherent in data distribution. The adaptability of diffusion models cover a wide range of applications, including image generation [99]–[102] and natural language processing [103].

3.3 Decision making

Deep reinforcement learning (DRL) is a deep learning approach that combines deep neural networks with reinforcement learning techniques to learn sequential decision-making in complex environments through trial and error [104], [105]. It is based on reinforcement learning, in which an agent learns to make sequential decisions in an environment to maximize a cumulative reward signal. In DRL, the agent employs a neural network that is trained using feedback from the environment, as shown in Figure 2. This feedback consists of rewards or penalties for the agent based on its actions. Through iterative interactions with the environment, collecting experiences, and updating its neural network, the DRL agent gradually learns an optimal policy or value function, enabling it to make effective decisions in complex and high-dimensional environments. In this way, DRL can do very complex tasks like playing Go [106], driving autonomous vehicles [107], and designing optical multi-layer thin films [108].

4 Deep learning for optical tweezers

The advantages of machine learning, such as simplicity, versatility and speed, enhance optical tweezers by improving particle detection and tracking, trajectory analysis and calibration, optical force calculation, and by enabling tasks such as real-time control of optical traps and new designs. When automated without deep learning, these tasks typically require manual tuning of parameters, low noise measurements, or extremely long calculations. This is undesirable because it is time consuming for the researchers and also risks introducing human biases. In the following subsections, we discuss different cases where deep learning has already been successfully combined with optical trapping and optical manipulation, and we propose new possible applications.

4.1 Particle tracking

In optical tweezers experiments, particle tracking is a key task. Deep neural networks have significantly enhanced this task, notably improving the speed and accuracy of detection, particularly when standard methods are not optimal, such as for irregularly shaped particles or biological samples. Leading tracking algorithms now frequently incorporate convolutional neural networks (CNNs) [30], [41], [109]. These CNNs exhibits greater resistance against noise compared to classical algorithms. This prevents tracking errors due to the presence of noise in the particle video and increases the accuracy of the extracted particle trajectory, as shown in Figure 3a. Nevertheless, acquiring enough training data from experiments is challenging because the true values of the position of the trapped particle are not known and may need to be collected manually or with standard methods. To solve this issue, it is possible to train the algorithms on simulated data [30], [41].

An alternative approach that has shown promise is to exploit the symmetries inherent to the tracking problem. This approach is employed by the recently developed the deep-learning approach called LodeSTAR (Localization and detection from symmetries, translations, and rotations) [110]. By systematically linking the roto-translation of the input image to a corresponding roto-translation of the expected result, it is possible to enable training on small datasets, even with as little as a single image, without the necessity of ground truth. In this way, a single training image is sufficient to train LodeSTAR.

In addition to the position from images of the particle, deep learning can extract more information, such as the particle's size and orientation. For example, deep learning has been recently used to track the orientation of sperms in an optical trap enabling the extraction of the sperm rotation rates [111]. Furthermore, going beyond analyzing images acquired with digital video microscopy, deep learning can potentially be applied also with data acquired with methods based on quadrant-photodiodes (QPDs) or position-sensitive detectors (PSDs). In these cases, deep learning can allow, for example, the extrapolation of the trajectory signal from noisy signals or with frequency higher than the detection bandwidth.

Importantly, deep learning often manages to excel even when standard methods fail. A potential application is to use U-nets to track multiple trapped particles that approach one to the other, as schematically illustrated in Figure 3b, a situation in which standard methods fail and require complex ad-hoc fixing [112]. This is especially relevant for multiple trapped particles and in case of defocusing (due for example to overlapping of two or many particles).

Hypothetically, TGANs could improve the tracking of particles from videos with missing frames or non-constant sampling frequencies. This is because they can generate particle images, such as bright-field images, that respect the

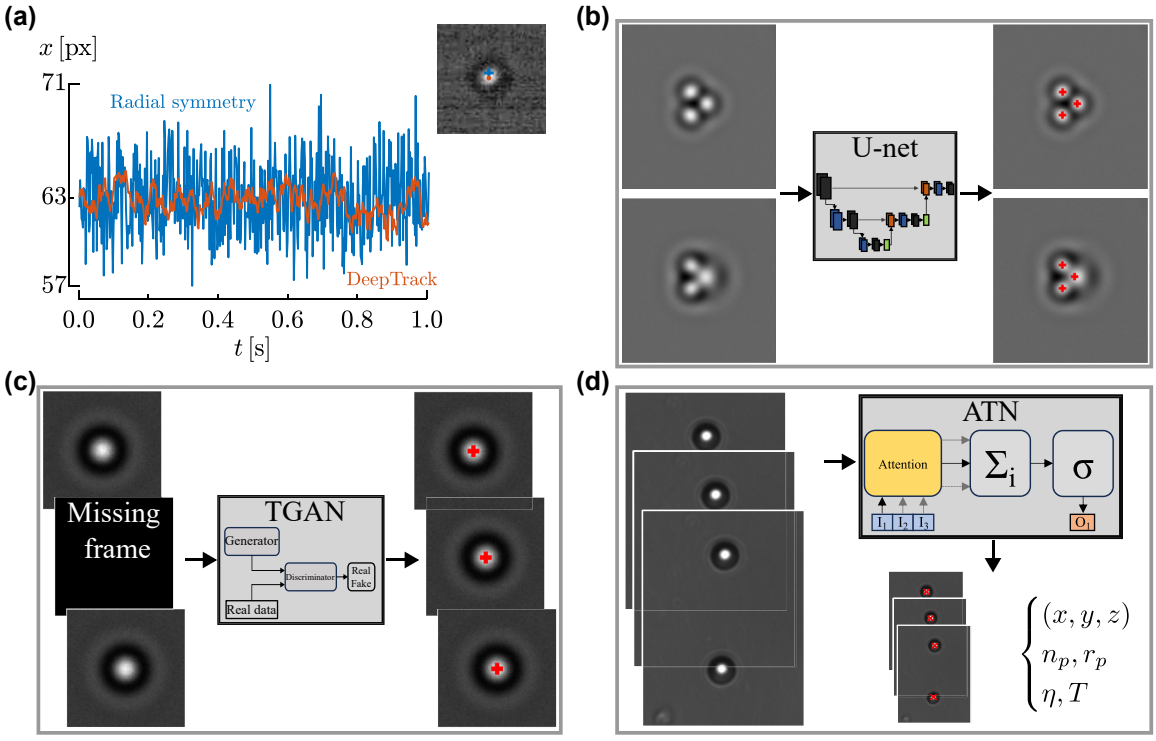

Figure 3: Deep learning for particle tracking. (a) Trajectory of an optically trapped particle obtained from a noisy video by DeepTrack (orange) compared to that obtained with the classical radial symmetry algorithm (blue line). Reproduced from Ref. [41]. (b) As potential application, a U-net can be used to track trapped particles that approach one to the other also when one particle overlaps with the other (defocused particle in the bottom picture on the left). (c) As potential application, a TGAN can fill missing frames in a video file (e.g., due to uneven sampling rate) and track the particles allowing the applications of calibration methods that require a constant sampling rate (e.g., those based on power spectral density, autocorrelation functions, and mean squared displacement). (d) As potential application, an ATN can find the trajectory of optically trapped particles in a video file and use it to determine the physical properties of the particles, such as their refractive index n_p and radius r, as well as information about the immersion media, such as its viscosity η and its temperature T.

temporal correlation of the inputs. As a result, they can generate missing data from the properties of the phenomenon being studied, as schematically shown in Figure 3c. It is possible, for instance, to create a constant sampling rate video from one that is non-constant, enabling the utilization of calibration techniques based on power spectral density, autocorrelation functions, and mean squared displacement. Instead, ATNs can be used to locate trapped particles in a set of many particles and evaluate their properties (such as dimensions and refractive index) or the fluid properties (such as temperature and viscosity) by identifying how distant points of the trajectory of the particle interact and influence one another, as schematically shown in Figure 3d.

4.2 Trajectory analysis and calibration

Deep learning has proven to be an efficient method for analyzing confined particle motion, especially when experimental conditions change, and has proven effective for calibrating optical tweezers in scenarios where traditional methods are inadequate, such as non-conservative force fields and limited data collection situations. Recently, the trajectory analysis with deep learning allowed the estimation of rheological properties by reducing the amount of data needed [113], as schematically shown in Figure 4a. This kind of analysis, which would ordinarily require measuring for several minutes, can now be obtained in a matter of seconds. This result was possible by training the neural network on simulated data, further showcasing the potential of synthetic data to be used to train models. In this case, simulating the training data are both essential to get sufficient amounts of data and relatively simple since the equations of motion of a trapped particle are well understood. When the equations of motion are unknown or too complex to be evaluated numerically, RNNs are a good choice due to their ability to retain and utilize historical information about the particle trajectory.

Deep learning has also been used to analyze particle trajectories within an optical trap measured using from

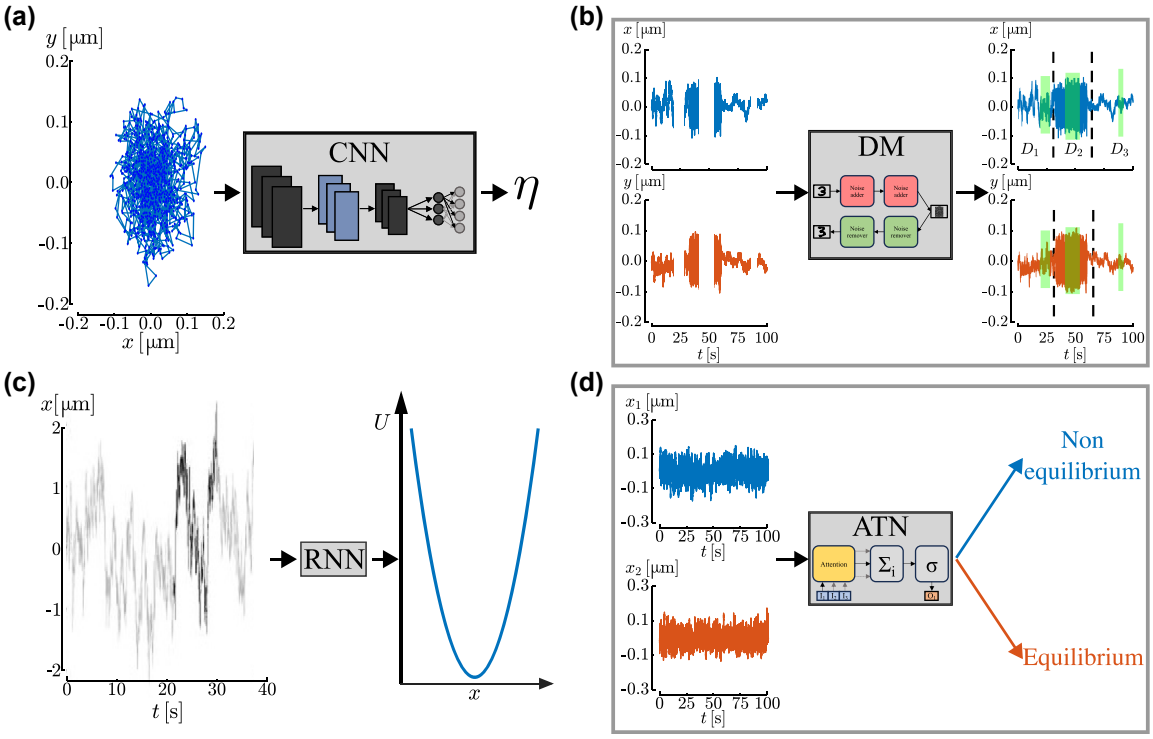

Figure 4: Deep learning for trajectory analysis and calibration. (a) A convolutional neural network is trained on simulated data in order to extrapolate from the particle trajectory the medium viscosity η. Reproduced from Ref. [113]. (b) As potential application, a diffusion model can be used to extract information about the diffusion processes of a trapped particle when there are missing points in the trajectory. (c) The DeepCalib method used a recurrent neural network trained on simulated data to extract the trap stiffness for a microparticle held in a harmonic potential. Reproduced from Ref. [40]. (d) As potential application, an attention-based transformer network can determine whether a trapped particle is in thermal equilibrium or in a non-equilibrium condition.

the forward scattering captured by a quadrant photodiode to discern different kinds of particles [114]. Potentially, deep learning architectures, such as diffusion models, can be utilized to estimate the properties of various diffusion processes experienced by a trapped particle, even when there are missing points in the trajectory. Indeed, the diffusion model can be employed to reconstruct the particle trajectory by effectively filling in the gaps and can estimate the required properties, as schematically illustrated in Figure 4b.

Deep learning can also be used for calibration purposes. This was demonstrated in Ref. [40], where RNNs were used to estimate force fields with limited data available (trajectory length <10 s) for harmonic potentials [40], as shown in Figure 4c, as well as for more complex and time-varying force fields. Recent findings underscore the capabilities of neural networks to go beyond determining the stiffness of optical traps, and to estimate properties of trapped particles such as their refractive index or radii [115]. For these reasons, the use of deep learning methods such as RNNs becomes particularly powerful when studying biological samples, even in challenging scenarios. Moreover, we propose the use of deep learning, specifically transformers network, can determine whether a trapped particle is in thermal equilibrium or not, as shown in Figure 4d, task that is challenging by using standard methods. This is possible because by training the transformer network with data from particles in non-equilibrium states, the attention-based architecture can focus on properties of the trajectory that are peculiar only to out-of-equilibrium particles. Moreover, transformer networks could potentially extract from the trajectories additional useful information, such as the entropy of the system.

4.3 Optical force calculations

Calculating optical forces can be computationally expensive, especially when optical forces require repeated calculations, such as when simulating the Brownian dynamics of an optically trapped particle [116], or for non-Gaussian beams, such as Laguerre–Gauss beams, or for particles with irregular shapes, such as cells. Deep learning offers

a solution to this problem. For example, neural networks have successfully predicted the forces acting on a spherical trapped particle both in the intermediate regime, even for complex beams [38], and in the geometrical-optics approximation [39]. Importantly, the improvement in speed does not come at the expense of accuracy. Quite the opposite, neural networks have also been shown to be able to overcome some artifacts caused by the restricted number of rays used in the geometrical-optics approximation [39]. Simple dense neural networks have been shown to perform well for this task, probably thanks to the low dimensionality of both inputs (e.g., the three coordinates of the particle position as well as some of the particle physical properties) and outputs (e.g., the three components of the force). The enhanced computational speed enables simulations of scenarios previously unattainable utilizing conventional computational methods. For instance, modeling a trapped particle that changes size [38] (Figure 5a), improving the performance and accuracy of geometrical-optics calculations [39] (Figure 5b), exploring the parameter space of an ellipsoid in a double beam configuration [39], simulating the dynamics of a trapped red blood cell [117], or evaluating forces produced by beams with amplitude profiles of arbitrary complexity [118].

As a perspective, DMs and GANs could be used to evaluate the optical forces of complex light fields (also random fields, as speckles field [55], [119]–[121]) from intensity images of the field acquired with a camera, as schematically shown in Figure 5c. This is not possible with standard methods, whereas DMs and GANs can learn how an intensity image relates to a force field during the generation process.

Moreover, CNNs, possibly trained with an adversarial approach, could be used to evaluate the optical forces produced by near-field optical trapping from the 2D design of the substrate, as schematically shown in Figure 5d. Currently this design requires the use of numerical methods that requires a lot of computational power and time for having acceptable results.

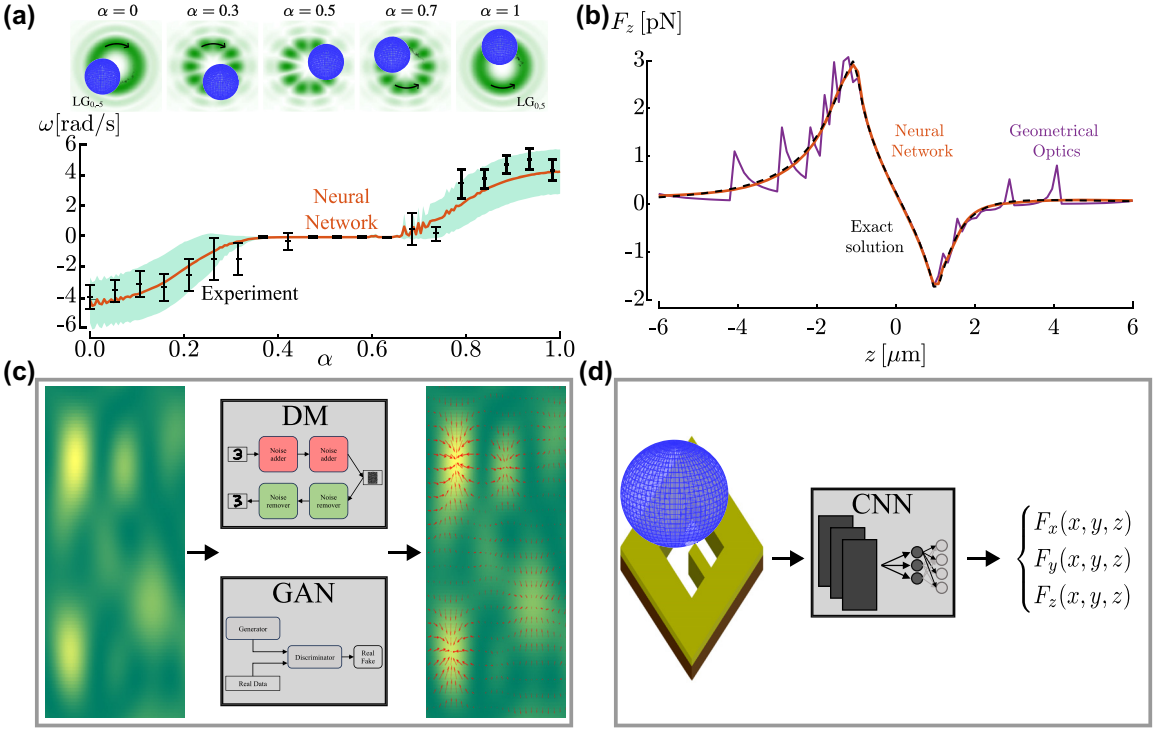

Figure 5: Deep learning for optical force calculation. (a) Experimental (black symbols) and neural-network-simulated (orange line) rotation rates ω as a function of the parameter α of the superposition of two Laguerre–Gaussian beams, $\alpha\,\mathrm{LG}_{0,+5} + (1-\alpha)\,\mathrm{LG}_{0,-5}$. The error bars represent standard errors. Reproduced from Ref. [38]. (b) A dense neural network calculates the optical forces in the geometrical-optics approximation increasing not only the calculation speed but also the accuracy when compared to the conventional geometrical-optics approach. The neural network (orange line) has been trained with data generated with geometrical optics using 100 rays (purple line) and approximates much better the exact solution (black line). Reproduced from Ref. [39]. (c) As potential application, a GNN could evaluate the force field (red arrows in the right panel) directly from images of the optical field (on the left). (d) As potential application, a CNN could be used to evaluate and optimize the trapping force directly from the 2D design of a near-field optical trap.

4.4 Controlling tweezers

Real-time control of optical tweezers using deep learning can improve their operational efficiency and reliability. In 2021 [122], a neural network was trained to guide optically trapped particles to precise target positions while avoiding collisions with other particles and obstacles. The first step in this process is to detect particles in images captured by a camera using a thresholding method. The particle positions are then used to determine the most efficient movements for the captured particle, resulting in its alignment with the desired target. This is done by training a deep reinforcement learning algorithm in a simulated environment. In this way, the NN can determine the most suitable direction for guiding the trapped particle to its target position, all while avoiding potential collisions with other particles, as shown in Figure 6a.

To achieve precise optical tweezers control, digital twins can be coupled with deep learning. Digital twins are virtual models of physical objects, systems, or processes, generated by collecting and integrating data from their corresponding physical counterparts [123]–[125]. By including optical tweezers within a digital twin framework, researchers can virtually execute and manage microscopic objects, such as individual molecules or nanoparticles, with great precision. This enables improved experimentation at the nanoscale and supplies an abundance of real-time data on the behavior and interactions of the objects. This data can then be analyzed by deep-learning algorithms to optimize experimental conditions and swiftly detect complex patterns and trends that may be difficult for human researchers to discern. For example, digital twins and VAE can be used to automatize trapping experiments of only particles with specific properties as schematically shown in Figure 6b. This experiment is not feasible using standard methods because of the need to extrapolate the properties of the particle in real time.

Moreover, Bayesian deep learning can be incorporated into the control structure of optical tweezers to consider possible uncertainties such as sensor noise and variations in particle characteristics. Bayesian deep learning is a deep learning approach using Bayesian modeling, which is a statistical model where the probability is influenced by the belief in the likelihood of a specific outcome [126]. This,

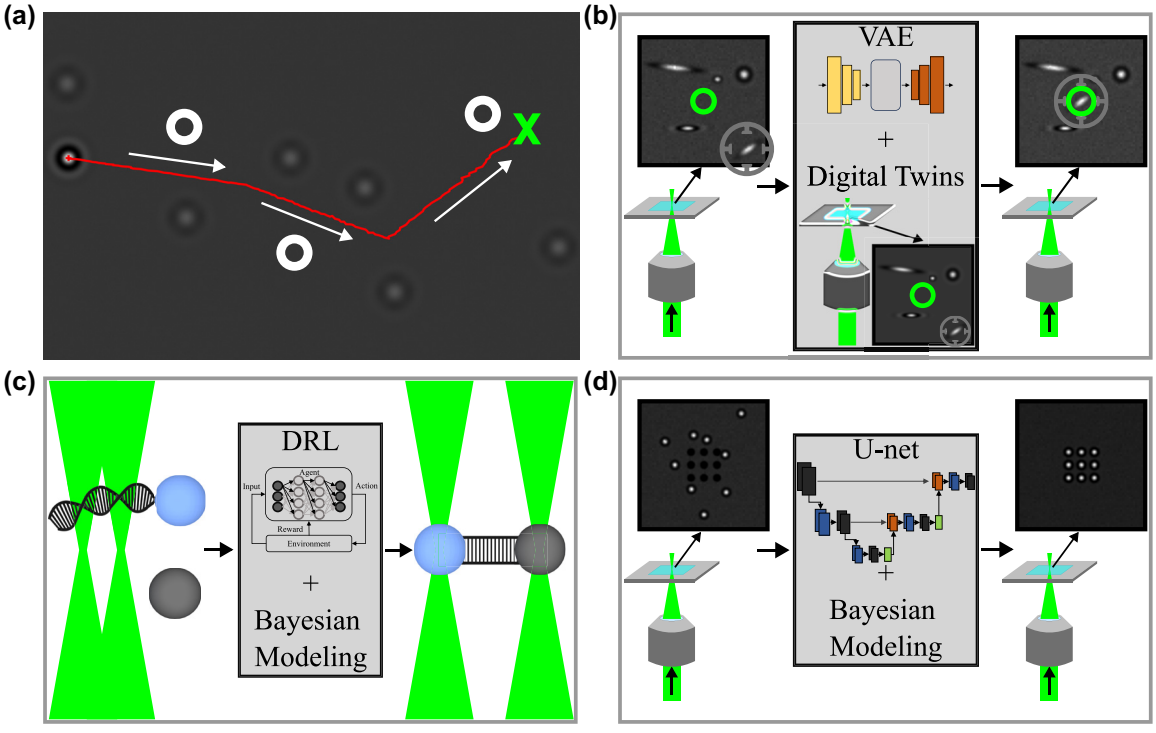

Figure 6: Real-time control of optical tweezers with deep learning. (a) Sketch of a trapped particle moved in real time by a neural network to avoid both physical (defocused particles) and virtual (white hollow circles) obstacles. The red solid line represents the trajectory, the white arrows the direction of the motion, and the green cross the destination point of the particle. Reproduced from Ref. [122]. (b) As potential application, digital twins and VAEs can be used to automatize trapping experiments of only particles with specific properties. (c) As potential application, deep reinforcement learning and Bayesian modeling can be used to automatize the DNA pulling experiment done with two optical traps. (d) As potential application, U-net and Bayesian modeling can improve the process of filling micro-holes in a microfluidic chamber with particles in order to create microstructures.

in turn, enables the precise and adaptable manipulation of particles, for example, for drug delivery, for studying biological processes, or for assembling microstructures, as schematically depicted in Figure 6c and d. The Bayesian framework empowers the system to continuously update its beliefs concerning the state of the particles, thereby enhancing the robustness and efficiency of optical tweezers experiments.

4.5 Designing optical tweezers

Optical tweezers are complex systems whose design can be challenging, especially when using adaptive optics or plasmonic structures. Deep learning has the potential to improve and simplify this design process. However, until now only probabilistic techniques, such as simulated annealing, have been used to design custom nanostructures that help improve the performance of plasmonic trapping [43], [127]. By evaluating the optical force produced by different shapes of the nanoaperture, it is possible to optimize its shape, enhance their electromagnetic field, and, therefore, maximize the trapping force, as shown in Figure 7a.

Deep Learning for designing nanophotonic devices is now widely used [128] and its extension for designing optical tweezers is straightforward. More advanced techniques, such as deep reinforcement learning combined with digital twins, may improve the design of plasmonic devices. For example, DRL might try different shapes of the nanodevice on the digital twin to find the best shape for the best performance. Another way to design optical tweezers is to use a spatial light modulator (SLM) [129] and deep learning algorithms to alter the beam shape. Then, the beam shape can be controlled by a diffusion model that generates the appropriate SLM mask, allowing, for example, the trapping of multiple particles with different beam shapes and/or to compensate the spherical aberrations of the optical system, as schematically shown in Figure 7b.

In addition, digital twins might be used with VAEs to design the optical elements (e.g., trapping lens properties, laser wavelength) to have specific properties of the optical trap such as a specific stiffness of the trap or a trap able to efficiently trap particles that typically are difficult to trap (e.g., gold nanoparticles, quantum dots, low refractive index particles).

5 Guidelines

Considering that many potential applications of deep learning in the optical tweezers domain remain to be developed, we provide here some guidelines. We also address some specific challenges, such as the availability of only limited datasets and the diversity of optical tweezers setups, which complicate the application of the same techniques broadly to different experiments.

The process of applying deep learning to solve an optical tweezers problem can be broadly split into the following steps: 1. Problem description. 2. Data collection/simulation. 3. Architecture selection. 4. Training. 5. Testing. Often, it is necessary to iterate the process multiple times before achieving an acceptable performance.

5.1 Problem description

The first step in implementing any deep learning model is to provide a detailed description of the problem, outlining what is known and what the deep learning model needs to predict. The knowledge of the input and output data, especially which types of data these will contain, is fundamental to choose the proper deep-learning architecture. For instance, the algorithm could use images from a camera as inputs and return the commands to send to the laser the beam properties as output. A key aspect is to define the specific requirements for the sought-after solution. These could

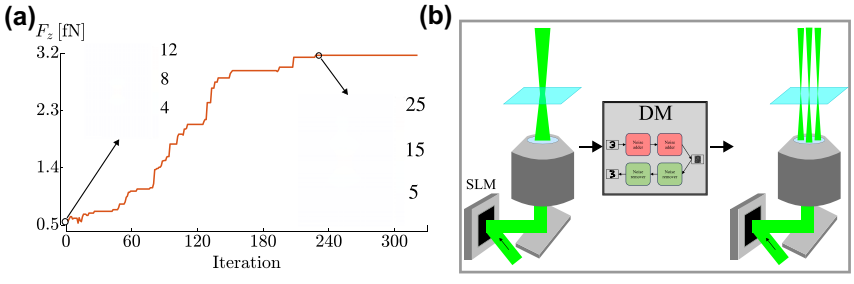

Figure 7: Deep learning for designing optical tweezers. (a) The design of a nanoaperture is optimized using simulated annealing. The algorithm iteratively updates the shape to find the best one for optical trapping. Reproduced from Ref. [127]. (b) As potential application, a diffusion model could be used in combination with a spatial light modulator to trap multiple particles and enhance the focusing, and therefore the trapping force.

be that the output is needed quickly, such as for real-time feedback control, or that the output needs to be accurate, as for image analysis.

When using deep learning to control the experiment, the choice of an architecture able to communicate with the experimental setup and manage the input and output signals is fundamental. A simple solution is to run the deep learning model on a desktop computer connected to the experimental setup. However, more specialized solutions might also be required, for example employing microcontrollers or field programmable gate arrays (FPGAs) with pre-trained neural networks.

Instead, if deep learning is used in data analysis, providing the inputs to the network and retrieving its output is rarely a technical problem. However, it is still recommended to run the algorithm on specialized hardware (GPUs or TPUs), relatively easy and accessible through local computers, servers, or on the cloud.

To enhance the effectiveness and simplify the training of the deep-learning algorithms, the problem needs to be written in as simple terms as possible. For example, the magnitude of the force applied on a sphere in standard optical tweezers depends only on two inputs (radial distance and height from the focus) and not on the three values of the cartesian coordinates (x, y, z) because of symmetry arguments. By exploiting this symmetry in the modelling of the problem, the deep-learning model can perform more accurately and computationally faster, while reducing the requirements of training data and the efforts in training.

Also at the initial stage, it is critical to consider whether deep learning is the best fit for the problem of interest. There are situations in which standard methods perform as well as deep learning with the additional advantage of interpretability and explainability of the results. Instead, a deep-learning model is intrinsically less transparent as it learns through a relatively mysterious training process. In general, deep learning is preferable when there is plenty of data for training or when the relation between the inputs and outputs is too complicated to be described analytically or with simple computational models.

5.2 Data collection/simulation

Any deep learning approach will require training data to fit the parameters of the model and these data will need to be as representative of the problem as possible. Depending on the problem at hand and the chosen architecture, the amount of data required for training the neural network will be different. Typically, the quantity of data should be substantial and diverse, representing the entire variable space of the problem. This can easily be the biggest obstacle when applying deep learning. For example, to track the position of a trapped particle, multiple images in different experimental conditions are required to achieve sufficient generality. Nevertheless, some cutting-edge techniques require only a single sample to complete the training, such as the LodeStar tracking algorithm [110].

In several situations, the training data can be produced through simulations allowing access to potentially infinite amounts of data. Multiple software packages help with this, such as DeepTrack [30], [41], for simulating images of particles, for calculating optical forces [38], [39], and for analysing trajectories [40]. However, the simulated data must be representative of the problem and, to ensure this, a small experimental dataset can be used as a validation set. Sometimes, combinations of simulated and experimental data can improve the learning process. Typically, one would then train the algorithm on the simulated data first and then fine-tune it on the experimental data.

It is important to highlight that the data should be split into three different subsets: a training set used to train the parameters of the architecture; a validation set used to tune its hyperparameters, i.e., the parameters related to the architecture properties (such as number of neurons, number of layers, dimensions of the layers); and a test set used to evaluate the final performance of the trained model on unseen data (these data should not be used during the optimization of the architecture or the training of the model).

Most algorithms employ supervised learning which requires labelled data. This means that the data must be labeled with the ground truth, i.e., each input of the training dataset needs to be associated to a known desired output that the deep-learning model should provide. Knowing the ground truth is challenging and requires the utilization of standard methods or alternative experimental setups. There are also unsupervised techniques (e.g., VAEs) that do not need labeled data. In this case, the preparation of the training dataset is much easier a problem, but the validation of the model becomes more challenging and often requires explicit analysis by the user.

5.3 Architecture selection

The choice of the architecture to use and its hyperparameters is a crucial point because it greatly influences the performance of the model. To assist with the selection of the appropriate architecture, we have compiled in Table 1 the most commonly utilized architectures for typical tasks relevant to optical trapping and optical manipulation. The first things to consider are the task to be achieved and the type of data to be analyzed.

Table 1: Summary of deep learning algorithms suitable for different problems related to optical tweezers. In the last column, we have listed references that deal with the technique on a general level or apply it in the context of optical trapping or a related field.

Problem	Model	References
Particle tracking – single particle	CNNs	[30], [41]
Particle tracking – multiple particles	U-nets	[30]
Particle classification	CNNs, U-nets	[30], [66]
Optical force calculations	DNNs	[38], [39]
Trajectory analysis – single particle	RNNS, ATNs	[40]
Trajectory analysis – multiple particles	GNNs	[130]
Calibration	RNNS, ATNs	[40]
Designing tweezers	Simulated annealing, VAEs	[127], [128]
Tweezers control – particle movement	DRL	[122]

In the case of tracking particles with digital video microscopy, the most commonly used architectures are variants of CNNs. If the goal is to track a single optically trapped particle, a standard CNN is often sufficient [41], [109], [114]. However, if many particles need to be tracked simultaneously, then using a U-net is often better than a standard CNN [30].

In the case of trajectory analysis and calibration, an architecture that can handle the time series data is required [40], [115]. RNNs have been used previously and will often suffice [40]. Also, TGANs and ATNs can perform well with various time series and are, therefore, a good option when there are missing data points or complex dependencies in the data. However, if one has a large number of particles that interact, then a GNN is a good choice – as demonstrated by the MAGIK algorithm [130].

To calculate optical forces, DNNs have been shown to work well [38], [39], [117], [118] and should therefore be the starting point. If the number of input parameters is small (up to a few tens, e.g., the particle position, rotation, and a limited number of parameters describing its shape), then a DNN will almost certainly perform well. Instead, when the number of parameters increases, such as in the case of biological cells which are also deformable, CNNs may be a better choice due to their capacity to capture spatial dependencies and their lower number of fitting parameters.

When deciding on an algorithm to use for controlling optical tweezers, the choice naturally falls on DRL [122], digital twins, and Bayesian modeling. However, the specific architecture to use is less obvious and depends on the input data.

Designing optical tweezers with deep learning is an area in which there has not been much research yet, but we believe that generative models, such as GANs and DMs, might be appropriate to deal with the need to generate different designs to find the most efficient one.

There are also cases when one wishes to combine different data types, for example when acquired by different sensors in the same experimental setup. In this case, one option is to use separate models for the different data types, but this restricts the algorithm by not giving the full picture preventing it from investigating correlations between the two different data streams. A superior option is to use hybrid models which combine several architectures. For instance, to handle a time series from a photodiode in combination with images from a camera, one can combine an RNN and a CNN as backbones to make the prediction using a DRL network as a head.

5.4 Training

Training consists of adjusting the parameters of a deep learning model to enhance its performance on the specific problem to solve. It is convenient to use a standard library to implement the models. The two most commonly used are PyTorch [35] (which has been on the rise for several years) and Keras/Tensorflow [36] (which is being slowly abandoned). Often, it is also possible to find already implemented architectures that can be used as a starting point for training your models. For example, the DeepTrack library [30], [131] offers an extensive toolkit for image analysis which has been shown to work well on microscopy data. The training process is often computationally demanding, which explains why we recommend running it on specialized hardware (e.g., using a GPU).

Before starting training, it is necessary to select loss, a performance metric that quantifies how far the model is from the ground truth, providing a quantity to be optimized. Therefore, the loss plays a fundamental role during the training process as its value quantifies the ability of the model to predict the real value of the desired parameter accurately. For example, this can be the square distance

between a predicted position and the actual position of a trapped particle, or the proportion of correctly classified samples.

Next, the initialization of the parameters is done, often automatically by the deep-learning framework being employed. Then, the training loop starts. In each iteration, known as an epoch, the training data are split into small batches on which the model is evaluated, and the loss is calculated. The loss is used with an optimization algorithm such as stochastic gradient descent to slightly change the weights of the model to minimize the loss value. Parallel to this, the value of the loss function is calculated on the validation set to see how well the model generalizes its prediction.

Generally, the performance of the model will increase epoch by epoch, but only up to a certain point when measured on the validation set. Afterwards, the validation performance tends to drop due to overfitting. It can be hard to tell for sure if a model is overfitting; generally, the more parameters the model has and the smaller the dataset, the larger the risk of overfitting. To avoid overfitting, it is possible to stop the training when the performance on the validation set has plateaued and before it starts worsening. Often, tuning of the hyperparameters, such as the number of layers in a CNN, optimizes the results and, also, reduces the risk of overfitting.

5.5 Testing

The final step is to test the model to ensure that it performs as desired when applied to new, never-seen-before data. By using as input to the model a validation dataset for which are known the desired outputs, the model output is compared with the expected one. If the performance is satisfactory, then the training process is finished. If the model has been trained on simulated data, then it is at this stage that the model is tested against real-world data or in an experimental setting. However, often the performance is not as good as desired. If the performance on simulated data is significantly better than that on real-world data, this may indicate a discrepancy between the simulations and the experiment. Similar problems may occur if the training data are experimental but gathered under different conditions (e.g., a different setup or with a different type of sample). If this happens, it is mandatory to train the model again by using a larger or more representative training dataset.

When employing the model in a real-time experimental setting, there is often a need for the model to make its predictions quickly. To achieve the required computational speed (especially when using the model in embedded systems, such as microcontrollers or FPGAs), connections or entire neurons may be removed from the neural network to reduce the size and increase the speed. This operation is called pruning. The aim is to strike an optimal trade-off between speed and accuracy for a real-time application and this requires further testing.

6 Conclusions

In this perspective, we investigate the application of deep learning for the optical tweezers field. As examples, we discuss the improvements in particle tracking at low signal-to-noise levels [41] and in quantifying the rotation of trapped particles [111]. Furthermore, we highlight the use of deep learning to address cases that traditional methods cannot deal with, such as accurately tracking multiple particles when they are close together, filling in missed frames in videos, or selectively tracking particles with unique characteristics, such as irregularly shaped particles or biological samples.

Then, we discuss the enhancement of trajectory analysis and optical tweezers calibration, which permit one to estimate rheological properties with only a few seconds of data instead of minutes [113] and, also, to discern different typologies of particles [114]. Moreover, we propose to use deep learning in some cases when standard methods fail: DMs may help to reconstruct trajectories with missing data points and estimate the desired properties; ATNs may help to determine whether a trapped particle is in thermal equilibrium or not.

Furthermore, deep learning has already improved the calculation of optical forces by increasing the computational speed and accuracy [39] and by allowing the study of non-trivial cases, such as with Laguerre-Gaussian beams [38], or with non-spherical particles like cells. In this scenario, optical forces could be calculated also in cases where standard methods are not viable. Indeed, DMs and GANs can calculate the force field starting from intensity images of the optical field, while CNNs can do the same from the design of a near-field optical trap.

When real-time control of optical tweezers is necessary, standard methods are often too computationally slow. Recently, NNs have allowed moving a trapped particle to a target position while avoiding collisions with real and virtual obstacles [122]. We believe that real-time control and automatization of optical tweezers can be further improved using deep learning. Digital twins with VAEs may be suitable when the automatic trapping of specific particles is desired. DRL with Bayesian modeling may automate experiments, such as DNA pulling, optimizing the search for favorable

experimental conditions. U-nets with Bayesian modeling may help automate the process of designing microstructures with optimal optical manipulation properties.

Designing optical tweezers can be challenging, especially when more complex designs are required. Deep learning can provide an effective solution for these requirements. Although the design of optical tweezers has thus far only utilized probabilistic methods like simulated annealing [127], deep learning has the potential to enhance this process. For example, DMs can design optical tweezers with a spatial light modulator for trapping multiple particles while enhancing the trapping force.

Finally, we provide guidelines for using deep learning in optical trapping and optical manipulation, highlighting step-by-step the process to follow to create an effective deep learning model, from the problem description to the model validation, while avoiding common pitfalls.

Acknowledgments: We would like to thank Agnese Callegari and Caroline B. Adiels for their helpful discussions about the format of this article, which significantly improved the final manuscript.

Research funding: We acknowledge support from the MSCA-ITN-ETN project ActiveMatter sponsored by the European Commission (Horizon 2020, Project No. 812780), the Horizon Europe ERC Consolidator Grant MAPEI (grant number 101001267), and the Knut and Alice Wallenberg Foundation (grant number 2019.0079), the European Union (NextGeneration EU), through the MUR-PNRR project SAMOTHRACE (ECS00000022), the PNRR MUR project PE0000023-NQSTI, and the PRIN2022 "Cosmic Dust II" (grant number 2022S5A2N7).

Author contributions: All authors have accepted responsibility for the entire content of this manuscript and approved its submission.

Conflict of interest: Authors state no conflicts of interest.

Data availability: Data sharing is not applicable to this article as no datasets were generated or analyzed during the current study.

References

[1] A. Ashkin, "Acceleration and trapping of particles by radiation pressure," *Phys. Rev. Lett.*, vol. 24, no. 4, p. 156, 1970.

[2] A. Ashkin, "Atomic-beam deflection by resonance-radiation pressure," *Phys. Rev. Lett.*, vol. 25, no. 19, p. 1321, 1970.

[3] A. Ashkin and J. Dziedzic, "Feedback stabilization of optically levitated particles," *Appl. Phys. Lett.*, vol. 30, no. 4, p. 202, 1977.

[4] A. Ashkin, J. M. Dziedzic, J. E. Bjorkholm, and S. Chu, "Observation of a single-beam gradient force optical trap for dielectric particles," *Opt. Lett.*, vol. 11, no. 5, p. 288, 1986.

[5] P. Jones, O. Maragó, and G. Volpe, *Optical Tweezers: Principles and Applications*, Cambridge, UK, Cambridge University Press, 2015.

[6] G. Volpe, *et al*., "Roadmap for optical tweezers," arXiv preprint arXiv:2206.13789, 2022.

[7] L. I. McCann, M. Dykman, and B. Golding, "Thermally activated transitions in a bistable three-dimensional optical trap," *Nature*, vol. 402, no. 6763, pp. 785—787, 1999.

[8] C. Bechinger, M. Brunner, and P. Leiderer, "Phase behavior of two-dimensional colloidal systems in the presence of periodic light fields," *Phys. Rev. Lett.*, vol. 86, no. 5, pp. 930—933, 2001.

[9] A. Ciarlo, R. Pastore, F. Greco, A. Sasso, and G. Pesce, "Fickian yet non-Gaussian diffusion of a quasi-2d colloidal system in an optical speckle field: experiment and simulations," *Sci. Rep.*, vol. 13, no. 1, p. 7408, 2023.

[10] R. Pastore, A. Ciarlo, G. Pesce, A. Sasso, and F. Greco, "A model-system of fickian yet non-Gaussian diffusion: light patterns in place of complex matter," *Soft Matter*, vol. 18, no. 2, pp. 351—364, 2022.

[11] R. Pastore, A. Ciarlo, G. Pesce, F. Greco, and A. Sasso, "Rapid fickian yet non-Gaussian diffusion after subdiffusion," *Phys. Rev. Lett.*, vol. 126, no. 15, p. 158003, 2021.

[12] J. Gieseler and J. Millen, "Levitated nanoparticles for microscopic thermodynamics—a review," *Entropy*, vol. 20, no. 5, p. 326, 2018.

[13] H. Löwen, "Colloidal soft matter under external control," *J. Phys. Condens. Matter*, vol. 13, no. 24, p. R415, 2001.

[14] D. V. Petrov, "Raman spectroscopy of optically trapped particles," *J. Opt. A Pure Appl. Opt.*, vol. 9, no. 8, p. S139, 2007.

[15] C. J. Bustamante, Y. R. Chemla, S. Liu, and M. D. Wang, "Optical tweezers in single-molecule biophysics," *Nat. Rev. Methods Primers*, vol. 1, no. 1, p. 25, 2021.

[16] A. Ashkin and J. M. Dziedzic, "Optical trapping and manipulation of viruses and bacteria," *Science*, vol. 235, no. 4795, pp. 1517—1520, 1987.

[17] T. N. Buican, M. J. Smyth, H. A. Crissman, G. C. Salzman, C. C. Stewart, and J. C. Martin, "Automated single-cell manipulation and sorting by light trapping," *Appl. Opt.*, vol. 26, no. 24, pp. 5311—5316, 1987.

[18] A. Ashkin and J. Dziedzic, "Optical trapping and manipulation of single living cells using infra-red laser beams," *Ber. Bunsengesellschaft Phys. Chem.*, vol. 93, no. 3, pp. 254—260, 1989.

[19] A. Ashkin and J. Dziedzic, "Internal cell manipulation using infrared laser traps," *Proc. Natl. Acad. Sci. U. S. A.*, vol. 86, no. 20, pp. 7914—7918, 1989.

[20] S. M. Block, L. S. Goldstein, and B. J. Schnapp, "Bead movement by single kinesin molecules studied with optical tweezers," *Nature*, vol. 348, no. 6299, pp. 348—352, 1990.

[21] J. T. Finer, R. M. Simmons, and J. A. Spudich, "Single myosin molecule mechanics: piconewton forces and nanometre steps," *Nature*, vol. 368, no. 6467, pp. 113—119, 1994.

[22] R. E. Holmlin, M. Schiavoni, C. Y. Chen, S. P. Smith, M. G. Prentiss, and G. M. Whitesides, "Light-driven microfabrication: assembly of multicomponent, three-dimensional structures by using optical tweezers," *Angew. Chem., Int. Ed.*, vol. 39, no. 19, pp. 3503—3506, 2000. Available at: .

[23] R. Agarwal, K. Ladavac, Y. Roichman, G. Yu, C. M. Lieber, and D. G. Grier, "Manipulation and assembly of nanowires with holographic optical traps," *Opt. Express*, vol. 13, no. 22, pp. 8906—8912, 2005.

[24] R. Grimm, M. Weidemüller, and Y. B. Ovchinnikov, "Optical dipole traps for neutral atoms," *Adv. Atom. Mol. Opt. Phys.*, vol. 42, pp. 95—170, 2000.
[25] T. Gustavson, *et al*., "Transport of bose-einstein condensates with optical tweezers," *Phys. Rev. Lett.*, vol. 88, no. 2, p. 020401, 2001.
[26] D. Meschede and A. Rauschenbeutel, "Manipulating single atoms," *Adv. Atom. Mol. Opt. Phys.*, vol. 53, pp. 75—104, 2006.
[27] G. Volpe, R. Quidant, G. Badenes, and D. Petrov, "Surface plasmon radiation forces," *Phys. Rev. Lett.*, vol. 96, no. 23, p. 238101, 2006.
[28] M. L. Juan, M. Righini, and R. Quidant, "Plasmon nano-optical tweezers," *Nat. Photonics*, vol. 5, no. 6, pp. 349—356, 2011.
[29] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436—444, 2015.
[30] B. Midtvedt, S. Helgadottir, A. Argun, J. Pineda, D. Midtvedt, and G. Volpe, "Quantitative digital microscopy with deep learning," *Appl. Phys. Rev.*, vol. 8, no. 1, pp. 011310-1-011310-22, 2021.
[31] J. Jumper, *et al*., "Highly accurate protein structure prediction with alphafold," *Nature*, vol. 596, no. 7873, pp. 583—589, 2021.
[32] S. Balaban, "Deep learning and face recognition: the state of the art," in *Biometric and Surveillance Technology for Human and Activity Identification XII*, vol. 9457, 2015, p. 68.
[33] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *Bull. Math. Biophys.*, vol. 5, pp. 115—133, 1943.
[34] A. Paszke, *et al*., "Automatic differentiation in pytorch," in *31st Conference on Neural Information Processing Systems*, 2017, pp. 1—4.
[35] A. Paszke, *et al*., "Pytorch: an imperative style, high-performance deep learning library," *Adv. Neural Inf. Process. Syst.*, vol. 32, pp. 8024—8035, 2019.
[36] M. Abadi, *et al*., "TensorFlow: large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org.
[37] F. Chollet, *et al*., *Keras*, 2015. Available at: https://github.com/fchollet/keras.
[38] I. C. Lenton, G. Volpe, A. B. Stilgoe, T. A. Nieminen, and H. Rubinsztein-Dunlop, "Machine learning reveals complex behaviours in optically trapped particles," *Mach. Learn. Sci. Technol.*, vol. 1, no. 4, p. 045009, 2020.
[39] D. Bronte Ciriza, *et al*., "Faster and more accurate geometrical-optics optical force calculation using neural networks," *ACS Photonics*, vol. 10, no. 1, pp. 234—241, 2022.
[40] A. Argun, T. Thalheim, S. Bo, F. Cichos, and G. Volpe, "Enhanced force-field calibration via machine learning," *Appl. Phys. Rev.*, vol. 7, no. 4, p. 041404, 2020.
[41] S. Helgadottir, A. Argun, and G. Volpe, "Digital video microscopy enhanced by deep learning," *Optica*, vol. 6, no. 4, pp. 506—513, 2019.
[42] T. Aggarwal and M. Salapaka, "Real-time nonlinear correction of back-focal-plane detection in optical tweezers," *Rev. Sci. Instrum.*, vol. 81, no. 12, p. 123105, 2010.
[43] N. Li, J. Cadusch, and K. Crozier, "Algorithmic approach for designing plasmonic nanotweezers," *Opt. Lett.*, vol. 44, no. 21, pp. 5250—5253, 2019.
[44] G. Binnig, C. F. Quate, and C. Gerber, "Atomic force microscope," *Phys. Rev. Lett.*, vol. 56, no. 9, p. 930, 1986.
[45] G. Pesce, P. H. Jones, O. M. Maragò, and G. Volpe, "Optical tweezers: theory and practice," *Eur. Phys. J. Plus*, vol. 135, no. 949, pp. 1—38, 2020.
[46] A. Ashkin, "Forces of a single-beam gradient laser trap on a dielectric sphere in the ray optics regime," *Biophys. J.*, vol. 61, no. 2, pp. 569—582, 1992.
[47] A. Callegari, M. Mijalkov, A. B. Gököz, and G. Volpe, "Computational toolbox for optical tweezers in geometrical optics," *JOSA B*, vol. 32, no. 5, pp. B11—B19, 2015.
[48] P. C. Chaumet and M. Nieto-Vesperinas, "Time-averaged total force on a dipolar sphere in an electromagnetic field," *Opt. Lett.*, vol. 25, no. 15, pp. 1065—1067, 2000.
[49] F. Borghese, P. Denti, and R. Saija, *Scattering from Model Nonspherical Particles: Theory and Applications to Environmental Physics*, Heidelberg, Springer Science & Business Media, 2007.
[50] M. I. Mishchenko, L. D. Travis, and A. A. Lacis, *Multiple Scattering of Light by Particles: Radiative Transfer and Coherent Backscattering*, Cambridge, Cambridge University Press, 2006.
[51] T. A. Nieminen, *et al*., "Optical tweezers computational toolbox," *J. Opt. A Pure Appl. Opt.*, vol. 9, no. 8, p. S196, 2007.
[52] E.-L. Florin, A. Pralle, E. Stelzer, and J. Hörber, "Photonic force microscope calibration by thermal noise analysis," *Appl. Phys. A*, vol. 66, pp. S75—S78, 1998.
[53] N. Viana, R. Freire, and O. Mesquita, "Dynamic light scattering from an optically trapped microsphere," *Phys. Rev. E*, vol. 65, no. 4, p. 041921, 2002.
[54] K. Berg-Sørensen and H. Flyvbjerg, "Power spectrum analysis for optical tweezers," *Rev. Sci. Instrum.*, vol. 75, no. 3, pp. 594—612, 2004.
[55] L. Pérez García, J. Donlucas Pérez, G. Volpe, A. V. Arzola, and G. Volpe, "High-performance reconstruction of microscopic force fields from brownian trajectories," *Nat. Commun.*, vol. 9, no. 1, p. 5166, 2018.
[56] R. M. Simmons, J. T. Finer, S. Chu, and J. A. Spudich, "Quantitative measurements of force and displacement using an optical trap," *Biophys. J.*, vol. 70, no. 4, pp. 1813—1822, 1996.
[57] A. E. Wallin, H. Ojala, E. Hæggström, and R. Tuma, "Stiffer optical tweezers through real-time feedback control," *Appl. Phys. Lett.*, vol. 92, no. 22, p. 224104, 2008.
[58] F. Kalantarifard, P. Elahi, G. Makey, O. M. Maragò, F. Ö. Ilday, and G. Volpe, "Intracavity optical trapping of microscopic particles in a ring-cavity fiber laser," *Nat. Commun.*, vol. 10, no. 1, p. 2683, 2019.
[59] C. Lu and X. Tang, "Surpassing human-level face verification performance on lfw with gaussianface," *Proc. AAAI Conf. Artif. Intell.*, vol. 29, no. 1, pp. 3811—3819, 2015.
[60] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533—536, 1986.
[61] K. Fukushima, "Neocognitron: a self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position," *Biol. Cybern.*, vol. 36, no. 4, pp. 193—202, 1980.
[62] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278—2324, 1998.
[63] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Adv. Neural Inf. Process. Syst.*, vol. 25, no. 1, pp. 1—9, 2012.
[64] R. Chauhan, K. K. Ghanshala, and R. Joshi, "Convolutional neural network (cnn) for image detection and recognition," in *2018 First International Conference on Secure Cyber Computing and Communication (ICSCCC)*, IEEE, 2018, pp. 278—282.

[65] J. Bullock, C. Cuesta-Lázaro, and A. Quera-Bofarull, "Xnet: a convolutional neural network (cnn) implementation for medical x-ray image segmentation suitable for small datasets," in *Medical Imaging 2019: Biomedical Applications in Molecular, Structural, and Functional Imaging*, vol. 10953, SPIE, 2019, pp. 453–463.

[66] O. Ronneberger, P. Fischer, and T. Brox, "U-net: convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5–9, 2015, Proceedings, Part III 18*, Springer, 2015, pp. 234–241.

[67] C. C. Shokiche, P. Baumann, R. Hlushchuk, V. Djonov, and M. Reyes, "High-throughput glomeruli analysis of ct kidney images using tree priors and scalable sparse computation," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*, Springer, 2016, pp. 370–378.

[68] A. Vojtekova, *et al*., "Learning to denoise astronomical images with u-nets," *Mon. Not. R. Astron. Soc.*, vol. 503, no. 3, pp. 3204–3215, 2021.

[69] S. S. Bangaru, C. Wang, X. Zhou, and M. Hassan, "Scanning electron microscopy (sem) image segmentation for microstructure analysis of concrete using u-net convolutional neural network," *Autom. Construct.*, vol. 144, p. 104602, 2022.

[70] D. E. Rumelhart, *et al*., "Learning Internal Representations by Error Propagation," in *Parallel Distributed Processing, Volume 1: Explorations in the Microstructure of Cognition: Foundations*, Cambridge, Massachusetts, The MIT Press, 1986, pp. 318–362. https://doi.org/10.21236/ada164453.

[71] B. Mehlig, *Machine Learning with Neural Networks: An Introduction for Scientists and Engineers*, Cambridge, Cambridge University Press, 2021.

[72] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[73] K. Cho, *et al*., "Learning phrase representations using rnn encoder-decoder for statistical machine translation," arXiv preprint arXiv:1406.1078, 2014, https://doi.org/10.3115/v1/d14-1179,

[74] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," *Adv. Neural Inf. Process. Syst.*, vol. 27, no. 1, pp. 1–9, 2014.

[75] T. Thireou and M. Reczko, "Bidirectional long short-term memory networks for predicting the subcellular localization of eukaryotic proteins," *IEEE ACM Trans. Comput. Biol. Bioinf.*, vol. 4, no. 3, pp. 441–446, 2007.

[76] S. Hochreiter, M. Heusel, and K. Obermayer, "Fast model-based protein homology detection without alignment," *Bioinformatics*, vol. 23, no. 14, pp. 1728–1736, 2007.

[77] J. Kugelman, D. Alonso-Caneiro, S. A. Read, S. J. Vincent, and M. J. Collins, "Automatic segmentation of oct retinal boundaries using recurrent neural networks and graph search," *Biomed. Opt. Express*, vol. 9, no. 11, pp. 5759–5777, 2018.

[78] R. Landman, S. Y. Haffert, V. M. Radhakrishnan, and C. U. Keller, "Self-optimizing adaptive optics control with reinforcement learning," in *Adaptive Optics Systems VII*, vol. 11448, SPIE, 2020, pp. 842–856.

[79] A. Vaswani, *et al*., "Attention is all you need," *Adv. Neural Inf. Process. Syst.*, vol. 30, no. 1, pp. 1–11, 2017.

[80] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: pre-training of deep bidirectional transformers for language understanding," arXiv preprint arXiv:1810.04805, 2018.

[81] T. Brown, *et al*., "Language models are few-shot learners," *Adv. Neural Inf. Process. Syst.*, vol. 33, pp. 1877–1901, 2020.

[82] N. Parmar, *et al*., "Image transformer," in *International Conference on Machine Learning*, PMLR, 2018, pp. 4055–4064.

[83] M. Gori, G. Monfardini, and F. Scarselli, "A new model for learning in graph domains," in *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005*, vol. 2, IEEE, 2005, pp. 729–734.

[84] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE Trans. Neural Netw.*, vol. 20, no. 1, pp. 61–80, 2008.

[85] C. Gallicchio and A. Micheli, "Graph echo state networks," in *The 2010 International Joint Conference on Neural Networks (IJCNN)*, IEEE, 2010, pp. 1–8.

[86] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," arXiv preprint arXiv:1609.02907, 2016.

[87] Y. Li, R. Yu, C. Shahabi, and Y. Liu, "Diffusion convolutional recurrent neural network: data-driven traffic forecasting," arXiv preprint arXiv:1707.01926, 2017.

[88] A. Fout, J. Byrd, B. Shariat, and A. Ben-Hur, "Protein interface prediction using graph convolutional networks," *Adv. Neural Inf. Process. Syst.*, vol. 30, no. 1, pp. 1–10, 2017.

[89] A. Creswell, T. White, V. Dumoulin, K. Arulkumaran, B. Sengupta, and A. A. Bharath, "Generative adversarial networks: an overview," *IEEE Signal Process. Mag.*, vol. 35, no. 1, pp. 53–65, 2018.

[90] J. Yoon, D. Jarrett, and M. Van der Schaar, "Time-series generative adversarial networks," *Adv. Neural Inf. Process. Syst.*, vol. 32, no. 1, pp. 1–11, 2019.

[91] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 1125–1134.

[92] C. Ledig, *et al*., "Photo-realistic single image super-resolution using a generative adversarial network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 4681–4690.

[93] T. Schlegl, P. Seeböck, S. M. Waldstein, U. Schmidt-Erfurth, and G. Langs, "Unsupervised anomaly detection with generative adversarial networks to guide marker discovery," in *International Conference on Information Processing in Medical Imaging*, Springer, 2017, pp. 146–157.

[94] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," arXiv preprint arXiv:1312.6114, 2013.

[95] I. A. Luchnikov, A. Ryzhov, P.-J. Stas, S. N. Filippov, and H. Ouerdane, "Variational autoencoder reconstruction of complex many-body physics," *Entropy*, vol. 21, no. 11, p. 1091, 2019.

[96] Q. Zhao, E. Adeli, N. Honnorat, T. Leng, and K. M. Pohl, "Variational autoencoder for regression: application to brain aging analysis," in *Medical Image Computing and Computer Assisted Intervention–MICCAI 2019: 22nd International Conference, Shenzhen, China, October 13–17, 2019, Proceedings, Part II 22*, Springer, 2019, pp. 823–831.

[97] J. A. Hennig, A. Umakantha, and R. C. Williamson, "A classifying variational autoencoder with application to polyphonic music generation," arXiv preprint arXiv:1711.07050, 2017.

[98] J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli, "Deep unsupervised learning using nonequilibrium

thermodynamics," in *International Conference on Machine Learning*, PMLR, 2015, pp. 2256—2265.
[99] C. Saharia, *et al*., "Photorealistic text-to-image diffusion models with deep language understanding," *Adv. Neural Inf. Process. Syst.*, vol. 35, pp. 36479—36494, 2022.
[100] J. Ho, C. Saharia, W. Chan, D. J. Fleet, M. Norouzi, and T. Salimans, "Cascaded diffusion models for high fidelity image generation," *J. Mach. Learn. Res.*, vol. 23, no. 47, pp. 1—33, 2022.
[101] W. H. Pinaya, *et al*., "Brain imaging generation with latent diffusion models," in *MICCAI Workshop on Deep Generative Models*, Springer, 2022, pp. 117—126.
[102] B. Kawar, *et al*., "Imagic: text-based real image editing with diffusion models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 6007—6017.
[103] Y. Liu, R. Guan, F. Giunchiglia, Y. Liang, and X. Feng, "Deep attention diffusion graph neural networks for text classification," in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 2021, pp. 8142—8152.
[104] V. Mnih, *et al*., "Playing atari with deep reinforcement learning," arXiv preprint arXiv:1312.5602, 2013.
[105] V. Mnih, *et al*., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529—533, 2015.
[106] D. Silver, *et al*., "Mastering the game of go with deep neural networks and tree search," *nature*, vol. 529, no. 7587, pp. 484—489, 2016.
[107] A. E. Sallab, M. Abdou, E. Perot, and S. Yogamani, "End-to-end deep reinforcement learning for lane keeping assist," arXiv preprint arXiv:1612.04340, 2016.
[108] H. Wang, Z. Zheng, C. Ji, and L. J. Guo, "Automated multi-layer optical design via deep reinforcement learning," *Mach. Learn. Sci. Technol.*, vol. 2, no. 2, p. 025013, 2021.
[109] J. M. Newby, A. M. Schaefer, P. T. Lee, M. G. Forest, and S. K. Lai, "Convolutional neural networks automate detection for tracking of submicron-scale particles in 2d and 3d," *Proc. Natl. Acad. Sci. U. S. A.*, vol. 115, no. 36, pp. 9026—9031, 2018.
[110] B. Midtvedt, *et al*., "Single-shot self-supervised particle tracking," arXiv preprint arXiv:2202.13546, 2022.
[111] J. Zhao, C. Bai, Z. Zhang, and Q. Zhang, "Deep learning-based method for analyzing the optically trapped sperm rotation," *Sci. Rep.*, vol. 13, no. 1, p. 12575, 2023.
[112] J. Baumgartl and C. Bechinger, "On the limits of digital video microscopy," *Europhys. Lett.*, vol. 71, no. 3, p. 487, 2005.
[113] M. G. Smith, *et al*., "Machine learning opens a doorway for microrheology with optical tweezers in living systems," arXiv preprint arXiv:2211.09689, 2022.
[114] I. A. Carvalho, N. A. Silva, C. C. Rosa, L. C. Coelho, and P. A. Jorge, "Particle classification through the analysis of the forward scattered signal in optical tweezers," *Sensors*, vol. 21, no. 18, p. 6181, 2021.
[115] L. Hamilton, *et al*., *Predicting Particle Properties in Optical Traps with Machine Learning*, California, US, SPIE-Intl Soc Optical Eng, 2020, p. 70.
[116] G. Volpe and G. Volpe, "Simulation of a brownian particle in an optical trap," *Am. J. Phys.*, vol. 81, no. 3, pp. 224—230, 2013.
[117] R. Tognato, D. Bronte-Ciriza, O. M. Maragò, and P. H. Jones, "Modelling red blood cell optical trapping by machine learning improved geometrical optics calculations," *Biomed. Opt. Express*, vol. 14, no. 7, pp. 3748—3762, 2023.
[118] K. S. Malik and B. R. Boruah, "Optical force calculation in the ray-optics regime for beams with arbitrary complex amplitude profiles," *Opt. Lett.*, vol. 47, no. 16, pp. 4151—4154, 2022.
[119] F. Evers, *et al*., "Particle dynamics in two-dimensional random-energy landscapes: experiments and simulations," *Phys. Rev. E*, vol. 88, no. 2, p. 022125, 2013.
[120] G. Volpe, G. Volpe, and S. Gigan, "Brownian motion in a speckle light field: tunable anomalous diffusion and selective optical manipulation," *Sci. Rep.*, vol. 4, no. 1, p. 3936, 2014.
[121] G. Volpe, L. Kurz, A. Callegari, G. Volpe, and S. Gigan, "Speckle optical tweezers: micromanipulation with random light fields," *Opt. Express*, vol. 22, no. 15, pp. 18159—18167, 2014.
[122] M. Praeger, Y. Xie, J. A. Grant-Jacob, R. W. Eason, and B. Mills, "Playing optical tweezers with deep reinforcement learning: in virtual, physical and augmented environments," *Mach. Learn. Sci. Technol.*, vol. 2, no. 3, p. 035024 1-11, 2021.
[123] D. Gelernter, *Mirror Worlds: Or the Day Software Puts the Universe in a Shoebox…How it Will Happen and What it Will Mean*, New York, Oxford University Press, 1993.
[124] M. Grieves, "Completing the cycle: using plm information in the sales and service functions [slides]," in *SME Management Forum*, 2002.
[125] E. Glaessgen and D. Stargel, "The digital twin paradigm for future nasa and us air force vehicles," in *53rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference 20th AIAA/ASME/AHS Adaptive Structures Conference 14th AIAA*, 2012, p. 1818.
[126] T. Bayes, "LII. An essay towards solving a problem in the doctrine of chances. By the late Rev. Mr. Bayes, F. R. S. communicated by Mr. Price, in a letter to John Canton, A. M. F. R. S," *Phil. Trans. Roy. Soc. Lond.*, vol. 53, pp. 370—418, 1763.
[127] N. Li, J. Cadusch, A. Liu, A. J. Barlow, A. Roberts, and K. B. Crozier, "Algorithm-designed plasmonic nanotweezers: quantitative comparison by theory, cathodoluminescence, and nanoparticle trapping," *Adv. Opt. Mater.*, vol. 9, no. 19, p. 2100758, 2021.
[128] P. R. Wiecha, A. Arbouet, C. Girard, and O. L. Muskens, "Deep learning in nano-photonics: inverse design and beyond," *Photon. Res.*, vol. 9, no. 5, pp. B182—B200, 2021.
[129] J. E. Curtis, B. A. Koss, and D. G. Grier, "Dynamic holographic optical tweezers," *Opt. Commun.*, vol. 207, no. 1—6, pp. 169—175, 2002.
[130] J. Pineda, *et al*., "Geometric deep learning reveals the spatiotemporal features of microscopic motion," *Nat. Mach. Intell.*, vol. 5, no. 1, pp. 71—82, 2023.
[131] B. Midtvedt, *et al*., DeepTrack2, 2024. Available at: https://github.com/deeptrackai/deeptrack2.